

Modus: Alle Zeilen

Linker Ursprungsordner: W:\BCAST\OMA\CRs\own\OMA-IOP-TTCN-2008-0014R01-CR\_BCAST\_ATS\_Corrections\ttcn

Rechter Ursprungsordner: W:\BCAST\OMA\CRs\own\OMA-IOP-TTCN-2008-0014R01-CR\_BCAST\_ATS\_Corrections\old

Datei: AtsBCast\_ContentProtectionTests.ttcn

<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies pilot tests for BCast content protection.  */ module AtsBCast_ContentProtectionTests {     import from LibBCast_Interface all;     import from AtsBCast_Main_Functions all;     import from LibBCast_ServiceGuide_Templates all;     import from AtsBCast_ServiceGuide_Functions all;     import from LibBCast_ServiceGuide_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_BasicTypesAndValues all;     import from AtsBCast_ModuleParameters all;     import from LibBCast_ModuleParameters all;     import from AtsBCast_TestConfiguration_Functions all;     import from AtsBCast_TestSystem all;      group bcastConformanceTestCases {         group clientConformanceTestCases {              /**              *              * @desc              *   &lt;p&gt;Service protection with IPSec via broadcast » channel (optional)&lt;/p&gt;              *   &lt;p&gt;Test Case Description&lt;/p&gt;              *   &lt;p&gt;with a terminal having received a service » guide containing a service with purchase information on the Service Guide » Delivery Channel and listening for an IPsec encrypted data stream on the » broadcast channel&lt;/p&gt;              *   &lt;p&gt;when the terminal subscribes to that service » on the interactive channel using a Service Request/Response message » exchange&lt;/p&gt;              *   &lt;p&gt;then the terminal is able to receive, » decrypt, and display the data stream correctly.&lt;/p&gt;              *   &lt;p&gt;Specification Reference&lt;/p&gt;              *   &lt;p&gt;[BCAST10-ServContProt] Section 6.8.1, </pre>	=	<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies pilot tests for BCast content protection.  */ module AtsBCast_ContentProtectionTests {     import from LibBCast_Interface all;     import from AtsBCast_Main_Functions all;     import from LibBCast_ServiceGuide_Templates all;     import from AtsBCast_ServiceGuide_Functions all;     import from LibBCast_ServiceGuide_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_BasicTypesAndValues all;     import from AtsBCast_ModuleParameters all;     import from LibBCast_ModuleParameters all;     import from AtsBCast_TestConfiguration_Functions all;     import from AtsBCast_TestSystem all;      group bcastConformanceTestCases {         group clientConformanceTestCases {              /**              *              * @desc              *   &lt;p&gt;Service protection with IPSec via broadcast » channel (optional)&lt;/p&gt;              *   &lt;p&gt;Test Case Description&lt;/p&gt;              *   &lt;p&gt;with a terminal having received a service » guide containing a service with purchase information on the Service Guide » Delivery Channel and listening for an IPsec encrypted data stream on the » broadcast channel&lt;/p&gt;              *   &lt;p&gt;when the terminal subscribes to that service » on the interactive channel using a Service Request/Response message » exchange&lt;/p&gt;              *   &lt;p&gt;then the terminal is able to receive, » decrypt, and display the data stream correctly.&lt;/p&gt;              *   &lt;p&gt;Specification Reference&lt;/p&gt;              *   &lt;p&gt;[BCAST10-ServContProt] Section 6.8.1, </pre>
---	---	---

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

» 9.1.</p>	» 9.1.</p>
<pre> * &lt;p&gt;Preconditions&lt;/p&gt; * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix: * &lt;ul&gt; * &lt;li&gt;Service fragment with » Name="PayTvChannel "&lt;/li&gt; * &lt;li&gt;Content fragment with » Name="Programme", and StartTime and EndTime elements indicating values » within the time of test&lt;/li&gt; * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt; * &lt;li&gt;Access fragment for schedule » fragment with KeyManagmentSystem, EncryptionType set to IPsec, and a » reference to the Session Description fragment &lt;/li&gt; * &lt;li&gt;Session Description fragment » containing SDP for "Programme" and the service protection information for » IPsec&lt;/li&gt; * &lt;li&gt;PurchaseItem fragment for service » fragment with Description "IPsec" element&lt;/li&gt; * &lt;li&gt;PurchaseData fragment for » purchaseItem fragment with PriceInfo and Description "Discount" » elements&lt;/li&gt; * &lt;li&gt;PurchaseChannel fragment for » purchaseItem fragment with PurchaseURL pointing to a subscription site&lt;/li&gt; * &lt;/ul&gt; * &lt;p&gt; * &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; * &lt;p&gt;Test Procedure&lt;/p&gt; * &lt;p&gt; * &lt;ul&gt; * &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using broadcast channel.&lt;/li&gt; * &lt;li&gt;Setup the test tool to stream via » IPsec encrypted audio and video for "Programme" as well as key stream on » broadcast channel&lt;/li&gt; * &lt;li&gt;Setup the SEK delivery to the » terminal either via DRM 2.0 ROAP (DRM) or authentication and push delivery </pre>	<pre> = * &lt;p&gt;Preconditions&lt;/p&gt; * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix: * &lt;ul&gt; * &lt;li&gt;Service fragment with » Name="PayTvChannel "&lt;/li&gt; * &lt;li&gt;Content fragment with » Name="Programme", and StartTime and EndTime elements indicating values » within the time of test&lt;/li&gt; * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt; * &lt;li&gt;Access fragment for schedule » fragment with KeyManagmentSystem, EncryptionType set to IPsec, and a » reference to the Session Description fragment &lt;/li&gt; * &lt;li&gt;Session Description fragment » containing SDP for "Programme" and the service protection information for » IPsec&lt;/li&gt; * &lt;li&gt;PurchaseItem fragment for service » fragment with Description "IPsec" element&lt;/li&gt; * &lt;li&gt;PurchaseData fragment for » purchaseItem fragment with PriceInfo and Description "Discount" » elements&lt;/li&gt; * &lt;li&gt;PurchaseChannel fragment for » purchaseItem fragment with PurchaseURL pointing to a subscription site&lt;/li&gt; * &lt;/ul&gt; * &lt;p&gt; * &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; * &lt;p&gt;Test Procedure&lt;/p&gt; * &lt;p&gt; * &lt;ul&gt; * &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using broadcast channel.&lt;/li&gt; * &lt;li&gt;Setup the test tool to stream via » IPsec encrypted audio and video for "Programme" as well as key stream on » broadcast channel&lt;/li&gt; * &lt;li&gt;Setup the SEK delivery to the » terminal either via DRM 2.0 ROAP (DRM) or authentication and push delivery </pre>

## Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

```

» (Smartcard) </li>
*           <li>Activate the BCAST application of
» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG in the terminal</li>
*           <li>Setup the test tool to authenticate
» and accept subscription request for the Service "PayTvChannel" from the
» terminal via interactive channel</li>
*           <li>Purchase "TvChannel" on
» terminal</li>
*           <li>Access "TvChannel" on terminal</li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible to the end
» user after the delivery of the SG
*           <ul>
*           <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information.</li>
*           <li>"Programme" can be viewed by the end
» user.</li>
*           </ul>
*           </p>
*           <p>The following should be observable for the
» test tool
*           <ul>
*           <li>The test tool receives HTTP request
» from the end user to subscribe to service "TvChannel".</li>
*           </ul>
*           </p>
*           @verdict
*           pass in case the client pass the conformance
» test.
*           @verdict
*           fail in case the client does not pass the
» conformance test.
*           @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_CONTPROT_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Interaction_up();

```

```

» (Smartcard) </li>
*           <li>Activate the BCAST application of
» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG in the terminal</li>
*           <li>Setup the test tool to authenticate
» and accept subscription request for the Service "PayTvChannel" from the
» terminal via interactive channel</li>
*           <li>Purchase "TvChannel" on
» terminal</li>
*           <li>Access "TvChannel" on terminal</li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible to the end
» user after the delivery of the SG
*           <ul>
*           <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information.</li>
*           <li>"Programme" can be viewed by the end
» user.</li>
*           </ul>
*           </p>
*           <p>The following should be observable for the
» test tool
*           <ul>
*           <li>The test tool receives HTTP request
» from the end user to subscribe to service "TvChannel".</li>
*           </ul>
*           </p>
*           @verdict
*           pass in case the client pass the conformance
» test.
*           @verdict
*           fail in case the client does not pass the
» conformance test.
*           @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_CONTPROT_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Interaction_up();

```

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre> f_cf_UpperTester_up();  // preamble f_startCommonUtPreamble(); // TODO: Omitted here is starting of » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue  // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>		<pre> f_cf_UpperTester_up();  // preamble f_startCommonUtPreamble(); // TODO: Omitted here is starting of » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue  // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>
<pre> f_main_ctrl_InitStartTime(c_one_hour,c_one » _hour);          // change 01 (WK 6): global time definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); // change 01 (WK 6): global time definition var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);   // change 01 (WK 6): global time definition </pre>	=	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime); var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(         PX_SDP_VIDEO_PROG_1_REF_ID,         PX_SDP_SERVICE_PROTECTION_IPSEC     ));  // TODO: Omitted here is setting of KMS value » - see open issues var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),         PX_SDP_VIDEO_PROG_1_REF_ID     ));  var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo(         1,         PX_MONETARY_PRICE,         PX_CURRENCY     ));  var ServiceType v_Service := » valueof(m_def_Service( </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(         PX_SDP_VIDEO_PROG_1_REF_ID,         PX_SDP_SERVICE_PROTECTION_IPSEC     ));  // TODO: Omitted here is setting of KMS value » - see open issues var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),         PX_SDP_VIDEO_PROG_1_REF_ID     ));  var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo(         1,         PX_MONETARY_PRICE,         PX_CURRENCY     ));  var ServiceType v_Service := » valueof(m_def_Service( </pre>

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre>                 PX_SGDU_SERVICE_ID,                 v_FragmentVersion,                 "PayTvChannel",                 c_basicTv_ServiceType             ));              var ContentType v_Content := » valueof(m_def_Content(                 PX_SGDU_CONTENT_ID_PROG_1,                 v_FragmentVersion,                 "Programme",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_startTime,                 v_endTime             )); </pre>		<pre>                 PX_SGDU_SERVICE_ID,                 v_FragmentVersion,                 "PayTvChannel",                 c_basicTv_ServiceType             ));              var ContentType v_Content := » valueof(m_def_Content(                 PX_SGDU_CONTENT_ID_PROG_1,                 v_FragmentVersion,                 "Programme",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_NTP_StartTime,                 v_NTP_EndTime             )); </pre>
<pre> » // change 01 (WK 6): global time definition                 v_endTime » // change 01 (WK 6): global time definition             )); </pre>	<>	<pre>                 v_NTP_StartTime,                 v_NTP_EndTime             )); </pre>
<pre>             var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             ));              var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(                 PX_SGDU_PURCHASE_ITEM_ID,                 v_FragmentVersion,                 PX_SGDU_PURCHASE_DATA_ID,                 PX_SGDU_SERVICE_ID,                 "PurchaseItem1"             ));              v_PurchaseItem.Descriptions := {{lang := </pre>	=	<pre>             var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             ));              var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(                 PX_SGDU_PURCHASE_ITEM_ID,                 v_FragmentVersion,                 PX_SGDU_PURCHASE_DATA_ID,                 PX_SGDU_SERVICE_ID,                 "PurchaseItem1"             ));              v_PurchaseItem.Descriptions := {{lang := </pre>

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

```

» omit, text_ := "IPsec" });

                                var PurchaseChannelType v_PurchaseChannel :=
» valueof(m_def_PurchaseChannel(
                                PX_SGDU_PURCHASE_CHANNEL_ID,
                                v_FragmentVersion,
                                "PurchaseChannel",
                                PX_PURCHASE_URL
                                ));

                                var PurchaseDataType v_PurchaseData :=
» valueof(m_def_PurchaseData(
                                PX_SGDU_PURCHASE_DATA_ID,
                                v_FragmentVersion,
                                {{omit, "Discount"}}},
                                PX_SGDU_PURCHASE_ITEM_ID,
                                PX_SGDU_PURCHASE_CHANNEL_ID,
                                v_PriceInfo
                                ));

                                f_addServiceFragment(v_SGDU, v_Service);
                                f_addContentFragment(v_SGDU, v_Content);
                                f_addScheduleFragment(v_SGDU, v_Schedule);
                                f_addAccessFragment(v_SGDU, v_Access);
                                f_addPurchaseItemFragment(v_SGDU,
» v_PurchaseItem);
                                f_addPurchaseChannelFragment(v_SGDU,
» v_PurchaseChannel);
                                f_addPurchaseDataFragment(v_SGDU,
» v_PurchaseData);
                                f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

                                f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

                                f_main_ut_RunBCastApplication();
                                if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
                                }
                                f_main_ut_CheckService("PayTvChannel");
                                f_main_ut_PurchaseService("PayTvChannel");

» f_main_ctrl_awaitingSubscription("PayTvChannel"); // TODO: This function
» currently does not include any checking and handling of authentication -
» see open issues

```

```

» omit, text_ := "IPsec" });

                                var PurchaseChannelType v_PurchaseChannel :=
» valueof(m_def_PurchaseChannel(
                                PX_SGDU_PURCHASE_CHANNEL_ID,
                                v_FragmentVersion,
                                "PurchaseChannel",
                                PX_PURCHASE_URL
                                ));

                                var PurchaseDataType v_PurchaseData :=
» valueof(m_def_PurchaseData(
                                PX_SGDU_PURCHASE_DATA_ID,
                                v_FragmentVersion,
                                {{omit, "Discount"}}},
                                PX_SGDU_PURCHASE_ITEM_ID,
                                PX_SGDU_PURCHASE_CHANNEL_ID,
                                v_PriceInfo
                                ));

                                f_addServiceFragment(v_SGDU, v_Service);
                                f_addContentFragment(v_SGDU, v_Content);
                                f_addScheduleFragment(v_SGDU, v_Schedule);
                                f_addAccessFragment(v_SGDU, v_Access);
                                f_addPurchaseItemFragment(v_SGDU,
» v_PurchaseItem);
                                f_addPurchaseChannelFragment(v_SGDU,
» v_PurchaseChannel);
                                f_addPurchaseDataFragment(v_SGDU,
» v_PurchaseData);
                                f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

                                f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

                                f_main_ut_RunBCastApplication();
                                if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
                                }
                                f_main_ut_CheckService("PayTvChannel");
                                f_main_ut_PurchaseService("PayTvChannel");

» f_main_ctrl_awaitingSubscription("PayTvChannel"); // TODO: This function
» currently does not include any checking and handling of authentication -
» see open issues

```

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre>                 // TODO: Omitted here is generation of proper » acceptance of terminal purchase request - see open issues                  f_main_ut_SelectService("PayTvChannel");                 f_main_ut_UseService("PayTvChannel");                 f_main_ut_CheckVideo("Programme");                  // postamble                 // TODO: Omitted here is stopping the » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue                 f_main_ut_PowerOff();                  f_cf_Broadcast_down();                 f_cf_Interaction_down();                 f_cf_UpperTester_down();              }// end test case TC_BCAST_INTER_CONF_102 </pre>	<>	<pre>                 // TODO: Omitted here is generation of proper » acceptance of terminal purchase request - see open issues                  f_main_ut_SelectService("PayTvChannel");                 f_main_ut_UseService("PayTvChannel");                 f_main_ut_CheckVideo("Programme");                  // postamble                 // TODO: Omitted here is stopping the » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue                 f_main_ut_PowerOff();                  f_cf_Broadcast_down();                 f_cf_Interaction_down();                 f_cf_UpperTester_down();              }// end test case TC_BCAST_CONTPROT_CONF_101 </pre>
<pre>             /**              *              * @desc              * &lt;p&gt;Service protection with SRTP via broadcast » channel (optional)&lt;/p&gt;              * &lt;p&gt;Test Case Description&lt;/p&gt;              * &lt;p&gt;with a terminal having received a service » guide containing a service with purchase information on the Service Guide » Delivery Channel and listening for an SRTP encrypted data stream on the » broadcast channel&lt;/p&gt;              * &lt;p&gt;when the terminal subscribes to that service » on the interactive channel Service Request/Response message exchange&lt;/p&gt;              * &lt;p&gt;then the terminal is able to receive, » decrypt, and display the data stream correctly&lt;/p&gt;              * &lt;p&gt;Specification Reference&lt;/p&gt;              * &lt;p&gt;[BCAST10-ServContProt] Section 6.8.1, » 9.2.&lt;/p&gt; </pre>	=	<pre>             /**              *              * @desc              * &lt;p&gt;Service protection with SRTP via broadcast » channel (optional)&lt;/p&gt;              * &lt;p&gt;Test Case Description&lt;/p&gt;              * &lt;p&gt;with a terminal having received a service » guide containing a service with purchase information on the Service Guide » Delivery Channel and listening for an SRTP encrypted data stream on the » broadcast channel&lt;/p&gt;              * &lt;p&gt;when the terminal subscribes to that service » on the interactive channel Service Request/Response message exchange&lt;/p&gt;              * &lt;p&gt;then the terminal is able to receive, » decrypt, and display the data stream correctly&lt;/p&gt;              * &lt;p&gt;Specification Reference&lt;/p&gt;              * &lt;p&gt;[BCAST10-ServContProt] Section 6.8.1, » 9.2.&lt;/p&gt; </pre>
<pre>             * &lt;p&gt;Preconditions&lt;/p&gt;             * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;             * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;             * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;             * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix:             * &lt;ul&gt;             * &lt;li&gt;Service fragment with » Name="PayTvChannel "&lt;/li&gt; </pre>	=	<pre>             * &lt;p&gt;Preconditions&lt;/p&gt;             * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;             * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;             * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;             * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix:             * &lt;ul&gt;             * &lt;li&gt;Service fragment with » Name="PayTvChannel "&lt;/li&gt; </pre>

## Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

```

*      <li>Content fragment with
» Name="Programme", and StartTime and EndTime elements indicating values
» within the time of test</li>
*      <li>Schedule fragment for content
» fragment with same values for startTime and endTime in the
» presentationWindow element</li>
*      <li>Access fragment for schedule
» fragment with KeyManagementSystem, EncryptionType set to SRTP, and a
» reference to the Session Description fragment</li>
*      <li>Session Description fragment
» containing SDP for "Programme" and the service protection information for
» SRTP</li>
*      <li>PurchaseItem fragment for service
» fragment with Description "SRTP" element</li>
*      <li>PurchaseData fragment for
» purchaseItem fragment with PriceInfo and Description "Discount"
» elements</li>
*      <li>PurchaseChannel fragment for
» purchaseItem fragment with PurchaseURL pointing to a subscription site</li>
*      </ul>
*      </p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*      <p>Test Procedure</p>
*      <p>
*      <ul>
*      <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
*      <li>Setup the test tool to stream via
» SRTP encrypted audio and video for "Programme" as well as key stream on
» broadcast channel</li>
*      <li>Setup the SEK delivery to the
» terminal either via DRM 2.0 ROAP (DRM) or authentication and push delivery
» (Smartcard)</li>
*      <li>Activate the BCAST application of
» the terminal</li>
*      <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*      <li>Browse the SG in the terminal</li>
*      <li>Setup the test tool to authenticate
» and accept subscription request for the Service "PayTvChannel" from the
» terminal via interactive channel</li>
*      <li>Purchase "TvChannel" on
» terminal</li>
*      <li>Access "TvChannel" on terminal</li>
*      </ul>

```

```

*      <li>Content fragment with
» Name="Programme", and StartTime and EndTime elements indicating values
» within the time of test</li>
*      <li>Schedule fragment for content
» fragment with same values for startTime and endTime in the
» presentationWindow element</li>
*      <li>Access fragment for schedule
» fragment with KeyManagementSystem, EncryptionType set to SRTP, and a
» reference to the Session Description fragment</li>
*      <li>Session Description fragment
» containing SDP for "Programme" and the service protection information for
» SRTP</li>
*      <li>PurchaseItem fragment for service
» fragment with Description "SRTP" element</li>
*      <li>PurchaseData fragment for
» purchaseItem fragment with PriceInfo and Description "Discount"
» elements</li>
*      <li>PurchaseChannel fragment for
» purchaseItem fragment with PurchaseURL pointing to a subscription site</li>
*      </ul>
*      </p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*      <p>Test Procedure</p>
*      <p>
*      <ul>
*      <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
*      <li>Setup the test tool to stream via
» SRTP encrypted audio and video for "Programme" as well as key stream on
» broadcast channel</li>
*      <li>Setup the SEK delivery to the
» terminal either via DRM 2.0 ROAP (DRM) or authentication and push delivery
» (Smartcard)</li>
*      <li>Activate the BCAST application of
» the terminal</li>
*      <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*      <li>Browse the SG in the terminal</li>
*      <li>Setup the test tool to authenticate
» and accept subscription request for the Service "PayTvChannel" from the
» terminal via interactive channel</li>
*      <li>Purchase "TvChannel" on
» terminal</li>
*      <li>Access "TvChannel" on terminal</li>
*      </ul>

```



Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

```

*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the delivery of the SG
*      <ul>
*          <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information.</li>
*          <li>"Programme" can be viewed by the end
» user.</li>
*      </ul>
*      </p>
*      <p>The following should be observable for the
» test tool
*      <ul>
*          <li>The test tool receives HTTP request
» from the end user to subscribe to service "TvChannel".</li>
*      </ul>
*      </p>
*      @verdict
*      pass in case the client pass the conformance
» test.
*      @verdict
*      fail in case the client does not pass the
» conformance test.
*      @verdict
*      inconc in case a guard timer expires.
*/
testcase TC_BCAST_CONTPROT_CONF_102() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Interaction_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();
    // TODO: Omitted here is starting of
» streaming server to broadcast the encrypted TV Channel program since
» handling of SEK delivery is open issue

    // test behavior
    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_FragmentVersion := 1;

```

f\_main\_ctrl\_InitStartTime(c\_one\_hour,c\_one

&lt;&gt;

```

*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the delivery of the SG
*      <ul>
*          <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information.</li>
*          <li>"Programme" can be viewed by the end
» user.</li>
*      </ul>
*      </p>
*      <p>The following should be observable for the
» test tool
*      <ul>
*          <li>The test tool receives HTTP request
» from the end user to subscribe to service "TvChannel".</li>
*      </ul>
*      </p>
*      @verdict
*      pass in case the client pass the conformance
» test.
*      @verdict
*      fail in case the client does not pass the
» conformance test.
*      @verdict
*      inconc in case a guard timer expires.
*/
testcase TC_BCAST_CONTPROT_CONF_102() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Interaction_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();
    // TODO: Omitted here is starting of
» streaming server to broadcast the encrypted TV Channel program since
» handling of SEK delivery is open issue

    // test behavior
    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_FragmentVersion := 1;

```

var UInt v\_NTP\_StartTime := 0;

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre> » _hour);                                // change 01 (WK 6): global time » definition </pre>		<pre> var UInt v_NTP_EndTime := 0; f_InitStartEndTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime);           // change 01 (WK 6): global » time definition var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);             // change 01 (WK 6): global » time definition </pre>	=	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime); var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_SERVICE_PROTECTION_S RTP )); var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID )); var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo(     1,     PX_MONETARY_PRICE,     PX_CURRENCY )); var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "PayTvChannel",     c_basicTv_ServiceType )); var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_FragmentVersion,     "Programme",     PX_SGDU_SERVICE_ID, </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_SERVICE_PROTECTION_S RTP )); var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID )); var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo(     1,     PX_MONETARY_PRICE,     PX_CURRENCY )); var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "PayTvChannel",     c_basicTv_ServiceType )); var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_FragmentVersion,     "Programme",     PX_SGDU_SERVICE_ID, </pre>

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre>                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_startTime, </pre>		<pre>                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_NTP_StartTime, </pre>
<pre> » // change 01 (WK 6): global time definition                 v_endTime » // change 01 (WK 6): global time definition </pre>	<>	<pre>                 v_NTP_EndTime </pre>
<pre>             ));              // TODO: Omitted here is setting of KMS value » - see open issues             var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             ));              var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(                 PX_SGDU_PURCHASE_ITEM_ID,                 v_FragmentVersion,                 PX_SGDU_PURCHASE_DATA_ID,                 PX_SGDU_SERVICE_ID,                 "PurchaseItem1"             ));             v_PurchaseItem.Descriptions := {{lang := » omit, text_:= "SRTP"}};              var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(                 PX_SGDU_PURCHASE_CHANNEL_ID,                 v_FragmentVersion,                 "PurchaseChannel",                 PX_PURCHASE_URL             )); </pre>	=	<pre>             ));              // TODO: Omitted here is setting of KMS value » - see open issues             var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             ));              var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(                 PX_SGDU_PURCHASE_ITEM_ID,                 v_FragmentVersion,                 PX_SGDU_PURCHASE_DATA_ID,                 PX_SGDU_SERVICE_ID,                 "PurchaseItem1"             ));             v_PurchaseItem.Descriptions := {{lang := » omit, text_:= "SRTP"}};              var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(                 PX_SGDU_PURCHASE_CHANNEL_ID,                 v_FragmentVersion,                 "PurchaseChannel",                 PX_PURCHASE_URL             )); </pre>

## Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

```

var PurchaseDataType v_PurchaseData :=
» valueof(m_def_PurchaseData(
    PX_SGDU_PURCHASE_DATA_ID,
    v_FragmentVersion,
    {{omit,"Discount"}}},
    PX_SGDU_PURCHASE_ITEM_ID,
    PX_SGDU_PURCHASE_CHANNEL_ID,
    v_PriceInfo
));

f_addServiceFragment(v_SGDU, v_Service);
f_addContentFragment(v_SGDU, v_Content);
f_addScheduleFragment(v_SGDU, v_Schedule);
f_addAccessFragment(v_SGDU, v_Access);
f_addPurchaseItemFragment(v_SGDU,
» v_PurchaseItem);
f_addPurchaseChannelFragment(v_SGDU,
» v_PurchaseChannel);
f_addPurchaseDataFragment(v_SGDU,
» v_PurchaseData);
f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

f_main_ut_RunBCastApplication();
if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("PayTvChannel");
    f_main_ut_PurchaseService("PayTvChannel");

» f_main_ctrl_awaitingSubscription("PayTvChannel"); // TODO: This function
» currently does not include any checking and handling of authentication -
» see open issues

    // TODO: Omitted here is generation of proper
» acceptance of terminal purchase request - see open issues

    f_main_ut_SelectService("PayTvChannel");
    f_main_ut_UseService("PayTvChannel");
    f_main_ut_CheckVideo("Programme");

    // postamble
    // TODO: Omitted here is stopping the
» streaming server to broadcast the encrypted TV Channel program since

```

```

var PurchaseDataType v_PurchaseData :=
» valueof(m_def_PurchaseData(
    PX_SGDU_PURCHASE_DATA_ID,
    v_FragmentVersion,
    {{omit,"Discount"}}},
    PX_SGDU_PURCHASE_ITEM_ID,
    PX_SGDU_PURCHASE_CHANNEL_ID,
    v_PriceInfo
));

f_addServiceFragment(v_SGDU, v_Service);
f_addContentFragment(v_SGDU, v_Content);
f_addScheduleFragment(v_SGDU, v_Schedule);
f_addAccessFragment(v_SGDU, v_Access);
f_addPurchaseItemFragment(v_SGDU,
» v_PurchaseItem);
f_addPurchaseChannelFragment(v_SGDU,
» v_PurchaseChannel);
f_addPurchaseDataFragment(v_SGDU,
» v_PurchaseData);
f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

f_main_ut_RunBCastApplication();
if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("PayTvChannel");
    f_main_ut_PurchaseService("PayTvChannel");

» f_main_ctrl_awaitingSubscription("PayTvChannel"); // TODO: This function
» currently does not include any checking and handling of authentication -
» see open issues

    // TODO: Omitted here is generation of proper
» acceptance of terminal purchase request - see open issues

    f_main_ut_SelectService("PayTvChannel");
    f_main_ut_UseService("PayTvChannel");
    f_main_ut_CheckVideo("Programme");

    // postamble
    // TODO: Omitted here is stopping the
» streaming server to broadcast the encrypted TV Channel program since

```

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre> » handling of SEK delivery is open issue     f_main_ut_PowerOff();      f_cf_Broadcast_down();     f_cf_Interaction_down();     f_cf_UpperTester_down(); </pre>		<pre> » handling of SEK delivery is open issue     f_main_ut_PowerOff();      f_cf_Broadcast_down();     f_cf_Interaction_down();     f_cf_UpperTester_down(); </pre>
<pre>     }// end test case TC_BCAST_INTER_CONF_102 </pre>	<>	<pre>     }// end test case TC_BCAST_CONTPROT_CONF_102 </pre>
<pre>     /**      *      * @desc      * &lt;p&gt;Service protection with ISMACrypt via » broadcast channel (optional)&lt;/p&gt;      * &lt;p&gt;Test Case Description&lt;/p&gt;      * &lt;p&gt;with a terminal having received a service » guide containing a service with purchase information on the Service Guide » Delivery Channel and listening for an ISMACrypt encrypted data stream on » the broadcast channel&lt;/p&gt;      * &lt;p&gt;when the terminal subscribes to that service » on the interactive channel Service Request/Response message exchange&lt;/p&gt;      * &lt;p&gt;then the terminal is able to receive, » decrypt, and display the data stream correctly&lt;/p&gt;      * &lt;p&gt;Specification Reference&lt;/p&gt; </pre>	=	<pre>     /**      *      * @desc      * &lt;p&gt;Service protection with ISMACrypt via » broadcast channel (optional)&lt;/p&gt;      * &lt;p&gt;Test Case Description&lt;/p&gt;      * &lt;p&gt;with a terminal having received a service » guide containing a service with purchase information on the Service Guide » Delivery Channel and listening for an ISMACrypt encrypted data stream on » the broadcast channel&lt;/p&gt;      * &lt;p&gt;when the terminal subscribes to that service » on the interactive channel Service Request/Response message exchange&lt;/p&gt;      * &lt;p&gt;then the terminal is able to receive, » decrypt, and display the data stream correctly&lt;/p&gt;      * &lt;p&gt;Specification Reference&lt;/p&gt; </pre>
<pre>      * &lt;p&gt;[BCAST10-ServContProt] Section 6.8.1, » 9.3.&lt;/p&gt; </pre>	<>	<pre>      * &lt;p&gt;[BCAST10-ServContProt] Section 6.8.1, » 9.3.&lt;/p&gt; </pre>
<pre>      * &lt;p&gt;Preconditions&lt;/p&gt;      * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;      * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;      * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;      * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix:      * &lt;ul&gt;      * &lt;li&gt;Service fragment with » Name="PayTvChannel "&lt;/li&gt;      * &lt;li&gt;Content fragment with » Name="Programme", and StartTime and EndTime elements indicating values » within the time of test&lt;/li&gt;      * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt;      * &lt;li&gt;Access fragment for schedule » fragment with KeyManagmentSystem, EncryptionType set to ISMACrypt, and a » reference to the Session Description fragment&lt;/li&gt;      * &lt;li&gt;Session Description fragment </pre>	=	<pre>      * &lt;p&gt;Preconditions&lt;/p&gt;      * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;      * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;      * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;      * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix:      * &lt;ul&gt;      * &lt;li&gt;Service fragment with » Name="PayTvChannel "&lt;/li&gt;      * &lt;li&gt;Content fragment with » Name="Programme", and StartTime and EndTime elements indicating values » within the time of test&lt;/li&gt;      * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt;      * &lt;li&gt;Access fragment for schedule » fragment with KeyManagmentSystem, EncryptionType set to ISMACrypt, and a » reference to the Session Description fragment&lt;/li&gt;      * &lt;li&gt;Session Description fragment </pre>

## Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

```

» containing SDP for "Programme" and the service protection information for
» ISMACrypt</li>
*          <li>PurchaseItem fragment for service
» fragment with Description "ISMACrypt" element</li>
*          <li>PurchaseData fragment for
» purchaseItem fragment with PriceInfo and Description "Discount"
» elements</li>
*          <li>PurchaseChannel fragment for
» purchaseItem fragment with PurchaseURL pointing to a subscription site</li>
*          </ul>
*          </p>
*          <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*          <p>Test Procedure</p>
*          <p>
*          <ul>
*          <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
*          <li>Setup the test tool to stream via
» ISMACrypt encrypted audio and video for "Programme" as well as key stream
» on broadcast channel</li>
*          <li>Setup the SEK delivery to the
» terminal either via DRM 2.0 ROAP (DRM) or authentication and push delivery
» (Smartcard)</li>
*          <li>Activate the BCAST application of
» the terminal</li>
*          <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*          <li>Browse the SG in the terminal</li>
*          <li>Setup the test tool to authenticate
» and accept subscription request for the Service "PayTvChannel" from the
» terminal via interactive channel</li>
*          <li>Purchase "TvChannel" on
» terminal</li>
*          <li>Access "TvChannel" on terminal</li>
*          </ul>
*          </p>
*          <p>Pass-Criteria</p>
*          <p>The following should be visible to the end
» user after the delivery of the SG
*          <ul>
*          <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information.</li>
*          <li>"Programme" can be viewed by the end
» user.</li>

```

```

» containing SDP for "Programme" and the service protection information for
» ISMACrypt</li>
*          <li>PurchaseItem fragment for service
» fragment with Description "ISMACrypt" element</li>
*          <li>PurchaseData fragment for
» purchaseItem fragment with PriceInfo and Description "Discount"
» elements</li>
*          <li>PurchaseChannel fragment for
» purchaseItem fragment with PurchaseURL pointing to a subscription site</li>
*          </ul>
*          </p>
*          <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*          <p>Test Procedure</p>
*          <p>
*          <ul>
*          <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
*          <li>Setup the test tool to stream via
» ISMACrypt encrypted audio and video for "Programme" as well as key stream
» on broadcast channel</li>
*          <li>Setup the SEK delivery to the
» terminal either via DRM 2.0 ROAP (DRM) or authentication and push delivery
» (Smartcard)</li>
*          <li>Activate the BCAST application of
» the terminal</li>
*          <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*          <li>Browse the SG in the terminal</li>
*          <li>Setup the test tool to authenticate
» and accept subscription request for the Service "PayTvChannel" from the
» terminal via interactive channel</li>
*          <li>Purchase "TvChannel" on
» terminal</li>
*          <li>Access "TvChannel" on terminal</li>
*          </ul>
*          </p>
*          <p>Pass-Criteria</p>
*          <p>The following should be visible to the end
» user after the delivery of the SG
*          <ul>
*          <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information.</li>
*          <li>"Programme" can be viewed by the end
» user.</li>

```

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre> *      &lt;/ul&gt; *      &lt;/p&gt; *      &lt;p&gt;The following should be observable for the » test tool *      &lt;ul&gt; *          &lt;li&gt;The test tool receives HTTP request » from the end user to subscribe to service "TvChannel".&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; * @verdict *      pass in case the client pass the conformance » test. * @verdict *      fail in case the client does not pass the » conformance test. * @verdict *      inconc in case a guard timer expires. */ testcase TC_BCAST_CONTPROT_CONF_103() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up();     f_cf_Interaction_up();     f_cf_UpperTester_up();      // preamble     f_startCommonUtPreamble();     // TODO: Omitted here is starting of » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>		<pre> *      &lt;/ul&gt; *      &lt;/p&gt; *      &lt;p&gt;The following should be observable for the » test tool *      &lt;ul&gt; *          &lt;li&gt;The test tool receives HTTP request » from the end user to subscribe to service "TvChannel".&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; * @verdict *      pass in case the client pass the conformance » test. * @verdict *      fail in case the client does not pass the » conformance test. * @verdict *      inconc in case a guard timer expires. */ testcase TC_BCAST_CONTPROT_CONF_103() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up();     f_cf_Interaction_up();     f_cf_UpperTester_up();      // preamble     f_startCommonUtPreamble();     // TODO: Omitted here is starting of » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>
<pre> » _hour);     f_main_ctrl_InitStartTime(c_one_hour,c_one     // change 01 (WK 6): global time definition </pre>	<>	<pre>     var UInt v_NTP_StartTime := 0;      var UInt v_NTP_EndTime := 0;     f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
	=	
<pre>     var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime);     // change 01 (WK 6): global time » definition     var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);     // change 01 (WK 6): global time </pre>	<>	<pre>     var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);      var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

» definition		
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_SERVICE_PROTECTION_ISMA ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp(  » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo(     1,     PX_MONETARY_PRICE,     PX_CURRENCY ));  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "PayTvChannel",     c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_FragmentVersion,     "Programme",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_FragmentVersion,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_startTime, </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_SERVICE_PROTECTION_ISMA ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp(  » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo(     1,     PX_MONETARY_PRICE,     PX_CURRENCY ));  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "PayTvChannel",     c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_FragmentVersion,     "Programme",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_FragmentVersion,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_NTP_StartTime, </pre>
<>	<>	<>



Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre> » // change 01 (WK 6): global time definition     v_endTime » // change 01 (WK 6): global time definition     ));      // TODO: Omitted here is setting of KMS value » - see open issues     var AccessType v_Access := » valueof(m_def_Access(         PX_SGDU_ACCESS_ID_PROG_1,         v_FragmentVersion,         v_AccessType,         PX_SGDU_SERVICE_ID,         PX_SGDU_SCHEDULE_ID_PROG_1,         c_class_SG     ));      var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(         PX_SGDU_PURCHASE_ITEM_ID,         v_FragmentVersion,         PX_SGDU_PURCHASE_DATA_ID,         PX_SGDU_SERVICE_ID,         "PurchaseItem1"     ));     v_PurchaseItem.Descriptions := {{lang := » omit, text_:= "ISMACrypt"}};      var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(         PX_SGDU_PURCHASE_CHANNEL_ID,         v_FragmentVersion,         "PurchaseChannel",         PX_PURCHASE_URL     ));      var PurchaseDataType v_PurchaseData := » valueof(m_def_PurchaseData(         PX_SGDU_PURCHASE_DATA_ID,         v_FragmentVersion,         {{omit, "Discount"}}},         PX_SGDU_PURCHASE_ITEM_ID,         PX_SGDU_PURCHASE_CHANNEL_ID,         v_PriceInfo     ));      f_addServiceFragment(v_SGDU, v_Service); </pre>	<pre> v_NTP_EndTime </pre>
<pre>     ));      // TODO: Omitted here is setting of KMS value » - see open issues     var AccessType v_Access := » valueof(m_def_Access(         PX_SGDU_ACCESS_ID_PROG_1,         v_FragmentVersion,         v_AccessType,         PX_SGDU_SERVICE_ID,         PX_SGDU_SCHEDULE_ID_PROG_1,         c_class_SG     ));      var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(         PX_SGDU_PURCHASE_ITEM_ID,         v_FragmentVersion,         PX_SGDU_PURCHASE_DATA_ID,         PX_SGDU_SERVICE_ID,         "PurchaseItem1"     ));     v_PurchaseItem.Descriptions := {{lang := » omit, text_:= "ISMACrypt"}};      var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(         PX_SGDU_PURCHASE_CHANNEL_ID,         v_FragmentVersion,         "PurchaseChannel",         PX_PURCHASE_URL     ));      var PurchaseDataType v_PurchaseData := » valueof(m_def_PurchaseData(         PX_SGDU_PURCHASE_DATA_ID,         v_FragmentVersion,         {{omit, "Discount"}}},         PX_SGDU_PURCHASE_ITEM_ID,         PX_SGDU_PURCHASE_CHANNEL_ID,         v_PriceInfo     ));      f_addServiceFragment(v_SGDU, v_Service); </pre>	<pre>     ));      // TODO: Omitted here is setting of KMS value » - see open issues     var AccessType v_Access := » valueof(m_def_Access(         PX_SGDU_ACCESS_ID_PROG_1,         v_FragmentVersion,         v_AccessType,         PX_SGDU_SERVICE_ID,         PX_SGDU_SCHEDULE_ID_PROG_1,         c_class_SG     ));      var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(         PX_SGDU_PURCHASE_ITEM_ID,         v_FragmentVersion,         PX_SGDU_PURCHASE_DATA_ID,         PX_SGDU_SERVICE_ID,         "PurchaseItem1"     ));     v_PurchaseItem.Descriptions := {{lang := » omit, text_:= "ISMACrypt"}};      var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(         PX_SGDU_PURCHASE_CHANNEL_ID,         v_FragmentVersion,         "PurchaseChannel",         PX_PURCHASE_URL     ));      var PurchaseDataType v_PurchaseData := » valueof(m_def_PurchaseData(         PX_SGDU_PURCHASE_DATA_ID,         v_FragmentVersion,         {{omit, "Discount"}}},         PX_SGDU_PURCHASE_ITEM_ID,         PX_SGDU_PURCHASE_CHANNEL_ID,         v_PriceInfo     ));      f_addServiceFragment(v_SGDU, v_Service); </pre>

Datei: AtsBCast\_ContentProtectionTests.ttcn (Fortsetzung)

<pre>                 f_addContentFragment(v_SGDU, v_Content);                 f_addScheduleFragment(v_SGDU, v_Schedule);                 f_addAccessFragment(v_SGDU, v_Access);                 f_addPurchaseItemFragment(v_SGDU, » v_PurchaseItem);                  f_addPurchaseChannelFragment(v_SGDU, » v_PurchaseChannel);                  f_addPurchaseDataFragment(v_SGDU, » v_PurchaseData);                  f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);                  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);                  f_main_ut_RunBCastApplication();                 if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);                 }                 f_main_ut_CheckService("PayTvChannel");                 f_main_ut_PurchaseService("PayTvChannel");  » f_main_ctrl_awaitingSubscription("PayTvChannel"); // TODO: This function » currently does not include any checking and handling of authentication - » see open issues                  // TODO: Omitted here is generation of proper » acceptance of terminal purchase request - see open issues                  f_main_ut_SelectService("PayTvChannel");                 f_main_ut_UseService("PayTvChannel");                 f_main_ut_CheckVideo("Programme");                  // postamble                 // TODO: Omitted here is stopping the » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue                 f_main_ut_PowerOff();                  f_cf_Broadcast_down();                 f_cf_Interaction_down();                 f_cf_UpperTester_down();                  } // end test case TC_BCAST_INTER_CONF_103             } // end group clientConformanceTestCases         } // end group bcastConformanceTestCases     } </pre>	<>	<pre>                 f_addContentFragment(v_SGDU, v_Content);                 f_addScheduleFragment(v_SGDU, v_Schedule);                 f_addAccessFragment(v_SGDU, v_Access);                 f_addPurchaseItemFragment(v_SGDU, » v_PurchaseItem);                  f_addPurchaseChannelFragment(v_SGDU, » v_PurchaseChannel);                  f_addPurchaseDataFragment(v_SGDU, » v_PurchaseData);                  f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);                  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);                  f_main_ut_RunBCastApplication();                 if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);                 }                 f_main_ut_CheckService("PayTvChannel");                 f_main_ut_PurchaseService("PayTvChannel");  » f_main_ctrl_awaitingSubscription("PayTvChannel"); // TODO: This function » currently does not include any checking and handling of authentication - » see open issues                  // TODO: Omitted here is generation of proper » acceptance of terminal purchase request - see open issues                  f_main_ut_SelectService("PayTvChannel");                 f_main_ut_UseService("PayTvChannel");                 f_main_ut_CheckVideo("Programme");                  // postamble                 // TODO: Omitted here is stopping the » streaming server to broadcast the encrypted TV Channel program since » handling of SEK delivery is open issue                 f_main_ut_PowerOff();                  f_cf_Broadcast_down();                 f_cf_Interaction_down();                 f_cf_UpperTester_down();                  } // end test case TC_BCAST_CONTPROT_CONF_103             } // end group clientConformanceTestCases         } // end group bcastConformanceTestCases     } </pre>
--	----	---

Datei: AtsBCast\_FileAndStreamDistributionTests.ttcn

```

/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies pilot tests for BCast file and stream
» distribution.
 */
module AtsBCast_FileAndStreamDistributionTests {
    import from LibBCast_Interface all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibCommon_BasicTypesAndValues all;
    import from AtsBCast_ModuleParameters all;
    import from LibBCast_ModuleParameters all;
    import from AtsBCast_TestConfiguration_Functions all;
    import from AtsBCast_TestSystem all;

    group bcastConformanceTestCases {
        group clientConformanceTestCases {
            /**
             *
             * @desc
             *   <p>Support of in-band delivery of meta-data and
» FLUTE (optional)</p>
             *   <p>Test Case Description</p>
             *   <p>with a terminal having received Service
» Guide information related to a file ready for download via broadcast
» channel using FLUTE.</p>
             *   <p>when the terminal downloads file via
» broadcast channel</p>
             *   <p>then the terminal receives file and is able
» to display it </p>
             *   <p>Specification Reference</p>
             *   <p>[BCAST10-Distribution] Section 5.2.6,
» 5.1.2.5.3.1</p>
             *   <p>Preconditions</p>
             *   <p>Set up the test tool for Service Guide
» delivery to provide file delivery information for "File2". "File2" is
» available on the broadcast channel.</p>
             *   <p>This test cases uses the following SG
» fragment instantiations found in the Appendix:
             *
             <ul>

```

```

= /**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies pilot tests for BCast file and stream
» distribution.
 */
module AtsBCast_FileAndStreamDistributionTests {
    import from LibBCast_Interface all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibCommon_BasicTypesAndValues all;
    import from AtsBCast_ModuleParameters all;
    import from LibBCast_ModuleParameters all;
    import from AtsBCast_TestConfiguration_Functions all;
    import from AtsBCast_TestSystem all;

    group bcastConformanceTestCases {
        group clientConformanceTestCases {
            /**
             *
             * @desc
             *   <p>Support of in-band delivery of meta-data and
» FLUTE (optional)</p>
             *   <p>Test Case Description</p>
             *   <p>with a terminal having received Service
» Guide information related to a file ready for download via broadcast
» channel using FLUTE.</p>
             *   <p>when the terminal downloads file via
» broadcast channel</p>
             *   <p>then the terminal receives file and is able
» to display it </p>
             *   <p>Specification Reference</p>
             *   <p>[BCAST10-Distribution] Section 5.2.6,
» 5.1.2.5.3.1</p>
             *   <p>Preconditions</p>
             *   <p>Set up the test tool for Service Guide
» delivery to provide file delivery information for "File2". "File2" is
» available on the broadcast channel.</p>
             *   <p>This test cases uses the following SG
» fragment instantiations found in the Appendix:
             *
             <ul>

```

Datei: AtsBCast\_FileAndStreamDistributionTests.ttcn  
(Fortsetzung)

```

*          <li>Service fragment with
» Name="FILE"</li>
*          <li>Content fragment with
» Name="File2"</li>
*          <li>Schedule fragment with
» contentLocation set to URI for "name-of-the-file1".</li>
*          <li>Access fragment with
» BroadcastServiceDelivery element containing </li>
*          <li>SessionDescription for a FLUTE
» session indicating FEC encoding 0 (Compact No-Code FEC) and not containing
» FileDescription.</li>
*          <li>Transport Object with TOI="1".</li>
*          </ul>
*        </p>
*        <p>Set the FLUTE session to contain an
» FDT-Instance with one File-element having attributes
» Content-Location="name-of-the-file1" and TOI="1"</p>
*        <p>Note:
*          <ul>
*            <li>All the fragments are associated
» with the same Service fragment.</li>
*            <li>IPv4 is used</li>
*            <li>File is GZIP encoded</li>
*          </ul>
*        </p>
*        <p>Test Procedure</p>
*        <p>
*          <ul>
*            <li>Setup the test tool to distribute
» file "File2" via the broadcast channel</li>
*            <li>Set up the terminal to receive BCAST
» service guide announcements</li>
*            <li>Browse the SG in the terminal and
» select the "File2" to download</li>
*            <li>Wait for the file download</li>
*          </ul>
*        </p>
*        <p>Note: "File2" can be a jpg picture</p>
*        <p>Pass-Criteria</p>
*        <p>The following should be visible to the end
» user
*          <ul>
*            <li>There is a service "FILE" associated
» with file "File2"</li>
*            <li>The file is successfully downloaded
» to the terminal.</li>

```

```

*          <li>Service fragment with
» Name="FILE"</li>
*          <li>Content fragment with
» Name="File2"</li>
*          <li>Schedule fragment with
» contentLocation set to URI for "name-of-the-file1".</li>
*          <li>Access fragment with
» BroadcastServiceDelivery element containing </li>
*          <li>SessionDescription for a FLUTE
» session indicating FEC encoding 0 (Compact No-Code FEC) and not containing
» FileDescription.</li>
*          <li>Transport Object with TOI="1".</li>
*          </ul>
*        </p>
*        <p>Set the FLUTE session to contain an
» FDT-Instance with one File-element having attributes
» Content-Location="name-of-the-file1" and TOI="1"</p>
*        <p>Note:
*          <ul>
*            <li>All the fragments are associated
» with the same Service fragment.</li>
*            <li>IPv4 is used</li>
*            <li>File is GZIP encoded</li>
*          </ul>
*        </p>
*        <p>Test Procedure</p>
*        <p>
*          <ul>
*            <li>Setup the test tool to distribute
» file "File2" via the broadcast channel</li>
*            <li>Set up the terminal to receive BCAST
» service guide announcements</li>
*            <li>Browse the SG in the terminal and
» select the "File2" to download</li>
*            <li>Wait for the file download</li>
*          </ul>
*        </p>
*        <p>Note: "File2" can be a jpg picture</p>
*        <p>Pass-Criteria</p>
*        <p>The following should be visible to the end
» user
*          <ul>
*            <li>There is a service "FILE" associated
» with file "File2"</li>
*            <li>The file is successfully downloaded
» to the terminal.</li>

```

Datei: AtsBCast\_FileAndStreamDistributionTests.ttcn  
(Fortsetzung)

<pre> *      &lt;/ul&gt; *      &lt;/p&gt; *      &lt;p&gt;Note: To verify the file was correctly » downloaded the picture should be correctly displayed.&lt;/p&gt; * @verdict *      pass in case the client pass the conformance » test.  * @verdict *      fail in case the client does not pass the » conformance test.  * @verdict *      inconc in case a guard timer expires. */ testcase TC_BCAST_DIST_CONF_102() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up();     f_cf_Data_up();     f_cf_UpperTester_up();      // preamble     var UInt v_TransferId := 1;     f_main_data_startFileTransfer(v_TransferId, » PX_SDP_FILE_DELIVERY, {{fileRef := PX_PICTURE_ID}});     f_startCommonUtPreamble();      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>		<pre> *      &lt;/ul&gt; *      &lt;/p&gt; *      &lt;p&gt;Note: To verify the file was correctly » downloaded the picture should be correctly displayed.&lt;/p&gt; * @verdict *      pass in case the client pass the conformance » test.  * @verdict *      fail in case the client does not pass the » conformance test.  * @verdict *      inconc in case a guard timer expires. */ testcase TC_BCAST_DIST_CONF_102() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up();     f_cf_Data_up();     f_cf_UpperTester_up();      // preamble     var UInt v_TransferId := 1;     f_main_data_startFileTransfer(v_TransferId, » PX_SDP_FILE_DELIVERY, {{fileRef := PX_PICTURE_ID}});     f_startCommonUtPreamble();      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>
<pre> » _hour);     f_main_ctrl_InitStartTime(c_one_hour,c_one     // change 01 (WK 6): global time definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime);    // change 01 (WK 6): global time » definition  var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);      // change 01 (WK 6): global time » definition </pre>	=	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(PX_SDP_VIDEO_PROG_1_REF_ID, </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(PX_SDP_VIDEO_PROG_1_REF_ID, </pre>

Datei: AtsBCast\_FileAndStreamDistributionTests.ttcn  
(Fortsetzung)

```

» PX_SDP_FILE_DELIVERY));
        var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(f_getAccessType(PX_BROADCAST_NETWORK_BEARER
» ), PX_SDP_VIDEO_PROG_1_REF_ID));

        var ServiceType v_Service :=
» valueof(m_def_Service(
                PX_SGDU_SERVICE_ID,
                v_FragmentVersion,
                "FILE",
                c_basicTv_ServiceType
        ));

        var ContentType v_Content :=
» valueof(m_def_Content(
                PX_SGDU_CONTENT_ID_PROG_1,
                v_FragmentVersion,
                "File2",
                PX_SGDU_SERVICE_ID,
                v_DateTime_StartTime,
                v_DateTime_EndTime
        ));

        var ScheduleType v_Schedule :=
» valueof(m_def_Schedule(
                PX_SGDU_SCHEDULE_ID_PROG_1,
                v_FragmentVersion,
                PX_SGDU_SERVICE_ID,
                PX_SGDU_CONTENT_ID_PROG_1,
                v_startTime,
» // change 01 (WK 6): global time definition
                v_endTime
» // change 01 (WK 6): global time definition
        ));

        var AccessType v_Access :=
» valueof(m_def_Access(
                PX_SGDU_ACCESS_ID_PROG_1,
                v_FragmentVersion,
                v_AccessType,
                PX_SGDU_SERVICE_ID,
                omit,
                c_class_SG
        ));

        f_addServiceFragment(v_SGDU, v_Service);

```

```

» PX_SDP_FILE_DELIVERY));
        var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(f_getAccessType(PX_BROADCAST_NETWORK_BEARER
» ), PX_SDP_VIDEO_PROG_1_REF_ID));

        var ServiceType v_Service :=
» valueof(m_def_Service(
                PX_SGDU_SERVICE_ID,
                v_FragmentVersion,
                "FILE",
                c_basicTv_ServiceType
        ));

        var ContentType v_Content :=
» valueof(m_def_Content(
                PX_SGDU_CONTENT_ID_PROG_1,
                v_FragmentVersion,
                "File2",
                PX_SGDU_SERVICE_ID,
                v_DateTime_StartTime,
                v_DateTime_EndTime
        ));

        var ScheduleType v_Schedule :=
» valueof(m_def_Schedule(
                PX_SGDU_SCHEDULE_ID_PROG_1,
                v_FragmentVersion,
                PX_SGDU_SERVICE_ID,
                PX_SGDU_CONTENT_ID_PROG_1,
                v_NTP_StartTime,
                v_NTP_EndTime
        ));

        var AccessType v_Access :=
» valueof(m_def_Access(
                PX_SGDU_ACCESS_ID_PROG_1,
                v_FragmentVersion,
                v_AccessType,
                PX_SGDU_SERVICE_ID,
                omit,
                c_class_SG
        ));

        f_addServiceFragment(v_SGDU, v_Service);

```

Datei: AtsBCast\_FileAndStreamDistributionTests.ttcn  
(Fortsetzung)

```

        f_addContentFragment(v_SGDU, v_Content);
        f_addScheduleFragment(v_SGDU, v_Schedule);
        f_addAccessFragment(v_SGDU, v_Access);
        f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
            }
            f_main_ut_CheckService("FILE");
            f_main_ut_SelectService("FILE");
            f_main_ut_CheckFile("File2");
            f_main_ut_SelectFile("File2");
            f_main_ut_CheckFileReceived("Picture");

            // postamble
            f_main_data_stopFileTransfer(v_TransferId);

            f_cf_Broadcast_down();
            f_cf_Data_down();
            f_cf_UpperTester_down();
            } // end test case TC_BCAST_DIST_CONF_102
        } // end group clientConformanceTestCases
    } // end group bcastConformanceTestCases
}

```

```

        f_addContentFragment(v_SGDU, v_Content);
        f_addScheduleFragment(v_SGDU, v_Schedule);
        f_addAccessFragment(v_SGDU, v_Access);
        f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
            }
            f_main_ut_CheckService("FILE");
            f_main_ut_SelectService("FILE");
            f_main_ut_CheckFile("File2");
            f_main_ut_SelectFile("File2");
            f_main_ut_CheckFileReceived("Picture");

            // postamble
            f_main_data_stopFileTransfer(v_TransferId);

            f_cf_Broadcast_down();
            f_cf_Data_down();
            f_cf_UpperTester_down();
            } // end test case TC_BCAST_DIST_CONF_102
        } // end group clientConformanceTestCases
    } // end group bcastConformanceTestCases
}

```

Datei: AtsBCast\_Main\_Functions.ttcn

```

/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies functions which create either BCAST control or
 *   Upper Tester components and start library BCAST or Upper Tester
» functions
 *   on them.
 */
module AtsBCast_Main_Functions {

```

```

=
/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies functions which create either BCAST control or
 *   Upper Tester components and start library BCAST or Upper Tester
» functions
 *   on them.
 */
module AtsBCast_Main_Functions {

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

<pre> import from LibBCast_Common_TypesAndValues all; import from LibBCast_ServiceGuide_TypesAndValues all; import from AtsBCast_ServiceGuide_Functions all; import from LibBCast_Common_Templates all; import from LibBCast_ServicePrimitives_TypesAndValues all; import from LibBCast_UpperTester_Functions all; import from AtsBCast_TestSystem all; import from AtsBCast_ModuleParameters all; import from LibBCast_ModuleParameters all; import from LibCommon_Time all; import from LibCommon_BasicTypesAndValues all; import from LibBCast_UpperTesterPrimitives_TypesAndValues all; import from LibBCast_BCastNetworkControl_Functions all; import from LibBCast_BCastNetworkData_Functions all; import from LibBCast_UpperTester_Functions all;  group bcastCtrlMainFunctions { </pre>		<pre> import from LibBCast_Common_TypesAndValues all; import from LibBCast_ServiceGuide_TypesAndValues all; import from AtsBCast_ServiceGuide_Functions all; import from LibBCast_Common_Templates all; import from LibBCast_ServicePrimitives_TypesAndValues all; import from LibBCast_UpperTester_Functions all; import from AtsBCast_TestSystem all; import from AtsBCast_ModuleParameters all; import from LibBCast_ModuleParameters all; import from LibCommon_Time all; import from LibCommon_BasicTypesAndValues all; import from LibBCast_UpperTesterPrimitives_TypesAndValues all; import from LibBCast_BCastNetworkControl_Functions all; import from LibBCast_BCastNetworkData_Functions all; import from LibBCast_UpperTester_Functions all;  group bcastCtrlMainFunctions { </pre>
<pre> // change 01 (WK 6): global time definition /**  *  * @desc  *   initialize the start and end time. The start time will » be  *   set to a value which is test execution plus the start » offset in seconds and the  *   endTime will be set to a value equal to the start time » plus the specified duration in secs.  * @param  *   p_startOffset Start Offset in seconds  * @param  *   p_duration Duration in seconds between the start and » the end  */ function f_main_ctrl_InitStartTime(integer p_startOffset, » integer p_duration) runs on BCastMainComponent {     v_startTime := f_getNTPTTime() + p_startOffset;     v_endTime := v_startTime + p_duration;     vc_CTRL.start » (f_ctrl_InitStartTime(v_startTime,v_endTime));     vc_CTRL.done; } </pre>	+-	
<pre> /**  *  * @desc  *   This function creates a test component and starts a </pre>	=	<pre> /**  *  * @desc  *   This function creates a test component and starts a </pre>



Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

» function that
    *   waits for the receiving of a http subscription
    *   indication.
    * @param
    *   p_serviceName The service name for the subscription
    * @verdict
    *   pass The substription was successfull received for the

» given
    *   service.
    * @verdict
    *   fail A message was received which is not expected.
    * @verdict
    *   inconc A guard timer expires.
    */

    function f_main_ctrl_awaitingSubscription(charstring
» p_serviceName) runs on BCastMainComponent {
        vc_CTRL.start
» (f_ctrl_awaitingHttpSubscriptionIndication(p_serviceName));
        vc_CTRL.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a

» function that
    *   waits for a mms indication.
    * @param
    *   content the content which should be inside of the mms.
    * @verdict
    *   pass The indication was successfull received for the

» given
    *   service.
    * @verdict
    *   fail A message was received which is not expected.
    * @verdict
    *   inconc A guard timer expires.
    */

    function f_main_ctrl_awaitingMMS(charstring content) runs on
» BCastMainComponent {

        vc_CTRL.start (f_ctrl_awaitingMMS(content));
        vc_CTRL.done;
    }

    /**
    *

```

```

» function that
    *   waits for the receiving of a http subscription
    *   indication.
    * @param
    *   p_serviceName The service name for the subscription
    * @verdict
    *   pass The substription was successfull received for the

» given
    *   service.
    * @verdict
    *   fail A message was received which is not expected.
    * @verdict
    *   inconc A guard timer expires.
    */

    function f_main_ctrl_awaitingSubscription(charstring
» p_serviceName) runs on BCastMainComponent {
        vc_CTRL.start
» (f_ctrl_awaitingHttpSubscriptionIndication(p_serviceName));
        vc_CTRL.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a

» function that
    *   waits for a mms indication.
    * @param
    *   content the content which should be inside of the mms.
    * @verdict
    *   pass The indication was successfull received for the

» given
    *   service.
    * @verdict
    *   fail A message was received which is not expected.
    * @verdict
    *   inconc A guard timer expires.
    */

    function f_main_ctrl_awaitingMMS(charstring content) runs on
» BCastMainComponent {

        vc_CTRL.start (f_ctrl_awaitingMMS(content));
        vc_CTRL.done;
    }

    /**
    *

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

    * @desc
    *   This function creates a test component and starts a
» function that
    *   waits for a sms indication.
    * @param
    *   content the content which should be inside of the mms.
    * @verdict
    *   pass The indication was successfull received for the
» given
    *   service.
    * @verdict
    *   fail A message was received which is not expected.
    * @verdict
    *   inconc A guard timer expires.
    */
    function f_main_ctrl_awaitingSMS(charstring content) runs on
» BCastMainComponent {
        vc_CTRL.start (f_ctrl_awaitingSMS(content));
        vc_CTRL.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   broadcasts a service guide over the flute protocol
    * @param
    *   p_SGDU The ServiceGuideDeliveryUnit which should be
    *   broadcasted
    * @param
    *   p_Encoding The encoding type for compression in case
» that the
    *   service guide should be compressed otherwise this
» parameter
    *   should be omitted.
    * @verdict
    *   pass in case the broadcast of the service guide was
    *   successful
    * @verdict
    *   fail in case that the broadcast was not successful
    * @verdict
    *   inconc in case of the timer runs out of time
    */
    function
» f_main_ctrl_broadcastServiceGuide(ServiceGuideDeliveryUnit p_SGDU, template
» charstring p_Encoding) runs on BCastMainComponent {

```

```

    * @desc
    *   This function creates a test component and starts a
» function that
    *   waits for a sms indication.
    * @param
    *   content the content which should be inside of the mms.
    * @verdict
    *   pass The indication was successfull received for the
» given
    *   service.
    * @verdict
    *   fail A message was received which is not expected.
    * @verdict
    *   inconc A guard timer expires.
    */
    function f_main_ctrl_awaitingSMS(charstring content) runs on
» BCastMainComponent {
        vc_CTRL.start (f_ctrl_awaitingSMS(content));
        vc_CTRL.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   broadcasts a service guide over the flute protocol
    * @param
    *   p_SGDU The ServiceGuideDeliveryUnit which should be
    *   broadcasted
    * @param
    *   p_Encoding The encoding type for compression in case
» that the
    *   service guide should be compressed otherwise this
» parameter
    *   should be omitted.
    * @verdict
    *   pass in case the broadcast of the service guide was
    *   successful
    * @verdict
    *   fail in case that the broadcast was not successful
    * @verdict
    *   inconc in case of the timer runs out of time
    */
    function
» f_main_ctrl_broadcastServiceGuide(ServiceGuideDeliveryUnit p_SGDU, template
» charstring p_Encoding) runs on BCastMainComponent {

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        vc_CTRL.start (
            f_ctrl_broadcastServiceGuide(
                p_SGDU,
                PX_SGDU_ID,
                PX_SGDD_ID,
                p_Encoding)
        );
        vc_CTRL.done;
    }

    /**
     *
     * @desc
     *   This functions expect a http post request in order to
» deliver
     *   the service guide over the interaction channel.
     * @param
     *   p_SGDU The Service Guide Delivery Unit to send
     * @verdict
     *   pass the service guide could successful delivered over
» the
     *   interaction channel
     * @verdict
     *   fail the service fuide could not successful
» deflivered.
     * @verdict
     *   incon the guard timer runs out of time.
     */
    function
» f_main_ctrl_ServiceGuideOnRequest(ServiceGuideDeliveryUnit p_SGDU) runs on
» BCastMainComponent {

        vc_UTC.start(f_ut_GetServiceGuide(PX_SGDD_ID));

        vc_CTRL.start (
            f_ctrl_ServiceGuideViaHttp(
                p_SGDU,
                PX_SGDU_ID,
                PX_SGDD_ID));

        vc_CTRL.done;

        vc_UTC.done;

    }

}

group bcastDataMainFunctions {

```

```

        vc_CTRL.start (
            f_ctrl_broadcastServiceGuide(
                p_SGDU,
                PX_SGDU_ID,
                PX_SGDD_ID,
                p_Encoding)
        );
        vc_CTRL.done;
    }

    /**
     *
     * @desc
     *   This functions expect a http post request in order to
» deliver
     *   the service guide over the interaction channel.
     * @param
     *   p_SGDU The Service Guide Delivery Unit to send
     * @verdict
     *   pass the service guide could successful delivered over
» the
     *   interaction channel
     * @verdict
     *   fail the service fuide could not successful
» deflivered.
     * @verdict
     *   incon the guard timer runs out of time.
     */
    function
» f_main_ctrl_ServiceGuideOnRequest(ServiceGuideDeliveryUnit p_SGDU) runs on
» BCastMainComponent {

        vc_UTC.start(f_ut_GetServiceGuide(PX_SGDD_ID));

        vc_CTRL.start (
            f_ctrl_ServiceGuideViaHttp(
                p_SGDU,
                PX_SGDU_ID,
                PX_SGDD_ID));

        vc_CTRL.done;

        vc_UTC.done;

    }

}

group bcastDataMainFunctions {

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

/**
 *
 * @desc This function trigger the file server to deliver
» files to the terminal.
 * @param p_TransferId the transfer id
 * @param p_Sdp the used sdp
 * @param p_FileList the list of files which should delivered
» to the terminal
 * @verdict
 * pass In case that the file transfer request was
» successful
 * @verdict
 * fail A wrong message was received.
 * @verdict
 * inconc A guard timer expires.
 */
function f_main_data_startFileTransfer(UInt p_TransferId,
» charstring p_Sdp, TransferFileList p_FileList) runs on BCastMainComponent {
    vc_DATA.start (f_data_startFileTransfer(p_TransferId,
» p_Sdp, p_FileList));
    vc_DATA.done;
}

/**
 *
 * @desc This function triggers the file server to stop a
» file transfer operation
 * @param p_TransferId the id of the transfer to be stopped
 * @verdict
 * pass in case that the stop streaming request was
» successful
 * @verdict
 * fail A wrong message was received.
 * @verdict
 * inconc A guard timer expires.
 */
function f_main_data_stopFileTransfer(UInt p_TransferId) runs
» on BCastMainComponent {
    vc_DATA.start
» (f_data_stopFileTransfer(p_TransferId));
    vc_DATA.done;
}

/**
 *
 * @desc This function triggers the streaming server to
» stream a file.

```

```

/**
 *
 * @desc This function trigger the file server to deliver
» files to the terminal.
 * @param p_TransferId the transfer id
 * @param p_Sdp the used sdp
 * @param p_FileList the list of files which should delivered
» to the terminal
 * @verdict
 * pass In case that the file transfer request was
» successful
 * @verdict
 * fail A wrong message was received.
 * @verdict
 * inconc A guard timer expires.
 */
function f_main_data_startFileTransfer(UInt p_TransferId,
» charstring p_Sdp, TransferFileList p_FileList) runs on BCastMainComponent {
    vc_DATA.start (f_data_startFileTransfer(p_TransferId,
» p_Sdp, p_FileList));
    vc_DATA.done;
}

/**
 *
 * @desc This function triggers the file server to stop a
» file transfer operation
 * @param p_TransferId the id of the transfer to be stopped
 * @verdict
 * pass in case that the stop streaming request was
» successful
 * @verdict
 * fail A wrong message was received.
 * @verdict
 * inconc A guard timer expires.
 */
function f_main_data_stopFileTransfer(UInt p_TransferId) runs
» on BCastMainComponent {
    vc_DATA.start
» (f_data_stopFileTransfer(p_TransferId));
    vc_DATA.done;
}

/**
 *
 * @desc This function triggers the streaming server to
» stream a file.

```

## Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @param p_StreamId The stream id
        * @param p_ContentList The list of files which should be
» streamed

        * @param p_SDP The used SDP
        * @verdict
        *   pass In case that the streaming request was successful
        * @verdict
        *   fail A wrong message was received.
        * @verdict
        *   inconc A guard timer expires.
        */
        function f_main_data_startStreamingFile(
            UInt p_StreamId,
            charstring p_FileName,
            charstring p_SDP) runs on BCastMainComponent {
» p_FileName, p_SDP));
            vc_DATA.start (f_data_startStreaming(p_StreamId,
                vc_DATA.done;
            }

            /**
            *
            * @desc This function trigger the streaming server to stop
» the streaming of a file
            * @param p_StreamId the id of the stream which should
» stopped

            * @verdict
            *   pass in case that the stop streaming request was
» successful

            * @verdict
            *   fail A wrong message was received.
            * @verdict
            *   inconc A guard timer expires.
            */
            function f_main_data_stopStreaming(UInt p_StreamId) runs on
» BCastMainComponent {
                vc_DATA.start (f_data_stopStreaming(p_StreamId));
                vc_DATA.done;
            }
        }

        group upperTesterMainFunctions {

            /**
            *
            * @desc
            *   This function creates a test component and starts a

```

```

        * @param p_StreamId The stream id
        * @param p_ContentList The list of files which should be
» streamed

        * @param p_SDP The used SDP
        * @verdict
        *   pass In case that the streaming request was successful
        * @verdict
        *   fail A wrong message was received.
        * @verdict
        *   inconc A guard timer expires.
        */
        function f_main_data_startStreamingFile(
            UInt p_StreamId,
            charstring p_FileName,
            charstring p_SDP) runs on BCastMainComponent {
» p_FileName, p_SDP));
            vc_DATA.start (f_data_startStreaming(p_StreamId,
                vc_DATA.done;
            }

            /**
            *
            * @desc This function trigger the streaming server to stop
» the streaming of a file
            * @param p_StreamId the id of the stream which should
» stopped

            * @verdict
            *   pass in case that the stop streaming request was
» successful

            * @verdict
            *   fail A wrong message was received.
            * @verdict
            *   inconc A guard timer expires.
            */
            function f_main_data_stopStreaming(UInt p_StreamId) runs on
» BCastMainComponent {
                vc_DATA.start (f_data_stopStreaming(p_StreamId));
                vc_DATA.done;
            }
        }

        group upperTesterMainFunctions {

            /**
            *
            * @desc
            *   This function creates a test component and starts a

```

## Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

» function that
    *      sends a select language request from the upper tester
» component.
    * @param
    *   p_langType the language type (audio or text)
    * @param
    *   p_langId the country id for the specific language
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function
» f_main_ut_SelectLanguage(LibBCast_UpperTesterPrimitives_TypesAndValues.Lang
» uageType p_langType, charstring p_langId) runs on BCastMainComponent {

        vc_UTC.start
» (f_ut_sendSelectLanguageRequest(p_langType, p_langId));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *      sends an update service guide request from the upper
» tester component.
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_UpdateServiceGuide() runs on
» BCastMainComponent {
        vc_UTC.start (f_ut_sendUpdateServiceGuideRequest());
        vc_UTC.done;

    }

    /**

```

```

» function that
    *      sends a select language request from the upper tester
» component.
    * @param
    *   p_langType the language type (audio or text)
    * @param
    *   p_langId the country id for the specific language
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function
» f_main_ut_SelectLanguage(LibBCast_UpperTesterPrimitives_TypesAndValues.Lang
» uageType p_langType, charstring p_langId) runs on BCastMainComponent {

        vc_UTC.start
» (f_ut_sendSelectLanguageRequest(p_langType, p_langId));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *      sends an update service guide request from the upper
» tester component.
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_UpdateServiceGuide() runs on
» BCastMainComponent {
        vc_UTC.start (f_ut_sendUpdateServiceGuideRequest());
        vc_UTC.done;

    }

    /**

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a power on request from the upper tester
» component

    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_PowerOn() runs on BCastMainComponent {
        vc_UTC.start (f_ut_sendPowerOnRequest());
        vc_UTC.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a power off request from the upper tester
» component

    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_PowerOff() runs on BCastMainComponent {
        vc_UTC.start (f_ut_sendPowerOffRequest());
        vc_UTC.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a run BCAST application request from the upper
» tester component

    * @verdict
    *   pass The request was successful
    * @verdict

```

```

    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a power on request from the upper tester
» component

    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_PowerOn() runs on BCastMainComponent {
        vc_UTC.start (f_ut_sendPowerOnRequest());
        vc_UTC.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a power off request from the upper tester
» component

    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_PowerOff() runs on BCastMainComponent {
        vc_UTC.start (f_ut_sendPowerOffRequest());
        vc_UTC.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a run BCAST application request from the upper
» tester component

    * @verdict
    *   pass The request was successful
    * @verdict

```

## Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * fail A wrong message was received
        * @verdict
        * inconc A guard timer expires
        */
        function f_main_ut_RunBCastApplication() runs on
» BCastMainComponent {
            vc_UTC.start (f_ut_RunBCastApplication());
            vc_UTC.done;
        }

        /**
        *
        * @desc
        * This function creates a test component and starts a
» function that
        * sends a clear service guide cache request from the
» upper
        * tester component.
        * @verdict
        * pass in case the clear service guide cache request was
        * successful.
        * @verdict
        * inconc case that the request was not successful or the
» timer
        * runs out of time.
        */
        function f_main_ut_ClearServiceGuide() runs on
» BCastMainComponent {
            vc_UTC.start (f_ut_ClearServiceGuideCache());
            vc_UTC.done;
        }

        /**
        *
        * @desc
        * This function creates a test component and starts a
» function that
        * sends a service check request from the upper tester
» component and expects
        * a success response back.
        * @param
        * p_content The name of the content displayed to the
» user, e.g., program 1
        * @verdict
        * pass in case service check request was successful.

```

```

        * fail A wrong message was received
        * @verdict
        * inconc A guard timer expires
        */
        function f_main_ut_RunBCastApplication() runs on
» BCastMainComponent {
            vc_UTC.start (f_ut_RunBCastApplication());
            vc_UTC.done;
        }

        /**
        *
        * @desc
        * This function creates a test component and starts a
» function that
        * sends a clear service guide cache request from the
» upper
        * tester component.
        * @verdict
        * pass in case the clear service guide cache request was
        * successful.
        * @verdict
        * inconc case that the request was not successful or the
» timer
        * runs out of time.
        */
        function f_main_ut_ClearServiceGuide() runs on
» BCastMainComponent {
            vc_UTC.start (f_ut_ClearServiceGuideCache());
            vc_UTC.done;
        }

        /**
        *
        * @desc
        * This function creates a test component and starts a
» function that
        * sends a service check request from the upper tester
» component and expects
        * a success response back.
        * @param
        * p_content The name of the content displayed to the
» user, e.g., program 1
        * @verdict
        * pass in case service check request was successful.

```



Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @verdict
        *   inconc case that the request was not successful or the
» timer
        *   runs out of time.
        */
        function f_main_ut_CheckService(charstring p_content) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckService(p_content));
            vc_UTC.done;

        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a Use Service Request from the upper tester
» component and expects a
    *   success response back.
    * @param
    *   p_service The name of the service to be used (e.g.,
» viewed) by
    *   the user, e.g., TV Channel 1
    * @verdict
    *   pass in case service check request was successful.
    * @verdict
    *   inconc case that the request was not successful or the
» timer
    *   runs out of time.
    */
    function f_main_ut_UseService(charstring p_service) runs on
» BCastMainComponent {
        vc_UTC.start(f_ut_UseService(p_service));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a Get Service Guide Request from the upper
» tester component and expects a
    *   success response back.
    * @param

```

```

        * @verdict
        *   inconc case that the request was not successful or the
» timer
        *   runs out of time.
        */
        function f_main_ut_CheckService(charstring p_content) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckService(p_content));
            vc_UTC.done;

        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a Use Service Request from the upper tester
» component and expects a
    *   success response back.
    * @param
    *   p_service The name of the service to be used (e.g.,
» viewed) by
    *   the user, e.g., TV Channel 1
    * @verdict
    *   pass in case service check request was successful.
    * @verdict
    *   inconc case that the request was not successful or the
» timer
    *   runs out of time.
    */
    function f_main_ut_UseService(charstring p_service) runs on
» BCastMainComponent {
        vc_UTC.start(f_ut_UseService(p_service));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a Get Service Guide Request from the upper
» tester component and expects a
    *   success response back.
    * @param

```

## Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

    *      p_ServiceGuideIdentifier The name of the service guide
» to retrieved
    * @verdict
    *      pass in case service check request was successful.
    * @verdict
    *      inconc case that the request was not successful or the
» timer
    *      runs out of time.
    */
    function f_main_ut_GetServiceGuide(charstring
» p_ServiceGuideIdentifier) runs on BCastMainComponent {
        vc_UTC.start(f_ut_GetServiceGuide(
» p_ServiceGuideIdentifier));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *      This function creates a test component and starts a
» function that sends a select service guide request to the
    *      upper tester.
    * @param
    *      p_service The name of the service to be selected by
» the user, e.g., TV Channel 1
    * @verdict
    *      pass The request was successful
    * @verdict
    *      fail A wrong message was received
    * @verdict
    *      inconc A guard timer expires
    */
    function f_main_ut_SelectService(charstring p_service) runs
» on BCastMainComponent {
        vc_UTC.start(f_ut_SelectService(p_service));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *      This function creates a test component and starts a
» function that
    *      sends a check content request from the upper tester
» component.

```

```

    *      p_ServiceGuideIdentifier The name of the service guide
» to retrieved
    * @verdict
    *      pass in case service check request was successful.
    * @verdict
    *      inconc case that the request was not successful or the
» timer
    *      runs out of time.
    */
    function f_main_ut_GetServiceGuide(charstring
» p_ServiceGuideIdentifier) runs on BCastMainComponent {
        vc_UTC.start(f_ut_GetServiceGuide(
» p_ServiceGuideIdentifier));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *      This function creates a test component and starts a
» function that sends a select service guide request to the
    *      upper tester.
    * @param
    *      p_service The name of the service to be selected by
» the user, e.g., TV Channel 1
    * @verdict
    *      pass The request was successful
    * @verdict
    *      fail A wrong message was received
    * @verdict
    *      inconc A guard timer expires
    */
    function f_main_ut_SelectService(charstring p_service) runs
» on BCastMainComponent {
        vc_UTC.start(f_ut_SelectService(p_service));
        vc_UTC.done;

    }

    /**
    *
    * @desc
    *      This function creates a test component and starts a
» function that
    *      sends a check content request from the upper tester
» component.

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @param
        *   p_content The name of the content displayed to the
» user

        * @param
        *   p_start Start of broadcasting of content displayed to
» the

        *   user, e.g., Program 1
        * @param
        *   p_end End of broadcasting of content displayed to the
» user

        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckContent(charstring p_content,
» template DateTime p_start, template DateTime p_end) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckContent(p_content, p_start,
» p_end));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check interactivity request from the upper
» tester component.
        * @param
        *   p_interactivity Name of the interactivity displayed to
» the user, e.g., vote
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckInteractivity(charstring
» p_interactivity) runs on BCastMainComponent {
» vc_UTC.start(f_ut_CheckInteractivity(p_interactivity));
            vc_UTC.done;

```

```

        * @param
        *   p_content The name of the content displayed to the
» user

        * @param
        *   p_start Start of broadcasting of content displayed to
» the

        *   user, e.g., Program 1
        * @param
        *   p_end End of broadcasting of content displayed to the
» user

        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckContent(charstring p_content,
» template DateTime p_start, template DateTime p_end) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckContent(p_content, p_start,
» p_end));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check interactivity request from the upper
» tester component.
        * @param
        *   p_interactivity Name of the interactivity displayed to
» the user, e.g., vote
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckInteractivity(charstring
» p_interactivity) runs on BCastMainComponent {
» vc_UTC.start(f_ut_CheckInteractivity(p_interactivity));
            vc_UTC.done;

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

    }

    /**
     *
     * @desc
     *   This function creates a test component and starts a
» function that
     *   sends a select interactivity request from the upper
» tester component.
     * @param
     *   p_interactivity Name of the interactivity to be
» selected to the user, e.g., vote
     * @verdict
     *   pass The request was successful
     * @verdict
     *   fail A wrong message was received
     * @verdict
     *   inconc A guard timer expires
     */
    function f_main_ut_SelectInteractivity(charstring
» p_interactivity) runs on BCastMainComponent {

» vc_UTC.start(f_ut_SelectInteractivity(p_interactivity));
    vc_UTC.done;
    }

    /**
     *
     * @desc
     *   This function creates a test component and starts a
» function that
     *   sends a check interactivity choices request from the
» upper tester component.
     * @param
     *   p_choices Name of the interactivity choices displayed
» to the
     *   user
     * @verdict
     *   pass The request was successful
     * @verdict
     *   fail A wrong message was received
     * @verdict
     *   inconc A guard timer expires
     */
    function
» f_main_ut_CheckInteractivityChoices(LanguageStringList p_ChoiceTexts) runs
» on BCastMainComponent {

```

```

    }

    /**
     *
     * @desc
     *   This function creates a test component and starts a
» function that
     *   sends a select interactivity request from the upper
» tester component.
     * @param
     *   p_interactivity Name of the interactivity to be
» selected to the user, e.g., vote
     * @verdict
     *   pass The request was successful
     * @verdict
     *   fail A wrong message was received
     * @verdict
     *   inconc A guard timer expires
     */
    function f_main_ut_SelectInteractivity(charstring
» p_interactivity) runs on BCastMainComponent {

» vc_UTC.start(f_ut_SelectInteractivity(p_interactivity));
    vc_UTC.done;
    }

    /**
     *
     * @desc
     *   This function creates a test component and starts a
» function that
     *   sends a check interactivity choices request from the
» upper tester component.
     * @param
     *   p_choices Name of the interactivity choices displayed
» to the
     *   user
     * @verdict
     *   pass The request was successful
     * @verdict
     *   fail A wrong message was received
     * @verdict
     *   inconc A guard timer expires
     */
    function
» f_main_ut_CheckInteractivityChoices(LanguageStringList p_ChoiceTexts) runs
» on BCastMainComponent {

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

» vc_UTC.start(f_ut_CheckInteractivityChoice(p_ChoiceTexts));
    vc_UTC.done;
}

/**
 *
 * @desc
 *   This function creates a test component and starts a
» function that
 *   sends a select interactivity coice request to
 *   the upper tester.
 * @param
 *   p_choice Name of the interactivity choice to be
» selected by the
 *   user, e.g., choice a
 * @verdict
 *   pass The request was successful
 * @verdict
 *   fail A wrong message was received
 * @verdict
 *   inconc A guard timer expires
 */
function f_main_ut_SelectInteractivityChoice(charstring
» p_choice) runs on BCastMainComponent {

» vc_UTC.start(f_ut_SelectInteractivityChoice(p_choice));
    vc_UTC.done;
}

/**
 *
 * @desc
 *   This function creates a test component and starts a
» function that
 *   sends a check audio language request from the upper
» tester component.
 * @param
 *   p_content The identifer of the content displayed to
» the user, e.g., Program 1
 * @param
 *   p_type Type of language setting to be changed by the
» user
 * @param
 *   p_lang Language that the user shall hear
 * @verdict
 *   pass The request was successful

```

```

» vc_UTC.start(f_ut_CheckInteractivityChoice(p_ChoiceTexts));
    vc_UTC.done;
}

/**
 *
 * @desc
 *   This function creates a test component and starts a
» function that
 *   sends a select interactivity coice request to
 *   the upper tester.
 * @param
 *   p_choice Name of the interactivity choice to be
» selected by the
 *   user, e.g., choice a
 * @verdict
 *   pass The request was successful
 * @verdict
 *   fail A wrong message was received
 * @verdict
 *   inconc A guard timer expires
 */
function f_main_ut_SelectInteractivityChoice(charstring
» p_choice) runs on BCastMainComponent {

» vc_UTC.start(f_ut_SelectInteractivityChoice(p_choice));
    vc_UTC.done;
}

/**
 *
 * @desc
 *   This function creates a test component and starts a
» function that
 *   sends a check audio language request from the upper
» tester component.
 * @param
 *   p_content The identifer of the content displayed to
» the user, e.g., Program 1
 * @param
 *   p_type Type of language setting to be changed by the
» user
 * @param
 *   p_lang Language that the user shall hear
 * @verdict
 *   pass The request was successful

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckLanguage(charstring p_content,
» LibBCast_UpperTesterPrimitives_TypesAndValues.LanguageType p_type,
» charstring p_lang) runs on BCastMainComponent {
            vc_UTC.start(f_ut_CheckLanguage(p_content, p_type,
» p_lang));
            vc_UTC.done;
        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a check browser request from the upper tester
» component.
    * @param
    *   p_xhtml Reference or description of the content of the
» xhtml
    *   file to be displayed to the user
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_CheckBrowser(charstring p_xhtml) runs on
» BCastMainComponent {
        vc_UTC.start(f_ut_CheckBrowser(p_xhtml));
        vc_UTC.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a check preview request from the upper tester
» component.
    * @param
    *   p_data Reference to icon or text to be displayed to
» the user, e.g., TV Channel logo

```

```

        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckLanguage(charstring p_content,
» LibBCast_UpperTesterPrimitives_TypesAndValues.LanguageType p_type,
» charstring p_lang) runs on BCastMainComponent {
            vc_UTC.start(f_ut_CheckLanguage(p_content, p_type,
» p_lang));
            vc_UTC.done;
        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a check browser request from the upper tester
» component.
    * @param
    *   p_xhtml Reference or description of the content of the
» xhtml
    *   file to be displayed to the user
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
    function f_main_ut_CheckBrowser(charstring p_xhtml) runs on
» BCastMainComponent {
        vc_UTC.start(f_ut_CheckBrowser(p_xhtml));
        vc_UTC.done;
    }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a check preview request from the upper tester
» component.
    * @param
    *   p_data Reference to icon or text to be displayed to
» the user, e.g., TV Channel logo

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckPreview(charstring p_data) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckPreview(p_data));
            vc_UTC.done;
        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a select file request from the upper tester
» component.
    * @param
    *   p_file File name to be selected by the user, e.g.,
» foo.c
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
        function f_main_ut_SelectFile(charstring p_file) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_SelectFile(p_file));
            vc_UTC.done;
        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a check file request from the upper tester
» component.
    * @param
    *   p_file Reference to file name or description of its
» content
    *   to be displayed to the user, e.g., foo.c

```

```

        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckPreview(charstring p_data) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckPreview(p_data));
            vc_UTC.done;
        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a select file request from the upper tester
» component.
    * @param
    *   p_file File name to be selected by the user, e.g.,
» foo.c
    * @verdict
    *   pass The request was successful
    * @verdict
    *   fail A wrong message was received
    * @verdict
    *   inconc A guard timer expires
    */
        function f_main_ut_SelectFile(charstring p_file) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_SelectFile(p_file));
            vc_UTC.done;
        }

    /**
    *
    * @desc
    *   This function creates a test component and starts a
» function that
    *   sends a check file request from the upper tester
» component.
    * @param
    *   p_file Reference to file name or description of its
» content
    *   to be displayed to the user, e.g., foo.c

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckFile(charstring p_file) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckFile(p_file));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check file received request from the upper
» tester component.
        * @param
        *   p_file File name or description of its content that is
» to be received
        *   by the user, e.g., foo.c
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckFileReceived(charstring p_file) runs
» on BCastMainComponent {
            vc_UTC.start(f_ut_CheckFileReceived(p_file));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check video request from the upper tester
» component.
        * @param
        *   p_video Reference to video name or description of its
» content

```

```

        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckFile(charstring p_file) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckFile(p_file));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check file received request from the upper
» tester component.
        * @param
        *   p_file File name or description of its content that is
» to be received
        *   by the user, e.g., foo.c
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckFileReceived(charstring p_file) runs
» on BCastMainComponent {
            vc_UTC.start(f_ut_CheckFileReceived(p_file));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check video request from the upper tester
» component.
        * @param
        *   p_video Reference to video name or description of its
» content

```



Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        *   to be displayed to the user, e.g., starwars
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckVideo(charstring p_video) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckVideo(p_video));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check purchase info request from the upper
» tester component.
        * @param
        *   p_info The purchase info content to be displayed to
» the user
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckPurchaseInfo(charstring p_info) runs
» on BCastMainComponent {
            vc_UTC.start(f_ut_CheckPurchaseInfo(p_info));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a purchase service request from the upper tester
» component.

```

```

        *   to be displayed to the user, e.g., starwars
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckVideo(charstring p_video) runs on
» BCastMainComponent {
            vc_UTC.start(f_ut_CheckVideo(p_video));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a check purchase info request from the upper
» tester component.
        * @param
        *   p_info The purchase info content to be displayed to
» the user
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_CheckPurchaseInfo(charstring p_info) runs
» on BCastMainComponent {
            vc_UTC.start(f_ut_CheckPurchaseInfo(p_info));
            vc_UTC.done;
        }

        /**
        *
        * @desc
        *   This function creates a test component and starts a
» function that
        *   sends a purchase service request from the upper tester
» component.

```

Datei: AtsBCast\_Main\_Functions.ttcn (Fortsetzung)

```

        * @param
        *   p_service The name of the service to be purchased by
» the user, e.g., TV Channel 1
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_PurchaseService(charstring p_service) runs
» on BCastMainComponent {
            vc_UTC.start(f_ut_CheckPurchaseInfo(p_service));
            vc_UTC.done;
        }

    }
}

```

```

        * @param
        *   p_service The name of the service to be purchased by
» the user, e.g., TV Channel 1
        * @verdict
        *   pass The request was successful
        * @verdict
        *   fail A wrong message was received
        * @verdict
        *   inconc A guard timer expires
        */
        function f_main_ut_PurchaseService(charstring p_service) runs
» on BCastMainComponent {
            vc_UTC.start(f_ut_CheckPurchaseInfo(p_service));
            vc_UTC.done;
        }

    }
}

```

Datei: AtsBCast\_ModuleParameters.ttcn

```

/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies ATS specific module parameters and their default
» values
 *   which can be still modified in their value as late as just prior to
 *   test case execution. The parameters in this file OMA BCAST specific.
 *   Module parameters realize so called PIXIT.
 */
module AtsBCast_ModuleParameters {
    import from LibBCast_UpperTesterPrimitives_TypesAndValues all;
    import from LibCommon_TextStrings {
        const c_CRLF
    }

    group testCaseSelectionSwitches {
        /**
         *
         * @desc Specifies to execute all the BCAST test cases
         */
        modulepar boolean PX_ALL_TCS := true;
    }
}

```

```

=
/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies ATS specific module parameters and their default
» values
 *   which can be still modified in their value as late as just prior to
 *   test case execution. The parameters in this file OMA BCAST specific.
 *   Module parameters realize so called PIXIT.
 */
module AtsBCast_ModuleParameters {
    import from LibBCast_UpperTesterPrimitives_TypesAndValues all;
    import from LibCommon_TextStrings {
        const c_CRLF
    }

    group testCaseSelectionSwitches {
        /**
         *
         * @desc Specifies to execute all the BCAST test cases
         */
        modulepar boolean PX_ALL_TCS := true;
    }
}

```

Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        /**
         *
         * @desc Specifies to execute all the Service Provisioning
» test cases
        */
        modulepar boolean PX_ALL_SP_TCS := false;

        /**
         *
         * @desc Specifies to execute all the Service Guide test
» cases
        */
        modulepar boolean PX_ALL_SG_TCS := false;

        /**
         *
         * @desc Specifies to execute all the File and Stream
» Distribution test cases
        */
        modulepar boolean PX_ALL_FD_TCS := false;

        /**
         *
         * @desc Specifies all Service Interaction test cases
        */
        modulepar boolean PX_ALL_SI_TCS := false;

        /**
         *
         * @desc Specifies to execute all the Content Protection test
» cases
        */
        modulepar boolean PX_ALL_CP_TCS := false;
    } // end group testCaseSelectionSwitches

    group terminalUserApi {

        /**
         *
         * @desc Indicates if BCAST application under test needs
» (manual) operator
         * interaction in order get service guide content displayed.
» If set to false
         * the application is assumed to display the service guide
» automatically.
        */

```

```

        /**
         *
         * @desc Specifies to execute all the Service Provisioning
» test cases
        */
        modulepar boolean PX_ALL_SP_TCS := false;

        /**
         *
         * @desc Specifies to execute all the Service Guide test
» cases
        */
        modulepar boolean PX_ALL_SG_TCS := false;

        /**
         *
         * @desc Specifies to execute all the File and Stream
» Distribution test cases
        */
        modulepar boolean PX_ALL_FD_TCS := false;

        /**
         *
         * @desc Specifies all Service Interaction test cases
        */
        modulepar boolean PX_ALL_SI_TCS := false;

        /**
         *
         * @desc Specifies to execute all the Content Protection test
» cases
        */
        modulepar boolean PX_ALL_CP_TCS := false;
    } // end group testCaseSelectionSwitches

    group terminalUserApi {

        /**
         *
         * @desc Indicates if BCAST application under test needs
» (manual) operator
         * interaction in order get service guide content displayed.
» If set to false
         * the application is assumed to display the service guide
» automatically.
        */

```

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        modulepar boolean PX_GET_SG_DISPLAY_MANUALLY := false;

        /**
         *
         * @desc Indicates if BCAST application under test needs
    » (manual) operator
         * interaction in order get a service guide update displayed.
    » If set to false
         * the application is assumed to display the service guide
    » update automatically.
         */
        modulepar boolean PX_GET_SG_UPDATE_MANUALLY := false;

        } // end group terminalUserApi

        group ServiceGuideIds {

            group ServiceFragment {
                /**
                 *
                 * @desc
                 * Specifies the globally unique URI for the
    » Service fragment
                 */
                modulepar charstring PX_SGDU_SERVICE_ID :=

    » "bcast://bcast.testingtech.com/service/service_id";

            }

            group ContentFragment {
                /**
                 *
                 * @desc
                 * Specifies the globally unique URI for the
    » Content fragment.
                 */
                modulepar charstring PX_SGDU_CONTENT_ID_PROG_1 :=

    » "bcast://bcast.testingtech.com/content/content_id_1";

                /**
                 *
                 * @desc
                 * Specifies the globally unique URI for the
    » Content fragment.
                 */

```

```

        modulepar boolean PX_GET_SG_DISPLAY_MANUALLY := false;

        /**
         *
         * @desc Indicates if BCAST application under test needs
    » (manual) operator
         * interaction in order get a service guide update displayed.
    » If set to false
         * the application is assumed to display the service guide
    » update automatically.
         */
        modulepar boolean PX_GET_SG_UPDATE_MANUALLY := false;

        } // end group terminalUserApi

        group ServiceGuideIds {

            group ServiceFragment {
                /**
                 *
                 * @desc
                 * Specifies the globally unique URI for the
    » Service fragment
                 */
                modulepar charstring PX_SGDU_SERVICE_ID :=

    » "bcast://bcast.testingtech.com/service/service_id";

            }

            group ContentFragment {
                /**
                 *
                 * @desc
                 * Specifies the globally unique URI for the
    » Content fragment.
                 */
                modulepar charstring PX_SGDU_CONTENT_ID_PROG_1 :=

    » "bcast://bcast.testingtech.com/content/content_id_1";

                /**
                 *
                 * @desc
                 * Specifies the globally unique URI for the
    » Content fragment.
                 */

```

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        modulepar charstring PX_SGDU_CONTENT_ID_PROG_2 :=
» "bcast://bcast.testingtech.com/content/content_id_2";

    }

    group ScheduleFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Schedule fragment.
         */
        modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_1 :=
» "bcast://bcast.testingtech.com/schedule/schedule_id_1";

        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Schedule fragment.
         */
        modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_2 :=
» "bcast://bcast.testingtech.com/schedule/schedule_id_2";
    }

    group AccessFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Access fragment
         */
        modulepar charstring PX_SGDU_ACCESS_ID_PROG_1 :=
» "bcast://bcast.testingtech.com/access/access_id_1";

        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Access fragment
         */
        modulepar charstring PX_SGDU_ACCESS_ID_PROG_2 :=

```

```

        modulepar charstring PX_SGDU_CONTENT_ID_PROG_2 :=
» "bcast://bcast.testingtech.com/content/content_id_2";

    }

    group ScheduleFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Schedule fragment.
         */
        modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_1 :=
» "bcast://bcast.testingtech.com/schedule/schedule_id_1";

        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Schedule fragment.
         */
        modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_2 :=
» "bcast://bcast.testingtech.com/schedule/schedule_id_2";
    }

    group AccessFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Access fragment
         */
        modulepar charstring PX_SGDU_ACCESS_ID_PROG_1 :=
» "bcast://bcast.testingtech.com/access/access_id_1";

        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Access fragment
         */
        modulepar charstring PX_SGDU_ACCESS_ID_PROG_2 :=

```

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

» "bcast://bcast.testingtech.com/access/access_id_2";
    }

    group PreviewDataFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PreviewData fragment
         */
        modulepar charstring PX_SGDU_PREVIEW_DATA_ID :=

» "bcast://bcast.testingtech.com/previewData/previewData_id_1";
    }

    group PurchaseItemFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PurchaseItem fragment
         */
        modulepar charstring PX_SGDU_PURCHASE_ITEM_ID :=

» "bcast://bcast.testingtech.com/purchaseItem/purchaseItem_id_1";
    }

    group PurchaseDataFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PurchaseData fragment
         */
        modulepar charstring PX_SGDU_PURCHASE_DATA_ID :=

» "bcast://bcast.testingtech.com/purchaseData/purchaseData_id_1";
    }

    group PurchaseChannelFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PurchaseChannel fragment
         */
        modulepar charstring PX_SGDU_PURCHASE_CHANNEL_ID :=

```

```

» "bcast://bcast.testingtech.com/access/access_id_2";
    }

    group PreviewDataFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PreviewData fragment
         */
        modulepar charstring PX_SGDU_PREVIEW_DATA_ID :=

» "bcast://bcast.testingtech.com/previewData/previewData_id_1";
    }

    group PurchaseItemFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PurchaseItem fragment
         */
        modulepar charstring PX_SGDU_PURCHASE_ITEM_ID :=

» "bcast://bcast.testingtech.com/purchaseItem/purchaseItem_id_1";
    }

    group PurchaseDataFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PurchaseData fragment
         */
        modulepar charstring PX_SGDU_PURCHASE_DATA_ID :=

» "bcast://bcast.testingtech.com/purchaseData/purchaseData_id_1";
    }

    group PurchaseChannelFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» PurchaseChannel fragment
         */
        modulepar charstring PX_SGDU_PURCHASE_CHANNEL_ID :=

```

Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

» "bcast://bcast.testingtech.com/purchaseChannel/purchaseChannel_id_1";
    }

    group InteractivityDataFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Interactivity fragment
         */
        modulepar charstring PX_SGDU_INTERACTIVITY_DATA_ID :=
» "bacast://bcast.testingtech.com/interactivityData/interactivityData_id_1";
    }

    group PurchasingData {
        /**
         *
         * @desc
         *     Specifies the PurchaseURL in the PurchaseChannel
» fragment
         */
        modulepar charstring PX_PURCHASE_URL :=
            "bcast://bcast.testingtech.com/purchaseUrl";

        /**
         *
         * @desc
         *     Specifies a value for the PriceInfo element of the
» PurchaseItem.
         */
        modulepar float PX_MONETARY_PRICE := 15.00;

        /**
         *
         * @desc
         *     Specifies the currency associated with the
» PX_MONETARY_PRICE, e.g., eur
         */
        modulepar charstring PX_CURRENCY := "EUR";
    }

    group InteractivityMediaDocument {
        /**

```

```

» "bcast://bcast.testingtech.com/purchaseChannel/purchaseChannel_id_1";
    }

    group InteractivityDataFragment {
        /**
         *
         * @desc
         *     Specifies the globally unique URI for the
» Interactivity fragment
         */
        modulepar charstring PX_SGDU_INTERACTIVITY_DATA_ID :=
» "bacast://bcast.testingtech.com/interactivityData/interactivityData_id_1";
    }

    group PurchasingData {
        /**
         *
         * @desc
         *     Specifies the PurchaseURL in the PurchaseChannel
» fragment
         */
        modulepar charstring PX_PURCHASE_URL :=
            "bcast://bcast.testingtech.com/purchaseUrl";

        /**
         *
         * @desc
         *     Specifies a value for the PriceInfo element of the
» PurchaseItem.
         */
        modulepar float PX_MONETARY_PRICE := 15.00;

        /**
         *
         * @desc
         *     Specifies the currency associated with the
» PX_MONETARY_PRICE, e.g., eur
         */
        modulepar charstring PX_CURRENCY := "EUR";
    }

    group InteractivityMediaDocument {
        /**

```

Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        *
        * @desc
        *     Specifies the "id" of the globally unique URI of
» the InteractivityMediaDocument
        */
        modulepar charstring PX_MEDIA_DOCUMENT_ID :=
            "InteractivityMediaDocumentId";

        /**
        *
        * @desc
        *     TSpecifies the globally unique URI of the
» Interactivity Media Document Group
        */
        modulepar charstring PX_MEDIA_DOCUMENT_GROUP_ID :=
            "InteractivityMediaDocumentGroupId";

        /**
        *
        * @desc
        *     Specifies the preListenIndicator of the
» InteractivityData fragment
        */
        modulepar boolean PX_PRELISTEN_INDICATOR := false;

        /**
        *
        * @desc
        *     Specifies the ContentLocation of the MediaObjectSet
        */
        modulepar charstring PX_MEDIA_OBJECT_SET_URI :=
            "http://www.testingtech.com/mediaObjectSet.gz";
    }

    group SMS {
        /**
        *
        * @desc
        *     Specifies the URI of the SMS
        */
        modulepar charstring PX_SMS_URI :=
» "http://www.openmobilealliance.org/";
    }

    group MMS {
        /**
        *

```

```

        *
        * @desc
        *     Specifies the "id" of the globally unique URI of
» the InteractivityMediaDocument
        */
        modulepar charstring PX_MEDIA_DOCUMENT_ID :=
            "InteractivityMediaDocumentId";

        /**
        *
        * @desc
        *     TSpecifies the globally unique URI of the
» Interactivity Media Document Group
        */
        modulepar charstring PX_MEDIA_DOCUMENT_GROUP_ID :=
            "InteractivityMediaDocumentGroupId";

        /**
        *
        * @desc
        *     Specifies the preListenIndicator of the
» InteractivityData fragment
        */
        modulepar boolean PX_PRELISTEN_INDICATOR := false;

        /**
        *
        * @desc
        *     Specifies the ContentLocation of the MediaObjectSet
        */
        modulepar charstring PX_MEDIA_OBJECT_SET_URI :=
            "http://www.testingtech.com/mediaObjectSet.gz";
    }

    group SMS {
        /**
        *
        * @desc
        *     Specifies the URI of the SMS
        */
        modulepar charstring PX_SMS_URI :=
» "http://www.openmobilealliance.org/";
    }

    group MMS {
        /**
        *

```



Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        * @desc
        *   Specifies the file name of a MMS template
        */
modulepar charstring PX_MMS_TEMPLATE := "vote.mtd";

/**
 *
 * @desc
 *   Specifies the file name of a picture
 */
modulepar charstring PX_MMS_VOTE_A_PICTURE := "ChoiceA.png";

/**
 *
 * @desc
 *   Specifies the file name of a picture
 */
modulepar charstring PX_MMS_VOTE_PICTURE := "ChoiceB.jpg";

/**
 *
 * @desc
 *   Specifies the file name of a text file
 */
modulepar charstring PX_MMS_TEXT_FILE := "Instructions.txt";
}

group XHTML {
    /**
    *
    * @desc
    *   Specifies the file name of the XHTML file
    */
    modulepar charstring PX_XHTML_FILE_ID := "vote-xhtml";

    /**
    *
    * @desc
    *   Specifies the content type of the XHTML file
    */
    modulepar charstring PX_XHTML_CONTENT_TYPE :=
        "application/vnd.wap.xhtml+xml";
}

group FileDelivery {
    /**
    *

```

```

        * @desc
        *   Specifies the file name of a MMS template
        */
modulepar charstring PX_MMS_TEMPLATE := "vote.mtd";

/**
 *
 * @desc
 *   Specifies the file name of a picture
 */
modulepar charstring PX_MMS_VOTE_A_PICTURE := "ChoiceA.png";

/**
 *
 * @desc
 *   Specifies the file name of a picture
 */
modulepar charstring PX_MMS_VOTE_PICTURE := "ChoiceB.jpg";

/**
 *
 * @desc
 *   Specifies the file name of a text file
 */
modulepar charstring PX_MMS_TEXT_FILE := "Instructions.txt";
}

group XHTML {
    /**
    *
    * @desc
    *   Specifies the file name of the XHTML file
    */
    modulepar charstring PX_XHTML_FILE_ID := "vote-xhtml";

    /**
    *
    * @desc
    *   Specifies the content type of the XHTML file
    */
    modulepar charstring PX_XHTML_CONTENT_TYPE :=
        "application/vnd.wap.xhtml+xml";
}

group FileDelivery {
    /**
    *

```

Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        * @desc
        *   Specifies the file name of a picture, e.g., file1.jpg
        */
modulepar charstring PX_PICTURE_ID := "TvChannel.jpg";

/**
 *
 * @desc
 *   Specifies the file name containing video, e.g.,
» video1.mov
 */
modulepar charstring PX_FILE_VIDEO_PROG1 := "video1.mov";

/**
 *
 * @desc
 *   Specifies the file name containing video, e.g.,
» video2.mov
 */
modulepar charstring PX_FILE_VIDEO_PROG2 := "video2.mov";

/**
 *
 * @desc
 *   Specifies the file name containing audio, e.g.,
» audio1.mp3
 */
modulepar charstring PX_FILE_AUDIO_PROG_1 := "audio1.mp3";

/**
 *
 * @desc
 *   Specifies the file name containing audio, e.g.,
» audio2.mp3
 */
modulepar charstring PX_FILE_AUDIO_PROG_2 := "audio2.mp3";

/**
 *
 * @desc
 *   Specifies the file name containing audio, e.g.,
» audio_eng.mp3
 */
modulepar charstring PX_FILE_AUDIO_ENG := "audio_eng.mp3";

/**
 *
```

```

        * @desc
        *   Specifies the file name of a picture, e.g., file1.jpg
        */
modulepar charstring PX_PICTURE_ID := "TvChannel.jpg";

/**
 *
 * @desc
 *   Specifies the file name containing video, e.g.,
» video1.mov
 */
modulepar charstring PX_FILE_VIDEO_PROG1 := "video1.mov";

/**
 *
 * @desc
 *   Specifies the file name containing video, e.g.,
» video2.mov
 */
modulepar charstring PX_FILE_VIDEO_PROG2 := "video2.mov";

/**
 *
 * @desc
 *   Specifies the file name containing audio, e.g.,
» audio1.mp3
 */
modulepar charstring PX_FILE_AUDIO_PROG_1 := "audio1.mp3";

/**
 *
 * @desc
 *   Specifies the file name containing audio, e.g.,
» audio2.mp3
 */
modulepar charstring PX_FILE_AUDIO_PROG_2 := "audio2.mp3";

/**
 *
 * @desc
 *   Specifies the file name containing audio, e.g.,
» audio_eng.mp3
 */
modulepar charstring PX_FILE_AUDIO_ENG := "audio_eng.mp3";

/**
 *
```

Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

<pre>         * @desc         * Specifies the file name containing audio, e.g., » audio_ger.mp3         */         modulepar charstring PX_FILE_AUDIO_GER := "audio_ger.mp3";          /**         *         * @desc         * Specifies the time in seconds for the video to play         */         modulepar integer PX_PLAY_TIME := 300;     } </pre>		<pre>         * @desc         * Specifies the file name containing audio, e.g., » audio_ger.mp3         */         modulepar charstring PX_FILE_AUDIO_GER := "audio_ger.mp3";          /**         *         * @desc         * Specifies the time in seconds for the video to play         */         modulepar integer PX_PLAY_TIME := 300;     } </pre>
<pre>         group FluteSessionParameters { » // change 02 (WK 6): default values for FLUTE parameters as PIXIT </pre>	<>	<pre>         group SessionDescription { </pre>
<pre>         /**         *         * @desc </pre>	=	<pre>         /**         *         * @desc </pre>
<pre>         * @desc Specifies the IP address for the DVB-H » bootstrapping session         */         modulepar charstring PX_DVB_H_FLUTE_SESSION_IP := » "224.00.23.14";           // change 02 (WK 6): default values for FLUTE » parameters as PIXIT          /**         * @desc Specifies the port for the DVB-H bootstrapping » session         */         modulepar integer PX_DVB_H_FLUTE_SESSION_PORT := 9214; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT          /**         * @desc Specifies the IP address for the Flute SGDD session         */         modulepar charstring PX_SGDD_FLUTE_SESSION_IP := "232.0.1.0"; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT          /**         * @desc Specifies the port for the Flute SGDD session         */         modulepar integer PX_SGDD_FLUTE_SESSION_PORT := 9000; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT          /** </pre>	<>	

Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

<pre>         * @desc Specifies the IP address for the Flute SGDU session         */         modulepar charstring PX_SGDU_FLUTE_SESSION_IP := "232.0.1.0"; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT          /**         * @desc Specifies the port for the Flute SGDU session         */         modulepar integer PX_SGDU_FLUTE_SESSION_PORT := 9003; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT          /**         * @desc Specifies the IP address for a Flute data session         */         modulepar charstring PX_DATA_FLUTE_SESSION_IP := "232.0.7.0"; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT          /**         * @desc Specifies the port for a Flute data session         */         modulepar integer PX_DATA_FLUTE_SESSION_PORT := 10080; » // change 02 (WK 6): default values for FLUTE parameters as PIXIT     } </pre>		
<pre>     group SessionDescription {         /**         * @desc Specifies the ID of the SDP         */ </pre>		<pre>         * Specifies the ID of the SDP         */ </pre>
<pre>         modulepar charstring PX_SDP_VIDEO_PROG_1_REF_ID := » "myFirstSDP";          /** </pre>	=	<pre>         modulepar charstring PX_SDP_VIDEO_PROG_1_REF_ID := » "myFirstSDP";          /** </pre>
<pre>         * @desc Specifies the ID of the SDP </pre>	<>	<pre>         *         * @desc         * Specifies the ID of the SDP </pre>
<pre>         */         modulepar charstring PX_SDP_VIDEO_PROG_2_REF_ID := » "mySecondSDP";          /**         *         * @desc         * Specifies an SDP with the default settings         */         modulepar charstring PX_SDP_VIDEO_PROG_1 :=             "v=0\r\no=- 424 3292855200 IN IP6 » FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6 </pre>	=	<pre>         */         modulepar charstring PX_SDP_VIDEO_PROG_2_REF_ID := » "mySecondSDP";          /**         *         * @desc         * Specifies an SDP with the default settings         */         modulepar charstring PX_SDP_VIDEO_PROG_1 :=             "v=0\r\no=- 424 3292855200 IN IP6 » FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6 </pre>

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies an SDP with the default settings
 */

```

```

modulepar charstring PX_SDP_VIDEO_PROG_2 :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies an SDP with a particular language
 */

```

```

modulepar charstring PX_SDP_AUDIO_ENG :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies an SDP with a particular language
 */

```

```

modulepar charstring PX_SDP_AUDIO_GER :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies an SDP with the default settings
 */

```

```

modulepar charstring PX_SDP_VIDEO_PROG_2 :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies an SDP with a particular language
 */

```

```

modulepar charstring PX_SDP_AUDIO_ENG :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies an SDP with a particular language
 */

```

```

modulepar charstring PX_SDP_AUDIO_GER :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies SDP for audio stream of TV Program 1
 */

```

```

modulepar charstring PX_SDP_AUDIO_PROG_1 :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies SDP for audio stream of TV Program 2
 */

```

```

modulepar charstring PX_SDP_AUDIO_PROG_2 :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies the SDP to be used for file delivery over
» FLUTE, e.g., test case BCAST-1.0-DIST-conf-102
 */

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies SDP for audio stream of TV Program 1
 */

```

```

modulepar charstring PX_SDP_AUDIO_PROG_1 :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies SDP for audio stream of TV Program 2
 */

```

```

modulepar charstring PX_SDP_AUDIO_PROG_2 :=
    "v=0\r\no=- 424 3292855200 IN IP6

```

```

» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nt=0 0\r\nm=audio 49172 RTP/AVP
» 96\r\nb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nm=video
» 49170 RTP/AVP 97\r\nb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

```

```

/**
 *
 * @desc
 * Specifies the SDP to be used for file delivery over
» FLUTE, e.g., test case BCAST-1.0-DIST-conf-102
 */

```

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        modulepar charstring PX_SDP_FILE_DELIVERY :=
            "v=0\r\no=user123 2890844526 2890842807 IN IP6
» 2201:056D::112E:144A:1E24\r\ns= Example of file delivery session
» description using FLUTE\r\ni=More information\r\nnt=2873397496
» 2873404696\r\na=FEC-declaration:0 encoding-id=0;\r\na=source-filter: incl
» IN IP6 *
» 2001:210:1:2:240:96FF:FE25:8EC9\r\na=flute-tsi:3\r\na=flute-ch:1\r\nnm=appli
» cation 12345 FLUTE/UDP 0\r\nnc=IN IP6
» FF1E:03AD::7F2E:172A:1E24/1\r\nnb=AS:64\r\na=FEC:0\r\n";

        /**
        *
        * @desc
        *     Specifies the SDP to be used for the Content
» Protection test cases using IPSec, e.g., test case
» BCAST-1.0-ConProt-conf-101
        */
        modulepar charstring PX_SDP_SERVICE_PROTECTION_IPSEC :=
            "v=0\r\no=- 424 3292855200 IN IP6
» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nnc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nnt=0 0\r\nnm=audio 49172 RTP/AVP
» 96\r\nnb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nnm=video
» 49170 RTP/AVP 97\r\nnb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

        /**
        *
        * @desc
        *     Specifies the SDP to be used for the Content Protection
» test cases using SRTP, e.g., test case BCAST-1.0-ConProt-conf-102
        */
        modulepar charstring PX_SDP_SERVICE_PROTECTION_SRTP :=
            "v=0\r\no=- 424 3292855200 IN IP6
» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nnc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nnt=0 0\r\nnm=audio 49172 RTP/AVP
» 96\r\nnb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nnm=video
» 49170 RTP/AVP 97\r\nnb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

        /**
        *

```

```

        modulepar charstring PX_SDP_FILE_DELIVERY :=
            "v=0\r\no=user123 2890844526 2890842807 IN IP6
» 2201:056D::112E:144A:1E24\r\ns= Example of file delivery session
» description using FLUTE\r\ni=More information\r\nnt=2873397496
» 2873404696\r\na=FEC-declaration:0 encoding-id=0;\r\na=source-filter: incl
» IN IP6 *
» 2001:210:1:2:240:96FF:FE25:8EC9\r\na=flute-tsi:3\r\na=flute-ch:1\r\nnm=appli
» cation 12345 FLUTE/UDP 0\r\nnc=IN IP6
» FF1E:03AD::7F2E:172A:1E24/1\r\nnb=AS:64\r\na=FEC:0\r\n";

        /**
        *
        * @desc
        *     Specifies the SDP to be used for the Content
» Protection test cases using IPSec, e.g., test case
» BCAST-1.0-ConProt-conf-101
        */
        modulepar charstring PX_SDP_SERVICE_PROTECTION_IPSEC :=
            "v=0\r\no=- 424 3292855200 IN IP6
» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nnc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nnt=0 0\r\nnm=audio 49172 RTP/AVP
» 96\r\nnb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nnm=video
» 49170 RTP/AVP 97\r\nnb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

        /**
        *
        * @desc
        *     Specifies the SDP to be used for the Content Protection
» test cases using SRTP, e.g., test case BCAST-1.0-ConProt-conf-102
        */
        modulepar charstring PX_SDP_SERVICE_PROTECTION_SRTP :=
            "v=0\r\no=- 424 3292855200 IN IP6
» FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nnc=IN IP6
» FF15:0:0:0:0:0:81:1BD\r\nnt=0 0\r\nnm=audio 49172 RTP/AVP
» 96\r\nnb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
» profile-level-id=15; mode=AAC-hbr; config=1290;
» SizeLength=13;IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nnm=video
» 49170 RTP/AVP 97\r\nnb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
» profile-level-id=42c00d;
» packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";

        /**
        *

```

## Datei: AtsBCast\_ModuleParameters.ttcn (Fortsetzung)

```

        * @desc
        * Specifies the SDP to be used for the Content Protection
    » test cases using ISMACrypt, e.g., test case BCAST-1.0-ConProt-conf-103
        */
        modulepar charstring PX_SDP_SERVICE_PROTECTION_ISMA :=
            "v=0\r\no=- 424 3292855200 IN IP6
    » FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nnc=IN IP6
    » FF15:0:0:0:0:0:81:1BD\r\nnt=0 0\r\nnm=audio 49172 RTP/AVP
    » 96\r\nnb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
    » profile-level-id=15; mode=AAC-hbr; config=1290;
    » SizeLength=13; IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nnm=video
    » 49170 RTP/AVP 97\r\nnb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
    » profile-level-id=42c00d;
    » packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";
    }

    group Misc {
        /**
        *
        * @desc
        * On true, test case runs in simulate mode.
        */
        modulepar boolean PX_SIMULATE_TC := false;
    }

} // end module AtsBCast_ModuleParameters

```

```

        * @desc
        * Specifies the SDP to be used for the Content Protection
    » test cases using ISMACrypt, e.g., test case BCAST-1.0-ConProt-conf-103
        */
        modulepar charstring PX_SDP_SERVICE_PROTECTION_ISMA :=
            "v=0\r\no=- 424 3292855200 IN IP6
    » FF15:0:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV Example\r\nnc=IN IP6
    » FF15:0:0:0:0:0:81:1BD\r\nnt=0 0\r\nnm=audio 49172 RTP/AVP
    » 96\r\nnb=AS:64\r\na=rtpmap:96 mpeg4-generic/32000\r\na=fmtp:96 streamtype=5;
    » profile-level-id=15; mode=AAC-hbr; config=1290;
    » SizeLength=13; IndexLength=3;\r\nIndexDeltaLength=3; Profile=1;\r\nnm=video
    » 49170 RTP/AVP 97\r\nnb=AS:250\r\na=rtpmap:97 H264/90000\r\na=fmtp:97
    » profile-level-id=42c00d;
    » packetization-mode=1;sprop-parameter-sets=Z0LADZtAoPiA,aN4liA==;\r\n";
    }

    group Misc {
        /**
        *
        * @desc
        * On true, test case runs in simulate mode.
        */
        modulepar boolean PX_SIMULATE_TC := false;
    }

} // end module AtsBCast_ModuleParameters

```

## Datei: AtsBCast\_ServiceGuide\_Functions.ttcn

```

/**
*
* @author
* ETSI CTI BCAST CON Project
* @version
* $Id$
* @desc
* This module specifies some function for the service guide.
*/

module AtsBCast_ServiceGuide_Functions {
    import from AtsBCast_TestSystem all;
    import from AtsBCast_ModuleParameters { modulepar PX_SIMULATE_TC };
    import from LibBCast_UpperTesterPrimitives_TypesAndValues all;
    import from LibBCast_ServiceGuide_Templates all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibCommon_DataStrings all;

```

```

=
/**
*
* @author
* ETSI CTI BCAST CON Project
* @version
* $Id$
* @desc
* This module specifies some function for the service guide.
*/

module AtsBCast_ServiceGuide_Functions {
    import from AtsBCast_TestSystem all;
    import from AtsBCast_ModuleParameters { modulepar PX_SIMULATE_TC };
    import from LibBCast_UpperTesterPrimitives_TypesAndValues all;
    import from LibBCast_ServiceGuide_Templates all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibCommon_DataStrings all;

```



## Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

import from LibCommon_BasicTypesAndValues all;
import from LibBCast_ServicePrimitives_TypesAndValues all;

group FDT {
  /**
   *
   * @desc
   *   This functions create a FDT for a
» ServiceGuideDeliveryUnit
   * @param
   *   p_SGDU_ID The Id of the ServiceGuideDeliveryUnit
   * @param
   *   p_ContentEncoding The encoding of the
   *   ServiceGuideDeliveryUnit
   * @param
   *   p_Expires The expire time how long the FDT is vaild.
   * @return
   *   The File Delivery Table Insstance
   */
  function f_createFDT4SGDU(
    AnyUri p_SGDU_Id,
    template charstring p_ContentEncoding,
    charstring p_Expires) return FDT_InstanceType {
    var UInt v_TOI := 1;
    var
» LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;

    //note: we assume for now, that all fragments are
» included in one file
    var
» LibBCast_ServicePrimitives_TypesAndValues.FileType v_File :=
      valueof( m_def_FileType(
        p_SGDU_Id,
        v_TOI,
        "application/vnd.oma.bcast.sgdu",
        p_ContentEncoding));

    v_FileList[0] := v_File;
    var FDT_InstanceType v_FDT :=
      valueof(m_def_FDT_Instance(v_FileList,
» p_Expires, omit, 0));
» FEC_OTI_FEC_Encoding_ID necessary for correct FDT creation

    return v_FDT;
  }
  /**

```

```

import from LibCommon_BasicTypesAndValues all;
import from LibBCast_ServicePrimitives_TypesAndValues all;

group FDT {
  /**
   *
   * @desc
   *   This functions create a FDT for a
» ServiceGuideDeliveryUnit
   * @param
   *   p_SGDU_ID The Id of the ServiceGuideDeliveryUnit
   * @param
   *   p_ContentEncoding The encoding of the
   *   ServiceGuideDeliveryUnit
   * @param
   *   p_Expires The expire time how long the FDT is vaild.
   * @return
   *   The File Delivery Table Insstance
   */
  function f_createFDT4SGDU(
    AnyUri p_SGDU_Id,
    template charstring p_ContentEncoding,
    charstring p_Expires) return FDT_InstanceType {
    var UInt v_TOI := 1;
    var
» LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;

    //note: we assume for now, that all fragments are
» included in one file
    var
» LibBCast_ServicePrimitives_TypesAndValues.FileType v_File :=
      valueof( m_def_FileType(
        p_SGDU_Id,
        v_TOI,
        "application/vnd.oma.bcast.sgdu",
        p_ContentEncoding));

    v_FileList[0] := v_File;
    var FDT_InstanceType v_FDT :=
      valueof(m_def_FDT_Instance(v_FileList,
» p_Expires, omit, omit));

    return v_FDT;
  }
  /**

```

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

*
* @desc
*   Creates a File Delivery Table Instance (FDT) for a
» Service
*   Guide Delivery Descriptor (SGDD).
* @param
*   p_SGDD_Id the Service Guide Delivery Descriptor id.
* @param
*   p_ContentEncoding the encoding of the SGDD
* @param
*   p_Expires the expire time how long this FDT is valid.
* @return
*   the File Delivery Table Instance (FDT)
*/
function f_createFDT4SGDD(
    AnyUri p_SGDD_Id,
    template charstring p_ContentEncoding,
    charstring p_Expires) return FDT_InstanceType {
    var UInt v_TOI := 1;
    var
» LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;

        var
» LibBCast_ServicePrimitives_TypesAndValues.FileType v_File :=
            valueof( m_def_FileType(
                p_SGDD_Id,
                v_TOI,
                "application/vnd.oma.bcast.sgdd+xml",
                p_ContentEncoding));

        v_FileList[0] := v_File;
        var FDT_InstanceType v_FDT :=
            valueof(m_def_FDT_Instance(v_FileList,
» p_Expires, omit, 0));
» FEC_OTI_FEC_Encoding_ID necessary for correct FDT creation

        return v_FDT;
    }

    // change 04 (WK 6): creates FDT instance for DVB-H
» bootstrapping
/**
*
* @desc
*   Creates a File Delivery Table Instance (FDT) for the
*   DVB-H bootstrap.
* @param

```

```

*
* @desc
*   Creates a File Delivery Table Instance (FDT) for a
» Service
*   Guide Delivery Descriptor (SGDD).
* @param
*   p_SGDD_Id the Service Guide Delivery Descriptor id.
* @param
*   p_ContentEncoding the encoding of the SGDD
* @param
*   p_Expires the expire time how long this FDT is valid.
* @return
*   the File Delivery Table Instance (FDT)
*/
function f_createFDT4SGDD(
    AnyUri p_SGDD_Id,
    template charstring p_ContentEncoding,
    charstring p_Expires) return FDT_InstanceType {
    var UInt v_TOI := 1;
    var
» LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;

        var
» LibBCast_ServicePrimitives_TypesAndValues.FileType v_File :=
            valueof( m_def_FileType(
                p_SGDD_Id,
                v_TOI,
                "application/vnd.oma.bcast.sgdd+xml",
                p_ContentEncoding));

        v_FileList[0] := v_File;
        var FDT_InstanceType v_FDT :=

```

&lt;&gt;

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

        *   p_Expires the expire time how long this FDT is vaild.
        * @return
        *   the File Delivery Table Instance (FDT)
        */
        function f_createFDT4DVBH(charstring p_Expires) return
» FDT_InstanceType {
            var UInt v_TOI := 1;
            var
» LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;

            var
» LibBCast_ServicePrimitives_TypesAndValues.FileType v_FileDiscoveryXML :=
» valueof(m_def_FileType("esg-discovery.xml", v_TOI, "text/xml", omit));

            var
» LibBCast_ServicePrimitives_TypesAndValues.FileType v_DiscoveryBIN :=
» valueof(m_def_FileType("esg-discovery.bin", v_TOI + 1,
» "application/vnd.oma.bcast.sgboot", omit));

            v_FileList[0] := v_FileDiscoveryXML;
            v_FileList[1] := v_DiscoveryBIN;

            var FDT_InstanceType v_FDT :=
» valueof(m_def_FDT_Instance(v_FileList, p_Expires, omit, 0));

```

```

        return v_FDT;
    }
}

group SGDU {
    /**
    *
    * @desc
    *   This function adds a service fragment at the last
» element of
    *   a SGDU.
    * @param
    *   p_SGDU The SGDU which shoud add the service fragment.
    * @param
    *   p_Service The service fragment to add
    */
    function f_addServiceFragment(
        inout ServiceGuideDeliveryUnit p_SGDU,
        ServiceType p_Service) runs on BCastMainComponent {

```

```

        valueof(m_def_FDT_Instance(v_FileList,
» p_Expires, omit, omit));

```

```

        return v_FDT;
    }
}

group SGDU {
    /**
    *
    * @desc
    *   This function adds a service fragment at the last
» element of
    *   a SGDU.
    * @param
    *   p_SGDU The SGDU which shoud add the service fragment.
    * @param
    *   p_Service The service fragment to add
    */
    function f_addServiceFragment(
        inout ServiceGuideDeliveryUnit p_SGDU,
        ServiceType p_Service) runs on BCastMainComponent {

```

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre> var integer v_size := sizeof (p_SGDU); p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, p_Service.version, » {      // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments » are distinguished with transport IDs Service := p_Service }}); vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs } </pre>	<>	<pre> var integer v_size := sizeof (p_SGDU); p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_Service.version, { Service := p_Service }}); </pre>
<pre> » element of * * @desc *   This function adds a content fragment at the last *   a SGDU. * @param *   p_SGDU The SGDU which should add the content fragment. * @param *   p_Content The content fragment to add */ function f_addContentFragment(   inout ServiceGuideDeliveryUnit p_SGDU,   ContentType p_Content) runs on BCastMainComponent {   var integer v_size := sizeof (p_SGDU); p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, p_Content.version, » {      // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG » fragments are distinguished with transport IDs Content := p_Content }}); vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs } </pre>	=	<pre> » element of * * @desc *   This function adds a content fragment at the last *   a SGDU. * @param *   p_SGDU The SGDU which should add the content fragment. * @param *   p_Content The content fragment to add */ function f_addContentFragment(   inout ServiceGuideDeliveryUnit p_SGDU,   ContentType p_Content) runs on BCastMainComponent {   var integer v_size := sizeof (p_SGDU); p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_Content.version, { Content := p_Content }}); </pre>
<pre> » element of * * @desc *   This function adds a schedule fragment at the last *   a SGDU. * @param *   p_SGDU The SGDU which should add the schedule fragment. * @param </pre>	=	<pre> » element of * * @desc *   This function adds a schedule fragment at the last *   a SGDU. * @param *   p_SGDU The SGDU which should add the schedule fragment. * @param </pre>

## Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre> *      p_Schedule The schedule fragment to add */ function f_addScheduleFragment(     inout ServiceGuideDeliveryUnit p_SGDU,     ScheduleType p_Schedule) runs on BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, p_Schedule.version, » {      // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments » are distinguished with transport IDs         Schedule := p_Schedule     });     vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs }  /**  *  * @desc  *      This function adds a access fragment at the last » element of a  *      SGDU.  * @param  *      p_SGDU The SGDU which should add the access fragment.  * @param  *      p_Access The access fragment to add  */ function f_addAccessFragment(     inout ServiceGuideDeliveryUnit p_SGDU,     AccessType p_Access) runs on BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, p_Access.version, { » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs         Access := p_Access     });     vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs }  /**  *  * @desc  *      This function adds a PreviewData fragment at the last </pre>		<pre> *      p_Schedule The schedule fragment to add */ function f_addScheduleFragment(     inout ServiceGuideDeliveryUnit p_SGDU,     ScheduleType p_Schedule) runs on BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_Schedule.version, {         Schedule := p_Schedule     });     +- }  /**  *  * @desc  *      This function adds a access fragment at the last » element of a  *      SGDU.  * @param  *      p_SGDU The SGDU which should add the access fragment.  * @param  *      p_Access The access fragment to add  */ function f_addAccessFragment(     inout ServiceGuideDeliveryUnit p_SGDU,     AccessType p_Access) runs on BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_Access.version, {         Access := p_Access     });     +- }  /**  *  * @desc  *      This function adds a PreviewData fragment at the last </pre>
---	--	--

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre> » element of a     *   SGDU.     * @param     *   p_SGDU The SGDU which should add the PreviewData » fragment.     * @param     *   p_PreviewData The PreviewData fragment to add     */     function f_addPreviewDataFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         PreviewDataType p_PreviewData) runs on » BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := </pre>	<pre> » element of a     *   SGDU.     * @param     *   p_SGDU The SGDU which should add the PreviewData » fragment.     * @param     *   p_PreviewData The PreviewData fragment to add     */     function f_addPreviewDataFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         PreviewDataType p_PreviewData) runs on » BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := </pre>
<pre> » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, » p_PreviewData.version, {    // change 05 (WK 6): renaming to » vc_BCASTTransportID, as SG fragments are distinguished with transport IDs     PreviewData := p_PreviewData     })); </pre>	<pre> &lt;&gt; » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_PreviewData.version, {     PreviewData := p_PreviewData     })); </pre>
<pre>     vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs } </pre>	<pre> + -     } </pre>
<pre>     /**     *     * @desc     *   This function adds a InteractivityData fragment at the » last element of a     *   SGDU.     * @param     *   p_SGDU The SGDU which should add the InteractivityData » fragment.     * @param     *   p_InteractivityData The InteractivityData fragment to » add     */     function f_addInteractivityDataFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         InteractivityDataType p_InteractivityData) runs on » BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := </pre>	<pre>     /**     *     * @desc     *   This function adds a InteractivityData fragment at the » last element of a     *   SGDU.     * @param     *   p_SGDU The SGDU which should add the InteractivityData » fragment.     * @param     *   p_InteractivityData The InteractivityData fragment to » add     */     function f_addInteractivityDataFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         InteractivityDataType p_InteractivityData) runs on » BCastMainComponent {     var integer v_size := sizeof (p_SGDU);     p_SGDU[v_size] := </pre>
<pre> » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, » p_InteractivityData.version, {    // change 05 (WK 6): renaming to » vc_BCASTTransportID, as SG fragments are distinguished with transport IDs     InteractivityData := p_InteractivityData     })); </pre>	<pre> &lt;&gt; » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_InteractivityData.version, {     InteractivityData := p_InteractivityData     })); </pre>

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre>                 }));                 vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs             }             /**              *              * @desc              * This function adds a PurchaseItem fragment at the last » element of a              * SGDU.              * @param              * p_SGDU The SGDU which should add the PurchaseItem » fragment.              * @param              * p_PurchaseItem The PurchaseItem fragment to add              */             function f_addPurchaseItemFragment(                 inout ServiceGuideDeliveryUnit p_SGDU,                 PurchaseItemType p_PurchaseItem) runs on » BCastMainComponent {                 var integer v_size := sizeof (p_SGDU);                 p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID, » p_PurchaseItem.version, { // change 05 (WK 6): renaming to » vc_BCASTTransportID, as SG fragments are distinguished with transport IDs                 PurchaseItem := p_PurchaseItem                 }));                 vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs             }             /**              *              * @desc              * This function adds a PurchaseData fragment at the last » element of a              * SGDU.              * @param              * p_SGDU The SGDU which should add the PurchaseData » fragment.              * @param              * p_PurchaseData The PurchaseData fragment to add              */             function f_addPurchaseDataFragment( </pre>	<pre>                 }));                 vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs             }             /**              *              * @desc              * This function adds a PurchaseItem fragment at the last » element of a              * SGDU.              * @param              * p_SGDU The SGDU which should add the PurchaseItem » fragment.              * @param              * p_PurchaseItem The PurchaseItem fragment to add              */             function f_addPurchaseItemFragment(                 inout ServiceGuideDeliveryUnit p_SGDU,                 PurchaseItemType p_PurchaseItem) runs on » BCastMainComponent {                 var integer v_size := sizeof (p_SGDU);                 p_SGDU[v_size] := » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_PurchaseItem.version, {                 PurchaseItem := p_PurchaseItem                 }));                 vc_BCASTTransportID := vc_BCASTTransportID + 1; » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are » distinguished with transport IDs             }             /**              *              * @desc              * This function adds a PurchaseData fragment at the last » element of a              * SGDU.              * @param              * p_SGDU The SGDU which should add the PurchaseData » fragment.              * @param              * p_PurchaseData The PurchaseData fragment to add              */             function f_addPurchaseDataFragment( </pre>
--	--

## Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre>         inout ServiceGuideDeliveryUnit p_SGDU,         PurchaseDataType p_PurchaseData) runs on     » BCastMainComponent {         var integer v_size := sizeof (p_SGDU);         p_SGDU[v_size] :=     » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID,     » p_PurchaseData.version, {      // change 05 (WK 6): renaming to     » vc_BCASTTransportID, as SG fragments are distinguished with transport IDs         PurchaseData := p_PurchaseData     }));     » vc_BCASTTransportID := vc_BCASTTransportID + 1;     » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are     » distinguished with transport IDs     }      /**      *      * @desc      * This function adds a PurchaseChannel fragment at the     » last element of a      * SGDU.      * @param      * p_SGDU The SGDU which should add the PurchaseChannel     » fragment.      * @param      * p_PurchaseChannel The PurchaseChannel fragment to add      */     function f_addPurchaseChannelFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         PurchaseChannelType p_PurchaseChannel) runs on     » BCastMainComponent {         var integer v_size := sizeof (p_SGDU);         p_SGDU[v_size] :=     » valueof(m_xml_ServiceGuideFragment(vc_BCASTTransportID,     » p_PurchaseChannel.version, {    // change 05 (WK 6): renaming to     » vc_BCASTTransportID, as SG fragments are distinguished with transport IDs         PurchaseChannel := p_PurchaseChannel     }));     » vc_BCASTTransportID := vc_BCASTTransportID + 1;     » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are     » distinguished with transport IDs     }      /**      *      * @desc      * This function adds a sdp fragment at the last element </pre>		<pre>         inout ServiceGuideDeliveryUnit p_SGDU,         PurchaseDataType p_PurchaseData) runs on     » BCastMainComponent {         var integer v_size := sizeof (p_SGDU);         p_SGDU[v_size] :=     » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_PurchaseData.version, {         PurchaseData := p_PurchaseData     }));     » vc_BCASTTransportID := vc_BCASTTransportID + 1;     » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are     » distinguished with transport IDs     }      /**      *      * @desc      * This function adds a PurchaseChannel fragment at the     » last element of a      * SGDU.      * @param      * p_SGDU The SGDU which should add the PurchaseChannel     » fragment.      * @param      * p_PurchaseChannel The PurchaseChannel fragment to add      */     function f_addPurchaseChannelFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         PurchaseChannelType p_PurchaseChannel) runs on     » BCastMainComponent {         var integer v_size := sizeof (p_SGDU);         p_SGDU[v_size] :=     » valueof(m_xml_ServiceGuideFragment(vc_TOI, p_PurchaseChannel.version, {         PurchaseChannel := p_PurchaseChannel     }));     » vc_BCASTTransportID := vc_BCASTTransportID + 1;     » // change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are     » distinguished with transport IDs     }      /**      *      * @desc      * This function adds a sdp fragment at the last element </pre>
	<>	
	=	
	+ -	
	=	
	<>	
	=	
	+ -	
	=	



Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre> » of a     *   SGDU.     * @param     *   p_SGDU The SGDU which should add the sdp fragment.     * @param     *   p_version The fragment version     * @param     *   p_SDP The sdp fragment to add     */     function f_addSdpFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         UInt32 p_version,         SdpFragment p_SDP) runs on BCastMainComponent {         var integer v_size := sizeof (p_SGDU);         p_SGDU[v_size] := </pre>		<pre> » of a     *   SGDU.     * @param     *   p_SGDU The SGDU which should add the sdp fragment.     * @param     *   p_version The fragment version     * @param     *   p_SDP The sdp fragment to add     */     function f_addSdpFragment(         inout ServiceGuideDeliveryUnit p_SGDU,         UInt32 p_version,         SdpFragment p_SDP) runs on BCastMainComponent {         var integer v_size := sizeof (p_SGDU);         p_SGDU[v_size] := </pre>
<pre>         valueof(m_sdp_ServiceGuideFragment(vc_BCASTTr » ansportID, p_version, p_SDP)); // change 05 (WK » 6): renaming to vc_BCASTTransportID, as SG fragments are distinguished with » transport IDs         vc_BCASTTransportID := vc_BCASTTransportID + » 1; // change 05 (WK » 6): renaming to vc_BCASTTransportID, as SG fragments are distinguished with » transport IDs </pre>	<>	<pre>         valueof(m_sdp_ServiceGuideFragment(vc_TOI, » p_version, p_SDP)); </pre>
<pre>     } }  group SGDD {     /**     *     * @desc     *   This function creates a SGDD fragment from a     *   ServiceGuideFragment.     * @param     *   p_ServiceGuideFragment The ServiceGuideFragment which     *   includes the informations about the Fragment     * @param     *   p_SGDU_Id The SGDU id     * @return     *   The SGDD Fragment     */     function f_createSGDDFragment(         ServiceGuideFragment p_ServiceGuideFragment,         charstring p_SGDU_Id) return FragmentType {         var ServiceGuideFragmentType v_Type :=             p_ServiceGuideFragment.fragmentType;         var FragmentType v_Fragment; </pre>	=	<pre>     } }  group SGDD {     /**     *     * @desc     *   This function creates a SGDD fragment from a     *   ServiceGuideFragment.     * @param     *   p_ServiceGuideFragment The ServiceGuideFragment which     *   includes the informations about the Fragment     * @param     *   p_SGDU_Id The SGDU id     * @return     *   The SGDD Fragment     */     function f_createSGDDFragment(         ServiceGuideFragment p_ServiceGuideFragment,         charstring p_SGDU_Id) return FragmentType {         var ServiceGuideFragmentType v_Type :=             p_ServiceGuideFragment.fragmentType;         var FragmentType v_Fragment; </pre>

## Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre>         if (ischosen (v_Type.xmlFragment)) {             var XmlFragment v_xml := v_Type.xmlFragment;             var UInt8 v_FragmentType := » f_getXmlFragmentType(v_xml);              v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>		<pre>         if (ischosen (v_Type.xmlFragment)) {             var XmlFragment v_xml := v_Type.xmlFragment;             var UInt8 v_FragmentType :=              v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>
<pre> » p_ServiceGuideFragment.fragmentVersion,                // change » 06 (WK 6): dynamic setting of fragment version number</pre>	+-	
<pre>             f_getXmlFragmentId(v_xml),             c_XML_BCAST_SG_fragment,             v_FragmentType));          } else if (ischosen (v_Type.sdpFragment)) {             v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>	=	<pre>             f_getXmlFragmentId(v_xml),             c_XML_BCAST_SG_fragment,             v_FragmentType));          } else if (ischosen (v_Type.sdpFragment)) {             v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>
<pre> » p_ServiceGuideFragment.fragmentVersion,                // change » 06 (WK 6): dynamic setting of fragment version number</pre>	+-	
<pre> » v_Type.sdpFragment.fragmentId,              c_SDP_fragment,             omit));          } else if (ischosen (v_Type.mbmsUsbdFragment)) {             v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>	=	<pre> » v_Type.sdpFragment.fragmentId,              c_SDP_fragment,             omit));          } else if (ischosen (v_Type.mbmsUsbdFragment)) {             v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>
<pre> » p_ServiceGuideFragment.fragmentVersion,                // change » 06 (WK 6): dynamic setting of fragment version number</pre>	+-	
<pre> » v_Type.mbmsUsbdFragment.fragmentId,              c_MBMS_USD_fragment,             omit));          } else if (ischosen (v_Type.adpFragment)) {             v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>	=	<pre> » v_Type.mbmsUsbdFragment.fragmentId,              c_MBMS_USD_fragment,             omit));          } else if (ischosen (v_Type.adpFragment)) {             v_Fragment :=                 valueof(m_def_Fragment(  » p_ServiceGuideFragment.fragmentTransportId,</pre>
<pre> » p_ServiceGuideFragment.fragmentVersion,                // change</pre>	+-	

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

» 06 (WK 6): dynamic setting of fragment version number

```

» v_Type.adpFragment.fragmentId,
                                c_XML_ADP_fragment,
                                omit));
    }

    return v_Fragment;
}

/**
 *
 * @desc
 *   this functions returns the id of a xml fragment
 * @param
 *   p_XmlFragment The xml fragment
 * @return
 *   The id of the xml fragment
 */
function f_getXmlFragmentId(XmlFragment p_XmlFragment) return
» charstring {
    var charstring v_id;

    if (ischosen (p_XmlFragment.Service)) {
        v_id := p_XmlFragment.Service.id;
    } else if (ischosen (p_XmlFragment.Content)) {
        v_id := p_XmlFragment.Content.id;
    } else if (ischosen (p_XmlFragment.Schedule)) {
        v_id := p_XmlFragment.Schedule.id;
    } else if (ischosen (p_XmlFragment.Access)) {
        v_id := p_XmlFragment.Access.id;
    } else if (ischosen (p_XmlFragment.PreviewData)) {
        v_id := p_XmlFragment.PreviewData.id;
    } else if (ischosen (p_XmlFragment.PurchaseItem)) {
        v_id := p_XmlFragment.PurchaseItem.id;
    } else if (ischosen (p_XmlFragment.PurchaseData)) {
        v_id := p_XmlFragment.PurchaseData.id;
    } else if (ischosen (p_XmlFragment.PurchaseChannel))
» {
        v_id := p_XmlFragment.PurchaseChannel.id;
    } else if (ischosen
» (p_XmlFragment.InteractivityData)) {
        v_id := p_XmlFragment.InteractivityData.id;
    }

    return v_id;
}

```

=

```

» v_Type.adpFragment.fragmentId,
                                c_XML_ADP_fragment,
                                omit));
    }

    return v_Fragment;
}

/**
 *
 * @desc
 *   this functions returns the id of a xml fragment
 * @param
 *   p_XmlFragment The xml fragment
 * @return
 *   The id of the xml fragment
 */
function f_getXmlFragmentId(XmlFragment p_XmlFragment) return
» charstring {
    var charstring v_id;

    if (ischosen (p_XmlFragment.Service)) {
        v_id := p_XmlFragment.Service.id;
    } else if (ischosen (p_XmlFragment.Content)) {
        v_id := p_XmlFragment.Content.id;
    } else if (ischosen (p_XmlFragment.Schedule)) {
        v_id := p_XmlFragment.Schedule.id;
    } else if (ischosen (p_XmlFragment.Access)) {
        v_id := p_XmlFragment.Access.id;
    } else if (ischosen (p_XmlFragment.PreviewData)) {
        v_id := p_XmlFragment.PreviewData.id;
    } else if (ischosen (p_XmlFragment.PurchaseItem)) {
        v_id := p_XmlFragment.PurchaseItem.id;
    } else if (ischosen (p_XmlFragment.PurchaseData)) {
        v_id := p_XmlFragment.PurchaseData.id;
    } else if (ischosen (p_XmlFragment.PurchaseChannel))
» {
        v_id := p_XmlFragment.PurchaseChannel.id;
    } else if (ischosen
» (p_XmlFragment.InteractivityData)) {
        v_id := p_XmlFragment.InteractivityData.id;
    }

    return v_id;
}

```

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

    /**
     *
     * @desc
     *   This function returns the fragment type for a xml
» fragment.
     * @param
     *   p_XmlFragment The xml fragment.
     * @return
     *   the fragment type of the xml fragment
     */
    function f_getXmlFragmentType(XmlFragment p_XmlFragment)
» return UInt8 {
        var UInt8 v_FragmentType;

        if (ischosen (p_XmlFragment.Service)) {
            v_FragmentType := c_SG_Service_Fragment;
        } else if (ischosen (p_XmlFragment.Content)) {
            v_FragmentType := c_SG_Content_Fragment;
        } else if (ischosen (p_XmlFragment.Schedule)) {
            v_FragmentType := c_SG_Schedule_Fragment;
        } else if (ischosen (p_XmlFragment.Access)) {
            v_FragmentType := c_SG_Access_Fragment;
        } else if (ischosen (p_XmlFragment.PreviewData)) {
            v_FragmentType := c_SG_PreviewData_Fragment;
        } else if (ischosen (p_XmlFragment.PurchaseItem)) {
            v_FragmentType := c_SG_PurchaseItem_Fragment;
        } else if (ischosen (p_XmlFragment.PurchaseData)) {
            v_FragmentType := c_SG_PurchaseData_Fragment;
        } else if (ischosen (p_XmlFragment.PurchaseChannel))
» {
            v_FragmentType :=
» c_SG_PurchaseChannel_Fragment;
        } else if (ischosen
» (p_XmlFragment.InteractivityData)) {
            v_FragmentType :=
» c_SG_InteractivityData_Fragment;
        }

        return v_FragmentType;
    }

    /**
     *
     * @desc
     *   This function create a SGDD for a SGDU.
     * @param

```

```

    /**
     *
     * @desc
     *   This function returns the fragment type for a xml
» fragment.
     * @param
     *   p_XmlFragment The xml fragment.
     * @return
     *   the fragment type of the xml fragment
     */
    function f_getXmlFragmentType(XmlFragment p_XmlFragment)
» return UInt8 {
        var UInt8 v_FragmentType;

        if (ischosen (p_XmlFragment.Service)) {
            v_FragmentType := c_SG_Service_Fragment;
        } else if (ischosen (p_XmlFragment.Content)) {
            v_FragmentType := c_SG_Content_Fragment;
        } else if (ischosen (p_XmlFragment.Schedule)) {
            v_FragmentType := c_SG_Schedule_Fragment;
        } else if (ischosen (p_XmlFragment.Access)) {
            v_FragmentType := c_SG_Access_Fragment;
        } else if (ischosen (p_XmlFragment.PreviewData)) {
            v_FragmentType := c_SG_PreviewData_Fragment;
        } else if (ischosen (p_XmlFragment.PurchaseItem)) {
            v_FragmentType := c_SG_PurchaseItem_Fragment;
        } else if (ischosen (p_XmlFragment.PurchaseData)) {
            v_FragmentType := c_SG_PurchaseData_Fragment;
        } else if (ischosen (p_XmlFragment.PurchaseChannel))
» {
            v_FragmentType :=
» c_SG_PurchaseChannel_Fragment;
        } else if (ischosen
» (p_XmlFragment.InteractivityData)) {
            v_FragmentType :=
» c_SG_InteractivityData_Fragment;
        }

        return v_FragmentType;
    }

    /**
     *
     * @desc
     *   This function create a SGDD for a SGDU.
     * @param

```

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

<pre> *   p_SGDU The SGDU * @param *   p_SGDU_ID The id of the SGDU * @param *   p_SGDD_ID The id of the SGDD * @return *   The SGDD */ function f_createSGDD4SGDU(     ServiceGuideDeliveryUnit p_SGDU,     AnyUri p_SGDU_ID,     AnyUri p_SGDD_ID,     UInt p_startTime,     UInt p_endTime) return ServiceGuideDeliveryDescriptorType {     // change 01 (WK 6): global time definition     time definition } </pre>		<pre> *   p_SGDU The SGDU * @param *   p_SGDU_ID The id of the SGDU * @param *   p_SGDD_ID The id of the SGDD * @return *   The SGDD */ function f_createSGDD4SGDU(     ServiceGuideDeliveryUnit p_SGDU,     AnyUri p_SGDU_ID,     AnyUri p_SGDD_ID) return ServiceGuideDeliveryDescriptorType { } </pre>
<pre> » // change 01 (WK 6): global time definition » ServiceGuideDeliveryDescriptorType { » time definition </pre>	<>	<pre> » ServiceGuideDeliveryDescriptorType { </pre>
<pre> var FragmentList v_list; var FragmentType v_Fragment; var integer v_SGDU_idx;  // go through the ServiceGuideDeliveryUnit to create » the FragmentList for (var integer i := 0; i &lt; sizeof (p_SGDU); i := i » + 1) {     var ServiceGuideFragment » v_ServiceGuideFragment := p_SGDU[i];     v_Fragment := » f_createSGDDFragment(v_ServiceGuideFragment, p_SGDU_ID);     v_list[i] := v_Fragment; }  var ServiceGuideDeliveryUnitType v_SGDU_field := » valueof(m_def_ServiceGuideDeliveryUnitField(1, // TOI     p_SGDU_ID, // location     v_list)); // fragment list var ServiceGuideDeliveryDescriptorType v_SGDD := </pre>	=	<pre> var FragmentList v_list; var FragmentType v_Fragment; var integer v_SGDU_idx;  // go through the ServiceGuideDeliveryUnit to create » the FragmentList for (var integer i := 0; i &lt; sizeof (p_SGDU); i := i » + 1) {     var ServiceGuideFragment » v_ServiceGuideFragment := p_SGDU[i];     v_Fragment := » f_createSGDDFragment(v_ServiceGuideFragment, p_SGDU_ID);     v_list[i] := v_Fragment; }  var ServiceGuideDeliveryUnitType v_SGDU_field := » valueof(m_def_ServiceGuideDeliveryUnitField(1, // TOI     p_SGDU_ID, // location     v_list)); // fragment list var ServiceGuideDeliveryDescriptorType v_SGDD := </pre>
<pre> » valueof(m_def_ServiceGuideDeliveryDescriptor(p_SGDD_ID, v_SGDU_field, » tTime, p_endTime)); // change 01 (WK 6): global time definition </pre>	<>	<pre> » valueof(m_def_ServiceGuideDeliveryDescriptor(p_SGDD_ID, v_SGDU_field)); </pre>
<pre> return v_SGDD; } </pre>	=	<pre> return v_SGDD; } </pre>

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

group misc {
    /**
     *
     * @desc
     *   This function increase the fragment version.
     * @param
     *   p_version the fragment version to increase
     * @return
     *   The increased fragment version
     */
    function f_IncFragmentVersion(in UInt32 p_version) return
» UInt32 {
        var UInt32 v_rc := p_version;
        if (v_rc < c_uInt32Max and v_rc >= 0) {
            v_rc := v_rc + 1;
        } else {
            log ("It is not possible to increase the
» Fragment Version");
        }
        return v_rc;
    }
}

```

```

group misc {
    /**
     *
     * @desc
     *   This function increase the fragment version.
     * @param
     *   p_version the fragment version to increase
     * @return
     *   The increased fragment version
     */
    function f_IncFragmentVersion(in UInt32 p_version) return
» UInt32 {
        var UInt32 v_rc := p_version;
        if (v_rc < c_uInt32Max and v_rc >= 0) {
            v_rc := v_rc + 1;
        } else {
            log ("It is not possible to increase the
» Fragment Version");
        }
        return v_rc;
    }
}

```

&lt;&gt;

```

    /**
     *
     * @desc
     *   initialize the start and end time. The start time will
» be
     *   set to a value which is one hour after test execution
» and the
     *   endtime will be set to a value one our after the start
» time.
     * @param
     *   startTime the start time to be initialize
     * @param
     *   endTime the end time to be initialize
     */
    function f_InitStartEndTime(inout UInt p_startTime, inout
» UInt p_endTime) {
        p_startTime := f_getNTPTime() + c_one_hour;
        p_endTime := p_startTime + c_one_hour;
    }
}

```

```

    /**
     *
     * @desc

```

=

```

    /**
     *
     * @desc

```

## Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

        * This function returns the access tpye for a specific
        * broadcast network bearer.
        * @param
        * p_type The BroadcastBearerType enumeration which
» contains the
        * access type.
        * @return
        * The access type used in access fragment
        */
        function f_getAccessType(BroadcastBearerType p_type) return
» UInt8 {
            var UInt8 v_rc;

            if (p_type == e_dvb_h) {
                v_rc := c_ipdcOverDvbH_bcType;
            } else if (p_type == e_mbms) {
                v_rc := c_3gppMbms_bcType;
            } else if (p_type == e_bmcms) {
                v_rc := c_3gpp2Bcmcs_bcType;
            }

            return v_rc;
        }

        /**
        *
        * @desc This external function returns the 32 Bits integer
» part of the current NTP timestamp.
        * @return The current NTP timestamp
        */
        external function fx_getNTPTime() return UInt;

        /**
        *
        * @desc This external function returns the DateTime
» representation for a NTP timestamp.
        * @param p_NTPTime The NTP timestamp which should be
» converted to the DateTime format.
        * @return The DateTime representation of the NTP timestamp
        */
        external function fx_getDateTime(UInt p_NTPTime) return
» DateTime;

        /**
        *
        * @desc

```

```

        * This function returns the access tpye for a specific
        * broadcast network bearer.
        * @param
        * p_type The BroadcastBearerType enumeration which
» contains the
        * access type.
        * @return
        * The access type used in access fragment
        */
        function f_getAccessType(BroadcastBearerType p_type) return
» UInt8 {
            var UInt8 v_rc;

            if (p_type == e_dvb_h) {
                v_rc := c_ipdcOverDvbH_bcType;
            } else if (p_type == e_mbms) {
                v_rc := c_3gppMbms_bcType;
            } else if (p_type == e_bmcms) {
                v_rc := c_3gpp2Bcmcs_bcType;
            }

            return v_rc;
        }

        /**
        *
        * @desc This external function returns the 32 Bits integer
» part of the current NTP timestamp.
        * @return The current NTP timestamp
        */
        external function fx_getNTPTime() return UInt;

        /**
        *
        * @desc This external function returns the DateTime
» representation for a NTP timestamp.
        * @param p_NTPTime The NTP timestamp which should be
» converted to the DateTime format.
        * @return The DateTime representation of the NTP timestamp
        */
        external function fx_getDateTime(UInt p_NTPTime) return
» DateTime;

        /**
        *
        * @desc

```

Datei: AtsBCast\_ServiceGuide\_Functions.ttcn (Fortsetzung)

```

    * This function returns the current NTP time as integer.
    * @return
    * The current NTP time.
    */
function f_getNTPTime() return UInt {
    if(PX_SIMULATE_TC) {
        return 0;
    }

    return fx_getNTPTime();
}

/**
 *
 * @desc
 * This function returns the DateTime for a NTP time.
 * @param
 * p_NTPTime The NTP time which should be converted
 * @return
 * The DateTime
 */
function f_getDateTime(UInt p_NTPTime) return DateTime {
    if(PX_SIMULATE_TC) {
        return "Day:Month:Year hh:mm:ss";
    }

    return fx_getDateTime(p_NTPTime);
}
}

```

```

    * This function returns the current NTP time as integer.
    * @return
    * The current NTP time.
    */
function f_getNTPTime() return UInt {
    if(PX_SIMULATE_TC) {
        return 0;
    }

    return fx_getNTPTime();
}

/**
 *
 * @desc
 * This function returns the DateTime for a NTP time.
 * @param
 * p_NTPTime The NTP time which should be converted
 * @return
 * The DateTime
 */
function f_getDateTime(UInt p_NTPTime) return DateTime {
    if(PX_SIMULATE_TC) {
        return "Day:Month:Year hh:mm:ss";
    }

    return fx_getDateTime(p_NTPTime);
}
}

```

Datei: AtsBCast\_ServiceGuideTests.ttcn

```

/**
 *
 * @author
 * ETSI CTI BCAST CON Project
 * @version
 * $Id$
 * @desc
 * This module specifies pilot tests for BCast service guide.
 */
module AtsBCast_ServiceGuideTests {
    import from LibBCast_Interface all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
}

```

```

=
/**
 *
 * @author
 * ETSI CTI BCAST CON Project
 * @version
 * $Id$
 * @desc
 * This module specifies pilot tests for BCast service guide.
 */
module AtsBCast_ServiceGuideTests {
    import from LibBCast_Interface all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
}

```



## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> import from LibBCast_UpperTesterPrimitives_TypesAndValues all; import from LibBCast_Common_TypesAndValues all; import from LibCommon_BasicTypesAndValues all; import from AtsBCast_ModuleParameters all; import from LibBCast_ModuleParameters all; import from AtsBCast_TestConfiguration_Functions all; import from AtsBCast_TestSystem all;  group bcastConformanceTestCases {     group clientConformanceTestCases {         /**          *          * @desc          * &lt;p&gt;Service Guide update via broadcast channel » (same fragment id, higher version number)&lt;/p&gt;          * &lt;p&gt;Test Case Description&lt;/p&gt;          * &lt;p&gt;with a terminal having received a Service » Guide and listening to the broadcast Service Guide Delivery Channel.&lt;/p&gt;          * &lt;p&gt;when the terminal receives Service Guide » update containing a content fragment with same id and higher version&lt;/p&gt;          * &lt;p&gt;then the terminal presents the information » of the updated Service Guide&lt;/p&gt;          * &lt;p&gt;Specification Reference&lt;/p&gt;          * &lt;p&gt;[BCAST10-ESG] Section 5.4.2.1.2, 5.5 3rd » bullet item&lt;/p&gt;          * &lt;p&gt;Preconditions&lt;/p&gt;          * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;          * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;          * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;          * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:&lt;/p&gt;          * &lt;p&gt;Initial service guide:          * &lt;ul&gt;          * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;          * &lt;li&gt;Content fragment with version = "1" » and Name = "Programm1" , and StartTime and EndTime elements indicating » values after the time of test&lt;/li&gt;          * &lt;li&gt;Schedule fragment for "Programm1" » content fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt;          * &lt;li&gt;Access fragment "Programm1" » schedule fragment&lt;/li&gt;          * &lt;/ul&gt; </pre>	<pre> import from LibBCast_UpperTesterPrimitives_TypesAndValues all; import from LibBCast_Common_TypesAndValues all; import from LibCommon_BasicTypesAndValues all; import from AtsBCast_ModuleParameters all; import from LibBCast_ModuleParameters all; import from AtsBCast_TestConfiguration_Functions all; import from AtsBCast_TestSystem all;  group bcastConformanceTestCases {     group clientConformanceTestCases {         /**          *          * @desc          * &lt;p&gt;Service Guide update via broadcast channel » (same fragment id, higher version number)&lt;/p&gt;          * &lt;p&gt;Test Case Description&lt;/p&gt;          * &lt;p&gt;with a terminal having received a Service » Guide and listening to the broadcast Service Guide Delivery Channel.&lt;/p&gt;          * &lt;p&gt;when the terminal receives Service Guide » update containing a content fragment with same id and higher version&lt;/p&gt;          * &lt;p&gt;then the terminal presents the information » of the updated Service Guide&lt;/p&gt;          * &lt;p&gt;Specification Reference&lt;/p&gt;          * &lt;p&gt;[BCAST10-ESG] Section 5.4.2.1.2, 5.5 3rd » bullet item&lt;/p&gt;          * &lt;p&gt;Preconditions&lt;/p&gt;          * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;          * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;          * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;          * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:&lt;/p&gt;          * &lt;p&gt;Initial service guide:          * &lt;ul&gt;          * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;          * &lt;li&gt;Content fragment with version = "1" » and Name = "Programm1" , and StartTime and EndTime elements indicating » values after the time of test&lt;/li&gt;          * &lt;li&gt;Schedule fragment for "Programm1" » content fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt;          * &lt;li&gt;Access fragment "Programm1" » schedule fragment&lt;/li&gt;          * &lt;/ul&gt; </pre>
--	--

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

*      </p>
*      <p>Updated service guide:
*          <ul>
*              <li>Service fragment with
» Name="TvChannel"</li>
*              <li>Content fragment with version = "2"
» and Name = "Programm1-version2" , and StartTime and EndTime elements
» indicating values after the time of test</li>
*              <li>Schedule fragment for
» "Programm1-version2" content fragment with same values for startTime and
» endTime in the presentationWindow element</li>
*              <li>Access fragment for
» "Programm1-version2" schedule fragment</li>
*          </ul>
*          Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide delivery.
*      </p>
*      <p>Test Procedure</p>
*      <p>
*          <ul>
*              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*              <li>Activate the BCAST application of
» the terminal</li>
*              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*              <li>Browse the SG on the terminal</li>
*              <li>Set up the test tool to produce
» updated BCAST service guide announcement and delivery using broadcast
» channel.</li>
*              <li>Request from BCAST application on
» terminal to update the service guide (optional).</li>
*              <li>Browse the SG on the terminal</li>
*          </ul>
*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the initial delivery of the SG
*          <ul>
*              <li>There is a service "TvChannel"
» associated with a program "Programm1" as well as start and end time values
» (There is a "TvChannel" that contains "Programm1" scheduled from startTime
» to endTime)</li>
*          </ul>
*      </p>
*      <p>The following should be visible to the end

```

```

*      </p>
*      <p>Updated service guide:
*          <ul>
*              <li>Service fragment with
» Name="TvChannel"</li>
*              <li>Content fragment with version = "2"
» and Name = "Programm1-version2" , and StartTime and EndTime elements
» indicating values after the time of test</li>
*              <li>Schedule fragment for
» "Programm1-version2" content fragment with same values for startTime and
» endTime in the presentationWindow element</li>
*              <li>Access fragment for
» "Programm1-version2" schedule fragment</li>
*          </ul>
*          Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide delivery.
*      </p>
*      <p>Test Procedure</p>
*      <p>
*          <ul>
*              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*              <li>Activate the BCAST application of
» the terminal</li>
*              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*              <li>Browse the SG on the terminal</li>
*              <li>Set up the test tool to produce
» updated BCAST service guide announcement and delivery using broadcast
» channel.</li>
*              <li>Request from BCAST application on
» terminal to update the service guide (optional).</li>
*              <li>Browse the SG on the terminal</li>
*          </ul>
*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the initial delivery of the SG
*          <ul>
*              <li>There is a service "TvChannel"
» associated with a program "Programm1" as well as start and end time values
» (There is a "TvChannel" that contains "Programm1" scheduled from startTime
» to endTime)</li>
*          </ul>
*      </p>
*      <p>The following should be visible to the end

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» user after the delivery of the update of the SG
    *      <ul>
    *          <li>There is a service "TvChannel"
» associated with a program "Programm1-version2" as well as start and end
» time values (There is a "TvChannel" that contains "Programm1-version2"
» scheduled from startTime to endTime)</li>
    *      </ul>
    *      </p>
    *      @verdict
    *          pass in case the client pass the conformance
» test.
    *      @verdict
    *          fail in case the client does not pass the
» conformance test.
    *      @verdict
    *          inconc in case a guard timer expires.
    */
    testcase TC_BCAST_ESG_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();

    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_FragmentVersion := 1;

```

```

» _hour);
    f_main_ctrl_InitStartEndTime(c_one_hour,c_one
    // change 01 (WK 6): global time definition

```

```

    var DateTime v_DateTime_StartTime :=
» f_getDateTime(v_startTime);
    // change 01 (WK 6): global time
» definition
    var DateTime v_DateTime_EndTime :=
» f_getDateTime(v_endTime);
    // change 01 (WK 6): global time
» definition

```

```

    var SdpFragment v_Sdp :=
» valueof(m_def_SdpFragment(
    PX_SDP_VIDEO_PROG_1_REF_ID,
    PX_SDP_VIDEO_PROG_1

```

```

» user after the delivery of the update of the SG
    *      <ul>
    *          <li>There is a service "TvChannel"
» associated with a program "Programm1-version2" as well as start and end
» time values (There is a "TvChannel" that contains "Programm1-version2"
» scheduled from startTime to endTime)</li>
    *      </ul>
    *      </p>
    *      @verdict
    *          pass in case the client pass the conformance
» test.
    *      @verdict
    *          fail in case the client does not pass the
» conformance test.
    *      @verdict
    *          inconc in case a guard timer expires.
    */
    testcase TC_BCAST_ESG_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();

    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_FragmentVersion := 1;

```

```

    var UInt v_NTP_StartTime := 0;

    var UInt v_NTP_EndTime := 0;
    f_InitStartEndTime(v_NTP_StartTime,
» v_NTP_EndTime);

```

```

    var DateTime v_DateTime_StartTime :=
» f_getDateTime(v_NTP_StartTime);
    var DateTime v_DateTime_EndTime :=
» f_getDateTime(v_NTP_EndTime);

```

```

    var SdpFragment v_Sdp :=
» valueof(m_def_SdpFragment(
    PX_SDP_VIDEO_PROG_1_REF_ID,
    PX_SDP_VIDEO_PROG_1

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre>                 ));                  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),                 PX_SDP_VIDEO_PROG_1_REF_ID                 ));                  var ServiceType v_Service := » valueof(m_def_Service(                 PX_SGDU_SERVICE_ID,                 v_FragmentVersion,                 "TvChannel",                 c_basicTv_ServiceType                 ));                  var ContentType v_Content := » valueof(m_def_Content(                 PX_SGDU_CONTENT_ID_PROG_1,                 v_FragmentVersion,                 "Programmel",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime                 ));                  var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_startTime,                 v_endTime » // change 01 (WK 6): global time definition » // change 01 (WK 6): global time definition </pre>		<pre>                 ));                  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),                 PX_SDP_VIDEO_PROG_1_REF_ID                 ));                  var ServiceType v_Service := » valueof(m_def_Service(                 PX_SGDU_SERVICE_ID,                 v_FragmentVersion,                 "TvChannel",                 c_basicTv_ServiceType                 ));                  var ContentType v_Content := » valueof(m_def_Content(                 PX_SGDU_CONTENT_ID_PROG_1,                 v_FragmentVersion,                 "Programmel",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime                 ));                  var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_NTP_StartTime,                 v_NTP_EndTime </pre>
<pre>                 ));                  var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG </pre>	=	<pre>                 ));                  var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre>         ));          f_addServiceFragment(v_SGDU, v_Service);         f_addContentFragment(v_SGDU, v_Content);         f_addScheduleFragment(v_SGDU, v_Schedule);         f_addAccessFragment(v_SGDU, v_Access);         f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);          f_main_ut_RunBCastApplication();         if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);         }         f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);          f_main_ut_CheckService("TvChannel");         f_main_ut_SelectService("TvChannel");         f_main_ut_CheckContent("Programmel", » v_DateTime_StartTime, v_DateTime_EndTime);          // test behavior </pre>		<pre>         ));          f_addServiceFragment(v_SGDU, v_Service);         f_addContentFragment(v_SGDU, v_Content);         f_addScheduleFragment(v_SGDU, v_Schedule);         f_addAccessFragment(v_SGDU, v_Access);         f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);          f_main_ut_RunBCastApplication();         if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);         }         f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);          f_main_ut_CheckService("TvChannel");         f_main_ut_SelectService("TvChannel");         f_main_ut_CheckContent("Programmel", » v_DateTime_StartTime, v_DateTime_EndTime);          // test behavior </pre>
<pre>         // reset TOI         vc_BCASTTransportID := 1; » // change 07 (WK 6): in order to send same TransportID </pre>	+ -	
<pre>         v_SGDU := {};         v_Content.version := » f_IncFragmentVersion(v_Content.version);         v_Content.Names := » {{omit,"Programmel-version2"}}; </pre>	=	<pre>         v_SGDU := {};         v_Content.version := » f_IncFragmentVersion(v_Content.version);         v_Content.Names := » {{omit,"Programmel-version2"}}; </pre>
<pre>         f_addServiceFragment(v_SGDU, v_Service); » // change 08 (WK 6): SGDU must contain all fragments         f_addContentFragment(v_SGDU, v_Content);         f_addScheduleFragment(v_SGDU, v_Schedule); » // change 08 (WK 6): SGDU must contain all fragments         f_addAccessFragment(v_SGDU, v_Access); » // change 08 (WK 6): SGDU must contain all fragments         f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);         // change 08 (WK 6): SGDU must contain all fragments </pre>	<>	<pre>         f_addContentFragment(v_SGDU, v_Content); </pre>
<pre>         f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);          if (PX_GET_SG_UPDATE_MANUALLY) {             f_main_ut_UpdateServiceGuide();         } </pre>	=	<pre>         f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);          if (PX_GET_SG_UPDATE_MANUALLY) {             f_main_ut_UpdateServiceGuide();         } </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> f_main_ut_CheckService("TvChannel"); f_main_ut_SelectService("TvChannel"); f_main_ut_CheckContent("Programm1-version2", » v_DateTime_StartTime, v_DateTime_EndTime);  // postamble f_main_ut_PowerOff();  f_cf_Broadcast_down(); f_cf_UpperTester_down(); } // end test case TC_BCAST_ESG_CONF_101  /**  *  * @desc  * &lt;p&gt;Service Guide update via broadcast channel » (same fragment id, lower version number)&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt;  * &lt;p&gt;with a terminal having received a Service » Guide and listening to the broadcast Service Guide Delivery Channel.&lt;/p&gt;  * &lt;p&gt;when the terminal receives Service Guide » update containing a content fragment with same id and lower version&lt;/p&gt;  * &lt;p&gt;then the terminal does not present the » information of the updated Service Guide&lt;/p&gt;  * &lt;p&gt;Specification Reference&lt;/p&gt;  * &lt;p&gt;[BCAST10-ESG] Section 5.4.2.1.2, 5.5 4th » bullet item&lt;/p&gt; </pre>		<pre> f_main_ut_CheckService("TvChannel"); f_main_ut_SelectService("TvChannel"); f_main_ut_CheckContent("Programm1-version2", » v_DateTime_StartTime, v_DateTime_EndTime);  // postamble f_main_ut_PowerOff();  f_cf_Broadcast_down(); f_cf_UpperTester_down(); } // end test case TC_BCAST_ESG_CONF_101  /**  *  * @desc  * &lt;p&gt;Service Guide update via broadcast channel » (same fragment id, lower version number)&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt;  * &lt;p&gt;with a terminal having received a Service » Guide and listening to the broadcast Service Guide Delivery Channel.&lt;/p&gt;  * &lt;p&gt;when the terminal receives Service Guide » update containing a content fragment with same id and lower version&lt;/p&gt;  * &lt;p&gt;then the terminal does not present the » information of the updated Service Guide&lt;/p&gt;  * &lt;p&gt;Specification Reference&lt;/p&gt;  * &lt;p&gt;[BCAST10 -ESG] Section 5.4.2.1.2, 5.5 4th » bullet item&lt;/p&gt; </pre>
<pre>  * &lt;p&gt;Preconditions&lt;/p&gt;  * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;  * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;  * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;  * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:&lt;/p&gt;  * &lt;p&gt;Initial service guide:  * &lt;ul&gt;  * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;  * &lt;li&gt;Content fragment with version = "2" » and Name = "Programm1-version2" , and StartTime and EndTime elements » indicating values after the time of test&lt;/li&gt;  * &lt;li&gt;Schedule fragment for » "Programm1-version2" content fragment with same values for startTime and » endTime in the presentationWindow element&lt;/li&gt;  * &lt;li&gt;Access fragment for </pre>	=	<pre>  * &lt;p&gt;Preconditions&lt;/p&gt;  * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;  * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;  * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;  * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:&lt;/p&gt;  * &lt;p&gt;Initial service guide:  * &lt;ul&gt;  * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;  * &lt;li&gt;Content fragment with version = "2" » and Name = "Programm1-version2" , and StartTime and EndTime elements » indicating values after the time of test&lt;/li&gt;  * &lt;li&gt;Schedule fragment for » "Programm1-version2" content fragment with same values for startTime and » endTime in the presentationWindow element&lt;/li&gt;  * &lt;li&gt;Access fragment for </pre>

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» "Programm1-version2" schedule fragment</li>
    *      </ul>
    *      </p>
    *      <p>Updated service guide:
    *      <ul>
    *          <li>Service fragment with
» Name="TvChannel"</li>
    *          <li>Content fragment with version = "1"
» and Name = "Programm1" , and StartTime and EndTime elements indicating
» values after the time of test</li>
    *          <li>Schedule fragment for "Programm1"
» content fragment with same values for startTime and endTime in the
» presentationWindow element</li>
    *          <li>Access fragment for "Programm1"
» schedule fragment</li>
    *      </ul>
    *      </p>
    *      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery</p>
    *      <p>Test Procedure</p>
    *      <p>
    *          <ul>
    *              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
    *              <li>Activate the BCAST application of
» the terminal</li>
    *              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
    *              <li>Browse the SG on the terminal</li>
    *              <li>Set up the test tool to produce
» updated BCAST service guide announcement and delivery using broadcast
» channel.</li>
    *              <li>Request from BCAST application on
» terminal to update the service guide (optional).</li>
    *              <li>Browse the SG on the terminal</li>
    *          </ul>
    *      </p>
    *      <p>Pass-Criteria</p>
    *      <p>The following should be visible to the end
» user after the initial delivery of the SG
    *      <ul>
    *          <li>There is a service called
» "TvChannel" associated with program "Programm1-version2" as well as start
» and end time values(There is a "TvChannel" that contains
» "Programm1-version2" scheduled from startTime to endTime</li>

```

```

» "Programm1-version2" schedule fragment</li>
    *      </ul>
    *      </p>
    *      <p>Updated service guide:
    *      <ul>
    *          <li>Service fragment with
» Name="TvChannel"</li>
    *          <li>Content fragment with version = "1"
» and Name = "Programm1" , and StartTime and EndTime elements indicating
» values after the time of test</li>
    *          <li>Schedule fragment for "Programm1"
» content fragment with same values for startTime and endTime in the
» presentationWindow element</li>
    *          <li>Access fragment for "Programm1"
» schedule fragment</li>
    *      </ul>
    *      </p>
    *      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery</p>
    *      <p>Test Procedure</p>
    *      <p>
    *          <ul>
    *              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
    *              <li>Activate the BCAST application of
» the terminal</li>
    *              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
    *              <li>Browse the SG on the terminal</li>
    *              <li>Set up the test tool to produce
» updated BCAST service guide announcement and delivery using broadcast
» channel.</li>
    *              <li>Request from BCAST application on
» terminal to update the service guide (optional).</li>
    *              <li>Browse the SG on the terminal</li>
    *          </ul>
    *      </p>
    *      <p>Pass-Criteria</p>
    *      <p>The following should be visible to the end
» user after the initial delivery of the SG
    *      <ul>
    *          <li>There is a service called
» "TvChannel" associated with program "Programm1-version2" as well as start
» and end time values(There is a "TvChannel" that contains
» "Programm1-version2" scheduled from startTime to endTime</li>

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> *      &lt;/ul&gt; *      &lt;/p&gt; *      &lt;p&gt;The following should be visible to the end » user after the delivery of the updated SG *      &lt;ul&gt; *          &lt;li&gt;There is a service called "TvChannel » associated with program "Programmel-version1" as well as start and end time » values(There is a "TvChannel" that contains "Programmel-version1" scheduled » from startTime to endTime)).&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; *      @verdict *          pass in case the client pass the conformance » test.  *      @verdict *          fail in case the client does not pass the » conformance test.  *      @verdict *          inconc in case a guard timer expires. */ testcase TC_BCAST_ESG_CONF_102() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up(); </pre>		<pre> *      &lt;/ul&gt; *      &lt;/p&gt; *      &lt;p&gt;The following should be visible to the end » user after the delivery of the updated SG *      &lt;ul&gt; *          &lt;li&gt;There is a service called "TvChannel » associated with program "Programmel-version1" as well as start and end time » values(There is a "TvChannel" that contains "Programmel-version1" scheduled » from startTime to endTime)).&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; *      @verdict *          pass in case the client pass the conformance » test.  *      @verdict *          fail in case the client does not pass the » conformance test.  *      @verdict *          inconc in case a guard timer expires. */ testcase TC_BCAST_ESG_CONF_102() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Interaction_up(); </pre>
<pre> » // change 09 (WK 6): set up broadcast channel     f_cf_UpperTester_up();      // preamble     f_startCommonUtPreamble();      var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>	=	<pre>     f_cf_UpperTester_up();      // preamble     f_startCommonUtPreamble();      var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>
<pre> » _hour); » definition     f_main_ctrl_InitStartTime(c_one_hour,c_one     // change 01 (WK 6): global time </pre>	<>	<pre>     var UInt v_NTP_StartTime := 0;      var UInt v_NTP_EndTime := 0;     f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); » definition     // change 01 (WK 6): global time  var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime); » time definition     // change 01 (WK 6): global </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>



Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp(  » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "TvChannel",     c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,  » f_IncFragmentVersion(v_FragmentVersion),     "Programmel-version2",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_FragmentVersion,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_startTime, </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp(  » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "TvChannel",     c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,  » f_IncFragmentVersion(v_FragmentVersion),     "Programmel-version2",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_FragmentVersion,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1, </pre>
<pre> » // change 01 (WK 6): global time definition     v_endTime » // change 01 (WK 6): global time definition </pre>	<>	<pre>     v_NTP_StartTime,      v_NTP_EndTime </pre>
<pre> ));  var AccessType v_Access := » valueof(m_def_Access( </pre>	=	<pre> ));  var AccessType v_Access := » valueof(m_def_Access( </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> PX_SGDU_ACCESS_ID_PROG_1, v_FragmentVersion, v_AccessType, PX_SGDU_SERVICE_ID, PX_SGDU_SCHEDULE_ID_PROG_1, c_class_SG  ));  f_addServiceFragment(v_SGDU, v_Service); f_addContentFragment(v_SGDU, v_Content); f_addScheduleFragment(v_SGDU, v_Schedule); f_addAccessFragment(v_SGDU, v_Access); f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);  f_main_ut_RunBCastApplication(); if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);     }     f_main_ut_CheckService("TvChannel");     f_main_ut_SelectService("TvChannel");     f_main_ut_CheckContent("Programm1-version2", » v_DateTime_StartTime, v_DateTime_EndTime);      // test behavior </pre>		<pre> PX_SGDU_ACCESS_ID_PROG_1, v_FragmentVersion, v_AccessType, PX_SGDU_SERVICE_ID, PX_SGDU_SCHEDULE_ID_PROG_1, c_class_SG  ));  f_addServiceFragment(v_SGDU, v_Service); f_addContentFragment(v_SGDU, v_Content); f_addScheduleFragment(v_SGDU, v_Schedule); f_addAccessFragment(v_SGDU, v_Access); f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);  f_main_ut_RunBCastApplication(); if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);     }     f_main_ut_CheckService("TvChannel");     f_main_ut_SelectService("TvChannel");     f_main_ut_CheckContent("Programm1-version2", » v_DateTime_StartTime, v_DateTime_EndTime);      // test behavior </pre>
<pre> vc_BCASTTransportID := 1; » // change 07 (WK 6): in order to send same TransportID v_SGDU := {}; » // change 10 (WK 6): empty SGDU to avoid multiple content fragments </pre>	<>	<pre> //v_SGDU := {}; // Change No. 1 from » OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review </pre>
<pre> v_Content.version := v_FragmentVersion; v_Content.Names := {{omit,"Programm1"}}; </pre>	=	<pre> v_Content.version := v_FragmentVersion; v_Content.Names := {{omit,"Programm1"}}; </pre>
<pre> f_addServiceFragment(v_SGDU, v_Service); » // change 08 (WK 6): SGDU must contain all fragments </pre>	+ -	
<pre> f_addContentFragment(v_SGDU, v_Content); </pre>	=	<pre> f_addContentFragment(v_SGDU, v_Content); </pre>
<pre> f_addScheduleFragment(v_SGDU, v_Schedule); » // change 08 (WK 6): SGDU must contain all fragments f_addAccessFragment(v_SGDU, v_Access); » // change 08 (WK 6): SGDU must contain all fragments f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp); // change 08 (WK 6): SGDU must contain all fragments  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit); // change 11 (WK 6): reset FLUTE; delivers new SGDU </pre>	+ -	

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre>         if (PX_GET_SG_UPDATE_MANUALLY) {             f_main_ut_UpdateServiceGuide();         }         f_main_ut_CheckService("TvChannel");         f_main_ut_SelectService("TvChannel");         f_main_ut_CheckContent("Programm1-version2", » v_DateTime_StartTime, v_DateTime_EndTime);          // postamble         f_main_ut_PowerOff();          f_cf_Broadcast_down();         f_cf_UpperTester_down();     } // end test case TC_BCAST_ESG_CONF_102      /**      *      * @desc      * &lt;p&gt;Service Guide Update with additional » fragments via broadcast channel&lt;/p&gt;      * &lt;p&gt;Test Case Description&lt;/p&gt;      * &lt;p&gt;with a terminal having received a Service » Guide and listening to the broadcast Service Guide Delivery Channel.&lt;/p&gt;      * &lt;p&gt;when the terminal receives an updated » Service Guide containing an additional content fragment&lt;/p&gt;      * &lt;p&gt;then the terminal presents the information » of the updated Service Guide&lt;/p&gt;      * &lt;p&gt;Specification Reference&lt;/p&gt;      * &lt;p&gt;[BCAST10-ESG] Section 5.4.2.1.1, 5.5 1st » bullet item&lt;/p&gt;      * &lt;p&gt;Preconditions&lt;/p&gt;      * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;      * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;      * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;      * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:&lt;/p&gt;      * &lt;p&gt;Initial service guide:      * &lt;ul&gt;      * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;      * &lt;li&gt;Content fragment with » Name="Programm1" , and StartTime and EndTime elements indicating values » after the time of test&lt;/li&gt;      * &lt;li&gt;Schedule fragment for content </pre>	<pre>         if (PX_GET_SG_UPDATE_MANUALLY) {             f_main_ut_UpdateServiceGuide();         }         f_main_ut_CheckService("TvChannel");         f_main_ut_SelectService("TvChannel");         f_main_ut_CheckContent("Programm1-version2", » v_DateTime_StartTime, v_DateTime_EndTime);          // postamble         f_main_ut_PowerOff();          f_cf_Broadcast_down();         f_cf_UpperTester_down();     } // end test case TC_BCAST_ESG_CONF_102      /**      *      * @desc      * &lt;p&gt;Service Guide Update with additional » fragments via broadcast channel&lt;/p&gt;      * &lt;p&gt;Test Case Description&lt;/p&gt;      * &lt;p&gt;with a terminal having received a Service » Guide and listening to the broadcast Service Guide Delivery Channel.&lt;/p&gt;      * &lt;p&gt;when the terminal receives an updated » Service Guide containing an additional content fragment&lt;/p&gt;      * &lt;p&gt;then the terminal presents the information » of the updated Service Guide&lt;/p&gt;      * &lt;p&gt;Specification Reference&lt;/p&gt;      * &lt;p&gt;[BCAST10-ESG] Section 5.4.2.1.1, 5.5 1st » bullet item&lt;/p&gt;      * &lt;p&gt;Preconditions&lt;/p&gt;      * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt;      * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt;      * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;      * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:&lt;/p&gt;      * &lt;p&gt;Initial service guide:      * &lt;ul&gt;      * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;      * &lt;li&gt;Content fragment with » Name="Programm1" , and StartTime and EndTime elements indicating values » after the time of test&lt;/li&gt;      * &lt;li&gt;Schedule fragment for content </pre>
--	--

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» fragment with same values for startTime and endTime in the
» presentationWindow element</li>
*      <li>Access fragment for schedule
» fragment with BroadcastServiceDelivery element and reference to
» "Programme1" session description fragment</li>
*      <li>Session description fragment for
» "Programme1"</li>
*      </ul>
*      </p>
*      <p>Updated service guide:
*      <ul>
*      <li>Service fragment with
» Name="TvChannel"</li>
*      <li>Content fragment with
» Name="Programme1" , and StartTime and EndTime elements indicating values
» after the time of test</li>
*      <li>Schedule fragment for "Programme1"
» content fragment with same values for startTime and endTime in the
» presentationWindow element</li>
*      <li>Access fragment for "Programme1"
» schedule fragment with BroadcastServiceDelivery element and a reference to
» "Programme1" session description fragment</li>
*      <li>Session description fragment for
» "Programme1"</li>
*      <li>Content fragment with
» Name="Programme2" , and different StartTime and EndTime elements indicating
» values after the time of test</li>
*      <li>Schedule fragment for "Programme2"
» content fragment with same different values for startTime and endTime in
» the presentationWindow element</li>
*      <li>Access fragment for "Programme2"
» schedule fragment with BroadcastServiceDelivery element and reference to
» "Programme2" session description fragment</li>
*      <li>Session description fragment for
» "Programme2" </li>
*      </ul>
*      </p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*      <p>Test Procedure</p>
*      <p>
*      <ul>
*      <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*      <li>Activate the BCAST application of

```

```

» fragment with same values for startTime and endTime in the
» presentationWindow element</li>
*      <li>Access fragment for schedule
» fragment with BroadcastServiceDelivery element and reference to
» "Programme1" session description fragment</li>
*      <li>Session description fragment for
» "Programme1"</li>
*      </ul>
*      </p>
*      <p>Updated service guide:
*      <ul>
*      <li>Service fragment with
» Name="TvChannel"</li>
*      <li>Content fragment with
» Name="Programme1" , and StartTime and EndTime elements indicating values
» after the time of test</li>
*      <li>Schedule fragment for "Programme1"
» content fragment with same values for startTime and endTime in the
» presentationWindow element</li>
*      <li>Access fragment for "Programme1"
» schedule fragment with BroadcastServiceDelivery element and a reference to
» "Programme1" session description fragment</li>
*      <li>Session description fragment for
» "Programme1"</li>
*      <li>Content fragment with
» Name="Programme2" , and different StartTime and EndTime elements indicating
» values after the time of test</li>
*      <li>Schedule fragment for "Programme2"
» content fragment with same different values for startTime and endTime in
» the presentationWindow element</li>
*      <li>Access fragment for "Programme2"
» schedule fragment with BroadcastServiceDelivery element and reference to
» "Programme2" session description fragment</li>
*      <li>Session description fragment for
» "Programme2" </li>
*      </ul>
*      </p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*      <p>Test Procedure</p>
*      <p>
*      <ul>
*      <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*      <li>Activate the BCAST application of

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           <li>Set up the test tool to produce
» updated BCAST service guide announcement and delivery using broadcast
» channel.</li>
*           <li>Request from BCAST application on
» terminal to update the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           </ul>
*           <p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible to the end
» user after the initial delivery of the SG
*           <ul>
*           <li>There is a service "TvChannel"
» associated with a program "Programm1" as well as start and end time
» values(There is a "TvChannel" that contains "Programm1" scheduled from
» startTime to endTime)</li>
*           </ul>
*           <p>
*           <p>The following should be visible to the end
» user after the delivery of the updated SG
*           <ul>
*           <li>There is a service "TvChannel"
» associated with programs "Programm1" and "Programme2" as well as different
» start and end time values(There is a "TvChannel" that contains "Programm1"
» and "Programme2" scheduled from different startTime to endTimes)</li>
*           </ul>
*           <p>
*           @verdict
*           pass in case the client pass the conformance
» test.
*           @verdict
*           fail in case the client does not pass the
» conformance test.
*           @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_103() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

```

```

» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           <li>Set up the test tool to produce
» updated BCAST service guide announcement and delivery using broadcast
» channel.</li>
*           <li>Request from BCAST application on
» terminal to update the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           </ul>
*           <p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible to the end
» user after the initial delivery of the SG
*           <ul>
*           <li>There is a service "TvChannel"
» associated with a program "Programm1" as well as start and end time
» values(There is a "TvChannel" that contains "Programm1" scheduled from
» startTime to endTime)</li>
*           </ul>
*           <p>
*           <p>The following should be visible to the end
» user after the delivery of the updated SG
*           <ul>
*           <li>There is a service "TvChannel"
» associated with programs "Programm1" and "Programme2" as well as different
» start and end time values(There is a "TvChannel" that contains "Programm1"
» and "Programme2" scheduled from different startTime to endTimes)</li>
*           </ul>
*           <p>
*           @verdict
*           pass in case the client pass the conformance
» test.
*           @verdict
*           fail in case the client does not pass the
» conformance test.
*           @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_103() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> // preamble f_startCommonUtPreamble();  var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>		<pre> // preamble f_startCommonUtPreamble();  var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>
<pre> » f_main_ctrl_InitStartTime(c_one_hour,c_one » _hour); » definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
	=	
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); » definition // change 01 (WK 6): global time  var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime); » time definition // change 01 (WK 6): global </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime2 := ""; var DateTime v_DateTime_EndTime2 := "";  var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_FragmentVersion, "TvChannel", c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content( PX_SGDU_CONTENT_ID_PROG_1, v_FragmentVersion, </pre>	=	<pre> var DateTime v_DateTime_StartTime2 := ""; var DateTime v_DateTime_EndTime2 := "";  var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_FragmentVersion, "TvChannel", c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content( PX_SGDU_CONTENT_ID_PROG_1, v_FragmentVersion, </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre>                 "Programm1",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_startTime, » // change 01 (WK 6): global time definition                 v_endTime » // change 01 (WK 6): global time definition             ));              var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             ));              f_addServiceFragment(v_SGDU, v_Service);             f_addContentFragment(v_SGDU, v_Content);             f_addScheduleFragment(v_SGDU, v_Schedule);             f_addAccessFragment(v_SGDU, v_Access);             f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);              f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);              f_main_ut_RunBCastApplication();             if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);                 }                 f_main_ut_CheckService("TvChannel");                 f_main_ut_SelectService("TvChannel");                 f_main_ut_CheckContent("Programm1", » v_DateTime_StartTime, v_DateTime_EndTime); </pre>	<>	<pre>                 "Programm1",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_FragmentVersion,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_NTP_StartTime,                 v_NTP_EndTime             ));              var AccessType v_Access := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_FragmentVersion,                 v_AccessType,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             ));              f_addServiceFragment(v_SGDU, v_Service);             f_addContentFragment(v_SGDU, v_Content);             f_addScheduleFragment(v_SGDU, v_Schedule);             f_addAccessFragment(v_SGDU, v_Access);             f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);              f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);              f_main_ut_RunBCastApplication();             if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);                 }                 f_main_ut_CheckService("TvChannel");                 f_main_ut_SelectService("TvChannel");                 f_main_ut_CheckContent("Programm1", » v_DateTime_StartTime, v_DateTime_EndTime); </pre>
--	----	--

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> // test behavior v_startTime := f_getNTPTime() + (2 * » c_one_hour); // one hour later than the first one // change 01 (WK 6): » global time definition v_endTime := v_startTime + c_one_hour; » // change 01 (WK 6): global time definition </pre>	<>	<pre> // test behavior v_NTP_StartTime := f_getNTPTime() + (2 * » c_one_hour); // one hour later than the first one v_NTP_EndTime := v_NTP_StartTime + » c_one_hour; </pre>
<pre> v_DateTime_StartTime2 := f_getDateTime(v_star » tTime); // change 01 (WK 6): global » time definition v_DateTime_EndTime2 := f_getDateTime(v_endTim » e); // change 01 (WK 6): global » time definition </pre>	<>	<pre> v_DateTime_StartTime2 := f_getDateTime(v_NTP_ » StartTime); v_DateTime_EndTime2 := f_getDateTime(v_NTP_En » dTime); </pre>
<pre> v_Sdp := valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_2_REF_ID, PX_SDP_VIDEO_PROG_2 )); v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_2_REF_ID )); v_Content := valueof(m_def_Content( PX_SGDU_CONTENT_ID_PROG_2, v_FragmentVersion, "Programme2", PX_SGDU_SERVICE_ID, v_DateTime_StartTime2, v_DateTime_EndTime2 )); v_Schedule := valueof(m_def_Schedule( PX_SGDU_SCHEDULE_ID_PROG_2, v_FragmentVersion, PX_SGDU_SERVICE_ID, PX_SGDU_CONTENT_ID_PROG_2, v_startTime, v_endTime )); </pre>	=	<pre> v_Sdp := valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_2_REF_ID, PX_SDP_VIDEO_PROG_2 )); v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_2_REF_ID )); v_Content := valueof(m_def_Content( PX_SGDU_CONTENT_ID_PROG_2, v_FragmentVersion, "Programme2", PX_SGDU_SERVICE_ID, v_DateTime_StartTime2, v_DateTime_EndTime2 )); v_Schedule := valueof(m_def_Schedule( PX_SGDU_SCHEDULE_ID_PROG_2, v_FragmentVersion, PX_SGDU_SERVICE_ID, PX_SGDU_CONTENT_ID_PROG_2, v_NTP_StartTime, v_NTP_EndTime )); </pre>
<pre> » // change 01 (WK 6): global time definition v_endTime » // change 01 (WK 6): global time definition </pre>	<>	<pre> v_NTP_StartTime, v_NTP_EndTime </pre>
<pre> )); v_Access := valueof(m_def_Access( </pre>	=	<pre> )); v_Access := valueof(m_def_Access( </pre>



Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> PX_SGDU_ACCESS_ID_PROG_2, v_FragmentVersion, v_AccessType, PX_SGDU_SERVICE_ID, PX_SGDU_SCHEDULE_ID_PROG_2, c_class_SG  )); </pre>		<pre> PX_SGDU_ACCESS_ID_PROG_2, v_FragmentVersion, v_AccessType, PX_SGDU_SERVICE_ID, PX_SGDU_SCHEDULE_ID_PROG_2, c_class_SG  )); </pre>
<pre> // v_SGDU := {}; » // change 12 (WK 6): SGDU must contain additional fragments for » 'Programme2' </pre>	<>	<pre> v_SGDU := {}; </pre>
<pre> f_addContentFragment(v_SGDU, v_Content); f_addScheduleFragment(v_SGDU, v_Schedule); f_addAccessFragment(v_SGDU, v_Access); f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);  if (PX_GET_SG_UPDATE_MANUALLY) {     f_main_ut_UpdateServiceGuide(); } f_main_ut_CheckService("TvChannel"); f_main_ut_SelectService("TvChannel"); f_main_ut_CheckContent("Programme1", » v_DateTime_StartTime, v_DateTime_EndTime); f_main_ut_CheckContent("Programme2", » v_DateTime_StartTime2, v_DateTime_EndTime2);  // postamble f_main_ut_PowerOff();  f_cf_Broadcast_down(); f_cf_UpperTester_down(); } // end test case TC_BCAST_ESG_CONF_103  /**  *  * @desc  * &lt;p&gt;GZIP compression of Service Guide Delivery » Unit on broadcast channel&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt;  * &lt;p&gt;with a terminal listening to the broadcast » Service Guide Delivery Channel.&lt;/p&gt;  * &lt;p&gt;when the terminal receives a GZIP compressed </pre>	=	<pre> f_addContentFragment(v_SGDU, v_Content); f_addScheduleFragment(v_SGDU, v_Schedule); f_addAccessFragment(v_SGDU, v_Access); f_addSdpFragment(v_SGDU, v_FragmentVersion, » v_Sdp);  f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);  if (PX_GET_SG_UPDATE_MANUALLY) {     f_main_ut_UpdateServiceGuide(); } f_main_ut_CheckService("TvChannel"); f_main_ut_SelectService("TvChannel"); f_main_ut_CheckContent("Programme1", » v_DateTime_StartTime, v_DateTime_EndTime); f_main_ut_CheckContent("Programme2", » v_DateTime_StartTime2, v_DateTime_EndTime2);  // postamble f_main_ut_PowerOff();  f_cf_Broadcast_down(); f_cf_UpperTester_down(); } // end test case TC_BCAST_ESG_CONF_103  /**  *  * @desc  * &lt;p&gt;GZIP compression of Service Guide Delivery » Unit on broadcast channel&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt;  * &lt;p&gt;with a terminal listening to the broadcast » Service Guide Delivery Channel.&lt;/p&gt;  * &lt;p&gt;when the terminal receives a GZIP compressed </pre>

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> » Service Guide&lt;/p&gt; *      &lt;p&gt;then the terminal presents the information » of the Service Guide&lt;/p&gt; *      &lt;p&gt;Specification Reference&lt;/p&gt; </pre>		<pre> » Service Guide&lt;/p&gt; *      &lt;p&gt;then the terminal presents the information » of the Service Guide&lt;/p&gt; *      &lt;p&gt;Specification Reference&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;[BCAST10-ESG] Section 5.4.1.4.&lt;/p&gt; </pre>	<>	<pre> *      &lt;p&gt;[BCAST10-ESG] Section 5.4.1.4.&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Preconditions&lt;/p&gt; *      &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; </pre>	=	<pre> *      &lt;p&gt;Preconditions&lt;/p&gt; *      &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; *      &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; *      &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix: *      &lt;ul&gt; </pre>		<pre> *      &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix: *      &lt;ul&gt; </pre>
<pre> *          &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt; </pre>		<pre> *          &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt; </pre>
<pre> *          &lt;li&gt;Content fragment with » Name="Programm1"&lt;/li&gt; </pre>		<pre> *          &lt;li&gt;Content fragment with » Name="Programm1"&lt;/li&gt; </pre>
<pre> *          &lt;li&gt;Schedule fragment for content » fragment&lt;/li&gt; </pre>		<pre> *          &lt;li&gt;Schedule fragment for content » fragment&lt;/li&gt; </pre>
<pre> *          &lt;li&gt;Access fragment for schedule » fragment&lt;/li&gt; </pre>		<pre> *          &lt;li&gt;Access fragment for schedule » fragment&lt;/li&gt; </pre>
<pre> *      &lt;/ul&gt; *      &lt;/p&gt; </pre>		<pre> *      &lt;/ul&gt; *      &lt;/p&gt; </pre>
<pre> *      &lt;p&gt;All fragments are packaged in one SGDU, » which is GZIP compressed.&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;All fragments are packaged in one SGDU, » which is GZIP compressed.&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Test Procedure&lt;/p&gt; *      &lt;p&gt; </pre>		<pre> *      &lt;p&gt;Test Procedure&lt;/p&gt; *      &lt;p&gt; </pre>
<pre> *          &lt;ul&gt; </pre>		<pre> *          &lt;ul&gt; </pre>
<pre> *              &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using broadcast channel.&lt;/li&gt; </pre>		<pre> *              &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using broadcast channel.&lt;/li&gt; </pre>
<pre> *              &lt;li&gt;Activate the BCAST application of » the terminal&lt;/li&gt; </pre>		<pre> *              &lt;li&gt;Activate the BCAST application of » the terminal&lt;/li&gt; </pre>
<pre> *              &lt;li&gt;Request from BCAST application on » terminal to get the service guide (optional)&lt;/li&gt; </pre>		<pre> *              &lt;li&gt;Request from BCAST application on » terminal to get the service guide (optional)&lt;/li&gt; </pre>
<pre> *              &lt;li&gt;Browse the SG on the terminal&lt;/li&gt; </pre>		<pre> *              &lt;li&gt;Browse the SG on the terminal&lt;/li&gt; </pre>
<pre> *          &lt;/ul&gt; *      &lt;/p&gt; </pre>		<pre> *          &lt;/ul&gt; *      &lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Pass-Criteria&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;Pass-Criteria&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;The following should be visible to the end » user </pre>		<pre> *      &lt;p&gt;The following should be visible to the end » user </pre>

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> *      &lt;ul&gt; *          &lt;li&gt;There is a service "TvChannel" » associated with programme "Programm1"(There is a "TvChannel" that contains » "Programm1")&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; *      @verdict *          pass in case the client pass the conformance » test. *      @verdict *          fail in case the client does not pass the » conformance test. *      @verdict *          inconc in case a guard timer expires. */ testcase TC_BCAST_ESG_CONF_104() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up();     f_cf_UpperTester_up();      f_startCommonUtPreamble();      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_Version := 1; </pre>		<pre> *      &lt;ul&gt; *          &lt;li&gt;There is a service "TvChannel" » associated with programme "Programm1"(There is a "TvChannel" that contains » "Programm1")&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; *      @verdict *          pass in case the client pass the conformance » test. *      @verdict *          fail in case the client does not pass the » conformance test. *      @verdict *          inconc in case a guard timer expires. */ testcase TC_BCAST_ESG_CONF_104() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Broadcast_up();     f_cf_UpperTester_up();      f_startCommonUtPreamble();      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_Version := 1; </pre>
<pre> » _hour); » definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartEndTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
	=	
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime);          // change 01 (WK 6): » global time definition var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);            // change 01 (WK 6): » global time definition </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime); var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
	=	
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_VIDEO_PROG_1 )); </pre>		<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_VIDEO_PROG_1 )); </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

        var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(
    f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
        PX_SDP_VIDEO_PROG_1_REF_ID
    ));

    var ServiceType v_Service :=
» valueof(m_def_Service(
        PX_SGDU_SERVICE_ID,
        v_Version,
        "TvChannel",
        c_basicTv_ServiceType
    ));

    var ContentType v_Content :=
» valueof(m_def_Content(
        PX_SGDU_CONTENT_ID_PROG_1,
        v_Version,
        "Programmel",
        PX_SGDU_SERVICE_ID,
        v_DateTime_StartTime,
        v_DateTime_EndTime
    ));

    var ScheduleType v_Schedule :=
» valueof(m_def_Schedule(
        PX_SGDU_SCHEDULE_ID_PROG_1,
        v_Version,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_CONTENT_ID_PROG_1,
        v_startTime,

```

```

» // change 01 (WK 6): global time definition
    v_endTime
» // change 01 (WK 6): global time definition

```

```

    ));

```

```

    var AccessType v_Access :=
» valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_1,
        v_Version,
        v_AccessType,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_SCHEDULE_ID_PROG_1,
        c_class_SG
    ));

```

```

        var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(
    f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
        PX_SDP_VIDEO_PROG_1_REF_ID
    ));

    var ServiceType v_Service :=
» valueof(m_def_Service(
        PX_SGDU_SERVICE_ID,
        v_Version,
        "TvChannel",
        c_basicTv_ServiceType
    ));

    var ContentType v_Content :=
» valueof(m_def_Content(
        PX_SGDU_CONTENT_ID_PROG_1,
        v_Version,
        "Programmel",
        PX_SGDU_SERVICE_ID,
        v_DateTime_StartTime,
        v_DateTime_EndTime
    ));

    var ScheduleType v_Schedule :=
» valueof(m_def_Schedule(
        PX_SGDU_SCHEDULE_ID_PROG_1,
        v_Version,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_CONTENT_ID_PROG_1,
        v_NTP_StartTime,

```

```

    v_NTP_EndTime

```

```

    ));

```

```

    var AccessType v_Access :=
» valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_1,
        v_Version,
        v_AccessType,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_SCHEDULE_ID_PROG_1,
        c_class_SG
    ));

```

&lt;&gt;

=

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

        f_addServiceFragment(v_SGDU, v_Service);
        f_addContentFragment(v_SGDU, v_Content);
        f_addScheduleFragment(v_SGDU, v_Schedule);
        f_addAccessFragment(v_SGDU, v_Access);
        f_addSdpFragment(v_SGDU, v_Version, v_Sdp);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,

» "gzip");

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");
        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckContent("Programm1",
» v_DateTime_StartTime, v_DateTime_EndTime );

        // postamble
        f_main_ut_PowerOff();

        f_cf_Broadcast_down();
        f_cf_UpperTester_down();
    }// end test case TC_BCAST_ESG_CONF_104

/**
 *
 * @desc
 * <p>Service provisioning with Content
» hierarchy</p>
 *
 * <p>Test Case Description</p>
 * <p>with a terminal listening to the Service
» Guide Delivery Channel and for a data stream on the broadcast channel</p>
 * <p>when the terminal receives an Service Guide
» containing service fragments with two content fragments that follow each
» other in time</p>
 *
 * <p>then the terminal presents the information
» of the Service Guide and is able to receive the data stream correctly</p>
 *
 * <p>Specification Reference</p>
 * <p>[BCAST10-ESG] Section 5.1.2.1, 5.1.2.2,
» 5.1.2.3, 5.1.2.4, 5.1.2.5.1, 5.4.2.1, 6.1.1</p>
 *
 * <p>Preconditions</p>
 * <p>Service guide cache of terminal is
» erased.</p>
 *
 * <p>Terminal is configured to listen to BCAST

```

```

        f_addServiceFragment(v_SGDU, v_Service);
        f_addContentFragment(v_SGDU, v_Content);
        f_addScheduleFragment(v_SGDU, v_Schedule);
        f_addAccessFragment(v_SGDU, v_Access);
        f_addSdpFragment(v_SGDU, v_Version, v_Sdp);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,

» "gzip");

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");
        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckContent("Programm1",
» v_DateTime_StartTime, v_DateTime_EndTime );

        // postamble
        f_main_ut_PowerOff();

        f_cf_Broadcast_down();
        f_cf_UpperTester_down();
    }// end test case TC_BCAST_ESG_CONF_104

/**
 *
 * @desc
 * <p>Service provisioning with Content
» hierarchy</p>
 *
 * <p>Test Case Description</p>
 * <p>with a terminal listening to the Service
» Guide Delivery Channel and for a data stream on the broadcast channel</p>
 * <p>when the terminal receives an Service Guide
» containing service fragments with two content fragments that follow each
» other in time</p>
 *
 * <p>then the terminal presents the information
» of the Service Guide and is able to receive the data stream correctly</p>
 *
 * <p>Specification Reference</p>
 * <p>[BCAST10-ESG] Section 5.1.2.1, 5.1.2.2,
» 5.1.2.3, 5.1.2.4, 5.1.2.5.1, 5.4.2.1, 6.1.1</p>
 *
 * <p>Preconditions</p>
 * <p>Service guide cache of terminal is
» erased.</p>
 *
 * <p>Terminal is configured to listen to BCAST

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» service guide announcements and delivery on the broadcast channel</p>
    *      <p>Access point information for service guide
» entry point is configured in test tool</p>
    *      <p>This test case uses the following SG
» fragment instantiations found in the Appendix:
    *      <ul>
    *          <li>Service fragment with
» Name="TvChannel"</li>
    *          <li>Content fragment for service
» fragment with Name="Programme1" with StartTime and EndTime elements
» indicating values within test execution time that result in a play time of
» three minutes</li>
    *          <li>Schedule fragment for content
» fragment "Programme1" with same values for startTime and endTime in the
» presentationWindow element</li>
    *          <li>Access fragment for "Programme1"
» schedule fragment with BroadcastServiceDelivery element and a reference to
» "Programme1" session description fragment</li>
    *          <li>Session description fragment for
» "Programme1"</li>
    *          <li>Content fragment for service
» fragment with Name="Programme2" with StartTime and EndTime elements
» indicating values within test execution time but after the ones of
» "Programme1"</li>
    *          <li>Schedule fragment for content
» fragment "Programme2" with same values for startTime and endTime in the
» presentationWindow element</li>
    *          <li>Access fragment for "Programme2"
» schedule fragment with BroadcastServiceDelivery element and a reference to
» "Programme2" session description fragment</li>
    *          <li>Session description fragment for
» "Programme2"</li>
    *      </ul>
    *      </p>
    *      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
    *      <p>Test Procedure</p>
    *      <p>
    *          <ul>
    *              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
    *              <li>Setup the test tool to stream audio
» and video for "Programme1" and "Programme2" via the broadcast channel</li>
    *              <li>Activate the BCAST application of
» the terminal</li>

```

```

» service guide announcements and delivery on the broadcast channel</p>
    *      <p>Access point information for service guide
» entry point is configured in test tool</p>
    *      <p>This test case uses the following SG
» fragment instantiations found in the Appendix:
    *      <ul>
    *          <li>Service fragment with
» Name="TvChannel"</li>
    *          <li>Content fragment for service
» fragment with Name="Programme1" with StartTime and EndTime elements
» indicating values within test execution time that result in a play time of
» three minutes</li>
    *          <li>Schedule fragment for content
» fragment "Programme1" with same values for startTime and endTime in the
» presentationWindow element</li>
    *          <li>Access fragment for "Programme1"
» schedule fragment with BroadcastServiceDelivery element and a reference to
» "Programme1" session description fragment</li>
    *          <li>Session description fragment for
» "Programme1"</li>
    *          <li>Content fragment for service
» fragment with Name="Programme2" with StartTime and EndTime elements
» indicating values within test execution time but after the ones of
» "Programme1"</li>
    *          <li>Schedule fragment for content
» fragment "Programme2" with same values for startTime and endTime in the
» presentationWindow element</li>
    *          <li>Access fragment for "Programme2"
» schedule fragment with BroadcastServiceDelivery element and a reference to
» "Programme2" session description fragment</li>
    *          <li>Session description fragment for
» "Programme2"</li>
    *      </ul>
    *      </p>
    *      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
    *      <p>Test Procedure</p>
    *      <p>
    *          <ul>
    *              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
    *              <li>Setup the test tool to stream audio
» and video for "Programme1" and "Programme2" via the broadcast channel</li>
    *              <li>Activate the BCAST application of
» the terminal</li>

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

*      <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*      <li>Browse the SG on the terminal</li>
*      <li>Access "TvChannel" on terminal</li>
*      </ul>
*      </p>
*      <p>The used video codec is:
*      <ul>
*          <li>H.264 (AVC)</li>
*      </ul>
*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the delivery of the SG
*      <ul>
*          <li>There is a service "TvChannel"
» associated with a program "Programme1" followed three minutes later by a
» program "Programme2"(There is a "TvChannel" that contains "Programme1"
» scheduled three minutes prior to "Programme2")</li>
*          <li>Terminal shows "Programme1" for
» three minutes which is followed by "Programme2"</li>
*      </ul>
*      </p>
*      @verdict
*      pass in case the client pass the conformance
» test.
*      @verdict
*      fail in case the client does not pass the
» conformance test.
*      @verdict
*      inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_105() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();

    // preamble

    f_main_data_startStreamingFile(0,
» PX_FILE_VIDEO_PROG1, PX_SDP_VIDEO_PROG_1);
    //f_main_data_startStreamingFile(1,
» PX_FILE_AUDIO_PROG_1, PX_SDP_AUDIO_PROG_1); // 1st programm broadcasts
» video and audio

```

```

*      <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*      <li>Browse the SG on the terminal</li>
*      <li>Access "TvChannel" on terminal</li>
*      </ul>
*      </p>
*      <p>The used video codec is:
*      <ul>
*          <li>H.264 (AVC)</li>
*      </ul>
*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the delivery of the SG
*      <ul>
*          <li>There is a service "TvChannel"
» associated with a program "Programme1" followed three minutes later by a
» program "Programme2"(There is a "TvChannel" that contains "Programme1"
» scheduled three minutes prior to "Programme2")</li>
*          <li>Terminal shows "Programme1" for
» three minutes which is followed by "Programme2"</li>
*      </ul>
*      </p>
*      @verdict
*      pass in case the client pass the conformance
» test.
*      @verdict
*      fail in case the client does not pass the
» conformance test.
*      @verdict
*      inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_105() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();

    // preamble

    f_main_data_startStreamingFile(0,
» PX_FILE_VIDEO_PROG1, PX_SDP_VIDEO_PROG_1);
    f_main_data_startStreamingFile(1,
» PX_FILE_AUDIO_PROG_1, PX_SDP_AUDIO_PROG_1);

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> f_main_data_startStreamingFile(1, » PX_FILE_VIDEO_PROG2, PX_SDP_VIDEO_PROG_2); //f_main_data_startStreamingFile(3, » PX_FILE_AUDIO_PROG_2, PX_SDP_AUDIO_PROG_2); // 2st programm broadcasts » video and audio </pre>	=	<pre> f_main_data_startStreamingFile(2, » PX_FILE_VIDEO_PROG2, PX_SDP_VIDEO_PROG_2); f_main_data_startStreamingFile(3, » PX_FILE_AUDIO_PROG_2, PX_SDP_AUDIO_PROG_2); </pre>
<pre> f_startCommonUtPreamble();  // test behavior // current time var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_Version := 1; </pre>	=	<pre> f_startCommonUtPreamble();  // test behavior // current time var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_Version := 1; </pre>
<pre> f_main_ctrl_InitStartTime(0,PX_PLAY_TIME); » // change 01 (WK 6): global time definition </pre>	<>	<pre> var UInt v_NTP_StartTime1 := f_getNTPTime();  // current time + paly time var UInt v_NTP_EndTime1 := v_NTP_StartTime1 + » PX_PLAY_TIME; </pre>
<pre> // time after v_endTime var UInt v_NTP_StartTime2 := v_endTime; » // change 01 (WK 6): global time definition </pre>	<>	<pre> // time after v_NTP_EndTime1 var UInt v_NTP_StartTime2 := v_NTP_EndTime1; </pre>
<pre> // 1 hour duration; var UInt v_NTP_EndTime2 := v_NTP_StartTime2 + » c_one_hour; » definition </pre>	=	<pre> // 1 hour duration; var UInt v_NTP_EndTime2 := v_NTP_StartTime2 + » c_one_hour; </pre>
<pre> var DateTime v_DateTime_StartTime1 := » f_getDateTime(v_startTime); // change 01 (WK 6): global » time definition var DateTime v_DateTime_EndTime1 := » f_getDateTime(v_endTime); // change 01 (WK 6): global » time definition </pre>	<>	<pre> var DateTime v_DateTime_StartTime1 := » f_getDateTime(v_NTP_StartTime1); var DateTime v_DateTime_EndTime1 := » f_getDateTime(v_NTP_EndTime1); </pre>
<pre> var DateTime v_DateTime_StartTime2 := » f_getDateTime(v_NTP_StartTime2); var DateTime v_DateTime_EndTime2 := » f_getDateTime(v_NTP_EndTime2);  var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := </pre>	=	<pre> var DateTime v_DateTime_StartTime2 := » f_getDateTime(v_NTP_StartTime2); var DateTime v_DateTime_EndTime2 := » f_getDateTime(v_NTP_EndTime2);  var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := </pre>



## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                  PX_SDP_VIDEO_PROG_1_REF_ID
                ));

    var ServiceType v_Service := valueof(

» m_def_Service(
                  PX_SGDU_SERVICE_ID,
                  v_Version,
                  "TvChannel",
                  c_basicTv_ServiceType
                ));

    var ContentType v_Content :=

» valueof(m_def_Content(
                  PX_SGDU_CONTENT_ID_PROG_1,
                  v_Version,
                  "Programmel",
                  PX_SGDU_SERVICE_ID,
                  v_DateTime_StartTime1,
                  v_DateTime_EndTime1
                ));

    var ScheduleType v_Schedule :=

» valueof(m_def_Schedule(
                  PX_SGDU_SCHEDULE_ID_PROG_1,
                  v_Version,
                  PX_SGDU_SERVICE_ID,
                  PX_SGDU_CONTENT_ID_PROG_1,
                  v_startTime,
                  v_endTime
                ));

» // change 01 (WK 6): global time definition
» // change 01 (WK 6): global time definition

    var AccessType v_Access :=

» valueof(m_def_Access(
                  PX_SGDU_ACCESS_ID_PROG_1,
                  v_Version,
                  v_AccessType,
                  PX_SGDU_SERVICE_ID,
                  PX_SGDU_SCHEDULE_ID_PROG_1,
                  c_class_SG
                ));

    var SdpFragment v_Sdp2 :=

```

```

» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                  PX_SDP_VIDEO_PROG_1_REF_ID
                ));

    var ServiceType v_Service := valueof(

» m_def_Service(
                  PX_SGDU_SERVICE_ID,
                  v_Version,
                  "TvChannel",
                  c_basicTv_ServiceType
                ));

    var ContentType v_Content :=

» valueof(m_def_Content(
                  PX_SGDU_CONTENT_ID_PROG_1,
                  v_Version,
                  "Programmel",
                  PX_SGDU_SERVICE_ID,
                  v_DateTime_StartTime1,
                  v_DateTime_EndTime1
                ));

    var ScheduleType v_Schedule :=

» valueof(m_def_Schedule(
                  PX_SGDU_SCHEDULE_ID_PROG_1,
                  v_Version,
                  PX_SGDU_SERVICE_ID,
                  PX_SGDU_CONTENT_ID_PROG_1,
                  v_NTP_StartTime1,
                  v_NTP_EndTime1
                ));

    var AccessType v_Access :=

» valueof(m_def_Access(
                  PX_SGDU_ACCESS_ID_PROG_1,
                  v_Version,
                  v_AccessType,
                  PX_SGDU_SERVICE_ID,
                  PX_SGDU_SCHEDULE_ID_PROG_1,
                  c_class_SG
                ));

    var SdpFragment v_Sdp2 :=

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» valueof(m_def_SdpFragment(
                                PX_SDP_VIDEO_PROG_2_REF_ID,
                                PX_SDP_VIDEO_PROG_2
                                ));

                                var AccessTypeType v_AccessType2 :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                                PX_SDP_VIDEO_PROG_2_REF_ID
                                ));

                                var ContentType v_Content2 :=
» valueof(m_def_Content(
                                PX_SGDU_CONTENT_ID_PROG_2,
                                v_Version,
                                "Programme2",
                                PX_SGDU_SERVICE_ID,
                                v_DateTime_StartTime2,
                                v_DateTime_EndTime2
                                ));

                                var ScheduleType v_Schedule2 :=
» valueof(m_def_Schedule(
                                PX_SGDU_SCHEDULE_ID_PROG_2,
                                v_Version,
                                PX_SGDU_SERVICE_ID,
                                PX_SGDU_CONTENT_ID_PROG_2,
                                v_NTP_StartTime2,
                                v_NTP_EndTime2
                                ));

                                var AccessType v_Access2 :=
» valueof(m_def_Access(
                                PX_SGDU_ACCESS_ID_PROG_2,
                                v_Version,
                                v_AccessType,
                                PX_SGDU_SERVICE_ID,
                                PX_SGDU_SCHEDULE_ID_PROG_2,
                                c_class_SG
                                ));

                                f_addServiceFragment(v_SGDU, v_Service);
                                f_addContentFragment(v_SGDU, v_Content);
                                f_addScheduleFragment(v_SGDU, v_Schedule);
                                f_addAccessFragment(v_SGDU, v_Access);
                                f_addContentFragment(v_SGDU, v_Content2);

```

```

» valueof(m_def_SdpFragment(
                                PX_SDP_VIDEO_PROG_2_REF_ID,
                                PX_SDP_VIDEO_PROG_2
                                ));

                                var AccessTypeType v_AccessType2 :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                                PX_SDP_VIDEO_PROG_2_REF_ID
                                ));

                                var ContentType v_Content2 :=
» valueof(m_def_Content(
                                PX_SGDU_CONTENT_ID_PROG_2,
                                v_Version,
                                "Programme2",
                                PX_SGDU_SERVICE_ID,
                                v_DateTime_StartTime2,
                                v_DateTime_EndTime2
                                ));

                                var ScheduleType v_Schedule2 :=
» valueof(m_def_Schedule(
                                PX_SGDU_SCHEDULE_ID_PROG_2,
                                v_Version,
                                PX_SGDU_SERVICE_ID,
                                PX_SGDU_CONTENT_ID_PROG_2,
                                v_NTP_StartTime2,
                                v_NTP_EndTime2
                                ));

                                var AccessType v_Access2 :=
» valueof(m_def_Access(
                                PX_SGDU_ACCESS_ID_PROG_2,
                                v_Version,
                                v_AccessType,
                                PX_SGDU_SERVICE_ID,
                                PX_SGDU_SCHEDULE_ID_PROG_2,
                                c_class_SG
                                ));

                                f_addServiceFragment(v_SGDU, v_Service);
                                f_addContentFragment(v_SGDU, v_Content);
                                f_addScheduleFragment(v_SGDU, v_Schedule);
                                f_addAccessFragment(v_SGDU, v_Access);
                                f_addContentFragment(v_SGDU, v_Content2);

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

        f_addScheduleFragment(v_SGDU, v_Schedule2);
        f_addAccessFragment(v_SGDU, v_Access2);
        f_addSdpFragment(v_SGDU, v_Version, v_Sdp);
        f_addSdpFragment(v_SGDU, v_Version, v_Sdp2);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
            }
            f_main_ut_CheckService("TvChannel");
            f_main_ut_SelectService("TvChannel");
            f_main_ut_CheckContent("Programme1",
» v_DateTime_StartTime1, v_DateTime_EndTime1);
            f_main_ut_CheckContent("Programme2",
» v_DateTime_StartTime2, v_DateTime_EndTime2);
            f_main_ut_UseService("TvChannel");

            f_main_ut_CheckVideo("Programme1");

            // waiting end of play time of Programme1
            timer t_wait;
            t_wait.start(int2float(PX_PLAY_TIME));
            t_wait.timeout;

            f_main_ut_CheckVideo("Programme2");

            // postamble
            f_main_ut_PowerOff();
            f_main_data_stopStreaming(0);
            f_main_data_stopStreaming(1);
            f_main_data_stopStreaming(2);
            f_main_data_stopStreaming(3);

            f_cf_Broadcast_down();
            f_cf_Data_down();
            f_cf_UpperTester_down();
        } // end test case TC_BCAST_ESG_CONF_105

/**
 *
 * @desc
 *      <p>Service guide with Preview Data via
» broadcast channel</p>

```

```

        f_addScheduleFragment(v_SGDU, v_Schedule2);
        f_addAccessFragment(v_SGDU, v_Access2);
        f_addSdpFragment(v_SGDU, v_Version, v_Sdp);
        f_addSdpFragment(v_SGDU, v_Version, v_Sdp2);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
            }
            f_main_ut_CheckService("TvChannel");
            f_main_ut_SelectService("TvChannel");
            f_main_ut_CheckContent("Programme1",
» v_DateTime_StartTime1, v_DateTime_EndTime1);
            f_main_ut_CheckContent("Programme2",
» v_DateTime_StartTime2, v_DateTime_EndTime2);
            f_main_ut_UseService("TvChannel");

            f_main_ut_CheckVideo("Programme1");

            // waiting end of play time of Programme1
            timer t_wait;
            t_wait.start(int2float(PX_PLAY_TIME));
            t_wait.timeout;

            f_main_ut_CheckVideo("Programme2");

            // postamble
            f_main_ut_PowerOff();
            f_main_data_stopStreaming(0);
            f_main_data_stopStreaming(1);
            f_main_data_stopStreaming(2);
            f_main_data_stopStreaming(3);

            f_cf_Broadcast_down();
            f_cf_Data_down();
            f_cf_UpperTester_down();
        } // end test case TC_BCAST_ESG_CONF_105

/**
 *
 * @desc
 *      <p>Service guide with Preview Data via
» broadcast channel</p>

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> *      &lt;p&gt;Test Case Description&lt;/p&gt; *      &lt;p&gt;with a terminal listening to the broadcast » Service Guide Delivery Channel.&lt;/p&gt; *      &lt;p&gt;when the terminal receives a Service Guide » containing a service and preview data fragments indicating a picture » reference and alternative text&lt;/p&gt; *      &lt;p&gt;then the terminal presents the service and » the picture or the alternative text&lt;/p&gt; *      &lt;p&gt;Specification Reference&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;Test Case Description&lt;/p&gt; *      &lt;p&gt;with a terminal listening to the broadcast » Service Guide Delivery Channel.&lt;/p&gt; *      &lt;p&gt;when the terminal receives a Service Guide » containing a service and preview data fragments indicating a picture » reference and alternative text&lt;/p&gt; *      &lt;p&gt;then the terminal presents the service and » the picture or the alternative text&lt;/p&gt; *      &lt;p&gt;Specification Reference&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;[BCAST10-ESG] Section 5.1.2.9&lt;/p&gt; </pre>	<>	<pre> *      &lt;p&gt;[BCAST10 -ESG] Section 5.1.2.9&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Preconditions&lt;/p&gt; *      &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; *      &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; *      &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; *      &lt;p&gt;Preview icon is delivered using a FLUTE » file delivery session.&lt;/p&gt; *      &lt;p&gt;This test cases uses the following SG » fragment instantiations and files found in the Appendix: *      &lt;ul&gt; *          &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt; *          &lt;li&gt;PreviewData fragment for service » fragment with PictureUri and AlternativeText element set to "TV Channel » icon"&lt;/li&gt; *          &lt;li&gt;Access fragment for previewData » fragment with reference to session description fragment&lt;/li&gt; *          &lt;li&gt;Session description fragment with » parameters for broadcast file delivery session&lt;/li&gt; *          &lt;li&gt;Image file "TvChannelIcon.jpg"&lt;/li&gt; *      &lt;/ul&gt; *      &lt;p&gt; *      &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; *      &lt;p&gt;Set the FLUTE session to contain an » FDT-instance with 'File'-element, where 'Content-Location= » "TvChannelIcon.jpg"'.&lt;/p&gt; *      &lt;p&gt;Test Procedure&lt;/p&gt; *      &lt;p&gt; *      &lt;ul&gt; *          &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using broadcast channel.&lt;/li&gt; *          &lt;li&gt;Setup the test tool to deliver the » image file "TvChannelIcon.jpg" using a file delivery session as described </pre>	=	<pre> *      &lt;p&gt;Preconditions&lt;/p&gt; *      &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; *      &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; *      &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; *      &lt;p&gt;Preview icon is delivered using a FLUTE » file delivery session.&lt;/p&gt; *      &lt;p&gt;This test cases uses the following SG » fragment instantiations and files found in the Appendix: *      &lt;ul&gt; *          &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt; *          &lt;li&gt;PreviewData fragment for service » fragment with PictureUri and AlternativeText element set to "TV Channel » icon"&lt;/li&gt; *          &lt;li&gt;Access fragment for previewData » fragment with reference to session description fragment&lt;/li&gt; *          &lt;li&gt;Session description fragment with » parameters for broadcast file delivery session&lt;/li&gt; *          &lt;li&gt;Image file "TvChannelIcon.jpg"&lt;/li&gt; *      &lt;/ul&gt; *      &lt;p&gt; *      &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; *      &lt;p&gt;Set the FLUTE session to contain an » FDT-instance with 'File'-element, where 'Content-Location= » "TvChannelIcon.jpg"'.&lt;/p&gt; *      &lt;p&gt;Test Procedure&lt;/p&gt; *      &lt;p&gt; *      &lt;ul&gt; *          &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using broadcast channel.&lt;/li&gt; *          &lt;li&gt;Setup the test tool to deliver the » image file "TvChannelIcon.jpg" using a file delivery session as described </pre>

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» by SDP.</li>
*           <li>Activate the BCAST application of
» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>
*           <ul>
*           <li>Terminal associates and displays
» either the icon "TvChannelIcon.jpg".or alternative text "TV Channel icon"
» with the service "TvChannel"</li>
*           </ul>
*           </p>
* @verdict
*           pass in case the client pass the conformance
» test.
* @verdict
*           fail in case the client does not pass the
» conformance test.
* @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_106() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();

    // preamble
    var UInt v_TransferId := 1;
    f_main_data_startFileTransfer(v_TransferId,
» PX_SDP_FILE_DELIVERY, {{fileRef := PX_PICTURE_ID}});
    f_startCommonUtPreamble();

    // test behavior
    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_Version := 1;

    f_main_ctrl_InitStartEndTime(c_one_hour,c_one
» _hour);
    // change 01 (WK 6): global time
» definition

```

```

» by SDP.</li>
*           <li>Activate the BCAST application of
» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>
*           <ul>
*           <li>Terminal associates and displays
» either the icon "TvChannelIcon.jpg".or alternative text "TV Channel icon"
» with the service "TvChannel"</li>
*           </ul>
*           </p>
* @verdict
*           pass in case the client pass the conformance
» test.
* @verdict
*           fail in case the client does not pass the
» conformance test.
* @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_106() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();

    // preamble
    var UInt v_TransferId := 1;
    f_main_data_startFileTransfer(v_TransferId,
» PX_SDP_FILE_DELIVERY, {{fileRef := PX_PICTURE_ID}});
    f_startCommonUtPreamble();

    // test behavior
    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_Version := 1;

    var UInt v_NTP_StartTime := 0;

    var UInt v_NTP_EndTime := 0;

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

			f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime);
	=		
» f_getDateTime(v_startTime); » time definition	var DateTime v_DateTime_StartTime := // change 01 (WK 6): global	<>	var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);
» f_getDateTime(v_endTime); » time definition	var DateTime v_DateTime_EndTime := // change 01 (WK 6): global		var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime);
	=		
» valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_FILE_DELIVERY ));	var SdpFragment v_Sdp :=		var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_FILE_DELIVERY ));
» valueof(m_def_AccessType_bc_sdp( f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));	var AccessTypeType v_AccessType :=		var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));
» valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_Version, "TvChannel", c_basicTv_ServiceType ));	var ServiceType v_Service :=		var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_Version, "TvChannel", c_basicTv_ServiceType ));
» valueof(m_def_picture_PreviewData( PX_SGDU_PREVIEW_DATA_ID, v_Version, PX_PICTURE_ID, {omit,"TV Channel icon"} ));	var PreviewDataType v_PreviewData :=		var PreviewDataType v_PreviewData := » valueof(m_def_picture_PreviewData( PX_SGDU_PREVIEW_DATA_ID, v_Version, PX_PICTURE_ID, {omit,"TV Channel icon"} ));
» valueof(m_def_Access( PX_SGDU_ACCESS_ID_PROG_1, v_Version, v_AccessType, PX_SGDU_SERVICE_ID, omit, c_class_SG	var AccessType v_Access :=		var AccessType v_Access := » valueof(m_def_Access( PX_SGDU_ACCESS_ID_PROG_1, v_Version, v_AccessType, PX_SGDU_SERVICE_ID, omit, c_class_SG

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addPreviewDataFragment(v_SGDU,
» v_PreviewData);

    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_Version, v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("TvChannel");
    f_main_ut_CheckPreview("icon
» TvChannelIcon.jpg or text ""TV Channel icon""");

    // postamble
    f_main_ut_PowerOff();
    f_main_data_stopFileTransfer(v_TransferId);

    f_cf_Broadcast_down();
    f_cf_Data_down();
    f_cf_UpperTester_down();
} // end test case TC_BCAST_ESG_CONF_106

/**
 *
 * @desc
 * <p>Service Guide with dual audio streams via
» broadcast channel</p>
 * <p>Test Case Description</p>
 * <p>with a terminal listening to the broadcast
» Service Guide Delivery Channel.</p>
 * <p>when the terminal receives Service Guide
» containing one content fragment associated with two access fragments
» pointing to different audio streams</p>
 * <p>then the terminal presents the information
» of the Service Guide and is capable of associating the content with both
» access fragments.</p>
 * <p>Specification Reference</p>
 * <p>[BCAST10-ESG] Section 5.5.1, 6.1.1, 7.2.1,
» Appendix C.3 (informative)</p>
 * <p>Preconditions</p>

```

```

    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addPreviewDataFragment(v_SGDU,
» v_PreviewData);

    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_Version, v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("TvChannel");
    f_main_ut_CheckPreview("icon
» TvChannelIcon.jpg or text ""TV Channel icon""");

    // postamble
    f_main_ut_PowerOff();
    f_main_data_stopFileTransfer(v_TransferId);

    f_cf_Broadcast_down();
    f_cf_Data_down();
    f_cf_UpperTester_down();
} // end test case TC_BCAST_ESG_CONF_106

/**
 *
 * @desc
 * <p>Service Guide with dual audio streams via
» broadcast channel</p>
 * <p>Test Case Description</p>
 * <p>with a terminal listening to the broadcast
» Service Guide Delivery Channel.</p>
 * <p>when the terminal receives Service Guide
» containing one content fragment associated with two access fragments
» pointing to different audio streams</p>
 * <p>then the terminal presents the information
» of the Service Guide and is capable of associating the content with both
» access fragments.</p>
 * <p>Specification Reference</p>
 * <p>[BCAST10 -ESG] Section 5.5.1, 6.1.1, 7.2.1,
» Appendix C.3 (informative)</p>
 * <p>Preconditions</p>

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

*      <p>Service guide cache of terminal is
» erased</p>
*      <p>Terminal is configured to listen to BCAST
» service guide announcements and delivery on the broadcast channel</p>
*      <p>Access point information for service guide
» entry point is configured in test tool</p>
*      <p>This test cases uses the following SG
» fragment instantiations and files found in the Appendix:
*      <ul>
*          <li>Service fragment with
» Name="TvChannel"</li>
*          <li>Content fragment with
» Name="Programme"</li>
*          <li>Schedule fragment for content
» fragment</li>
*          <li>Access fragment for schedule
» fragment with BroadcastServiceDelivery element and a reference to the
» "Programme_eng" session description fragment </li>
*          <li>Session Description fragment
» containing SDP for "Programme_eng"(audio language is English)</li>
*          <li>Second schedule fragment for
» content fragment</li>
*          <li>Access fragment for second schedule
» fragment with BroadcastServiceDelivery element and a reference to the
» "Programme_ger" session description fragment</li>
*          <li>Session Description fragment
» containing SDP for "Programme_ger"(audio language is German)</li>
*          <li>Terminal provides means for the user
» in selecting the audio language.</li>
*      </ul>
*      <p>
» the same Service fragment and are sent in the same service guide
» delivery</p>
*      <p>Test Procedure</p>
*      <p>
*          <ul>
*              <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
*              <li>Activate the BCAST application of
» the terminal</li>
*              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*              <li>Browse the SG in the terminal</li>
*              <li>Setup the test tool to stream video,
» English audio, and German audio via the broadcast channel as defined by the
» SDP fragments</li>

```

```

*      <p>Service guide cache of terminal is
» erased</p>
*      <p>Terminal is configured to listen to BCAST
» service guide announcements and delivery on the broadcast channel</p>
*      <p>Access point information for service guide
» entry point is configured in test tool</p>
*      <p>This test cases uses the following SG
» fragment instantiations and files found in the Appendix:
*      <ul>
*          <li>Service fragment with
» Name="TvChannel"</li>
*          <li>Content fragment with
» Name="Programme"</li>
*          <li>Schedule fragment for content
» fragment</li>
*          <li>Access fragment for schedule
» fragment with BroadcastServiceDelivery element and a reference to the
» "Programme_eng" session description fragment </li>
*          <li>Session Description fragment
» containing SDP for "Programme_eng"(audio language is English)</li>
*          <li>Second schedule fragment for
» content fragment</li>
*          <li>Access fragment for second schedule
» fragment with BroadcastServiceDelivery element and a reference to the
» "Programme_ger" session description fragment</li>
*          <li>Session Description fragment
» containing SDP for "Programme_ger"(audio language is German)</li>
*          <li>Terminal provides means for the user
» in selecting the audio language.</li>
*      </ul>
*      <p>
» the same Service fragment and are sent in the same service guide
» delivery</p>
*      <p>Test Procedure</p>
*      <p>
*          <ul>
*              <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
*              <li>Activate the BCAST application of
» the terminal</li>
*              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*              <li>Browse the SG in the terminal</li>
*              <li>Setup the test tool to stream video,
» English audio, and German audio via the broadcast channel as defined by the
» SDP fragments</li>

```



Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

*           <li>Select English as the audio language
» for "Programme"</li>
*           <li>Start viewing "Programme"</li>
*           <li>Select German as the audio language
» for "Programme"</li>
*           <li>Continue viewing "Programme" </li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible and audible
» to the end user after the delivery of the SG
*           <ul>
*           <li>There is a service "TvChannel"
» associated with a programme "Programme".</li>
*           <li>"Programme" can be viewed by the end
» user with English as the audio language.</li>
*           <li>"Programme" can be viewed by the end
» user with German as the audio language.</li>
*           </ul>
*           </p>
* @verdict
*   pass in case the client pass the conformance
» test.
* @verdict
*   fail in case the client does not pass the
» conformance test.
* @verdict
*   inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_107() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_main_data_startStreamingFile(0,
» PX_FILE_VIDEO_PROG1, PX_SDP_VIDEO_PROG_1);
    f_main_data_startStreamingFile(1,
» PX_FILE_AUDIO_ENG, PX_SDP_AUDIO_ENG);
    f_main_data_startStreamingFile(2,
» PX_FILE_AUDIO_GER, PX_SDP_AUDIO_GER);

    f_startCommonUtPreamble();

    // test behavior

```

```

*           <li>Select English as the audio language
» for "Programme"</li>
*           <li>Start viewing "Programme"</li>
*           <li>Select German as the audio language
» for "Programme"</li>
*           <li>Continue viewing "Programme" </li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible and audible
» to the end user after the delivery of the SG
*           <ul>
*           <li>There is a service "TvChannel"
» associated with a programme "Programme".</li>
*           <li>"Programme" can be viewed by the end
» user with English as the audio language.</li>
*           <li>"Programme" can be viewed by the end
» user with German as the audio language.</li>
*           </ul>
*           </p>
* @verdict
*   pass in case the client pass the conformance
» test.
* @verdict
*   fail in case the client does not pass the
» conformance test.
* @verdict
*   inconc in case a guard timer expires.
*/
testcase TC_BCAST_ESG_CONF_107() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_main_data_startStreamingFile(0,
» PX_FILE_VIDEO_PROG1, PX_SDP_VIDEO_PROG_1);
    f_main_data_startStreamingFile(1,
» PX_FILE_AUDIO_ENG, PX_SDP_AUDIO_ENG);
    f_main_data_startStreamingFile(2,
» PX_FILE_AUDIO_GER, PX_SDP_AUDIO_GER);

    f_startCommonUtPreamble();

    // test behavior

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_Version := 1; </pre>		<pre> var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_Version := 1; </pre>
<pre> » f_main_ctrl_InitStartTime(c_one_hour,c_one » definition // change 01 (WK 6): global time </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); » time definition // change 01 (WK 6): global var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime); » global time definition // change 01 (WK 6): </pre>	=	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp1 := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_AUDIO_ENG ));  var SdpFragment v_Sdp2 := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_2_REF_ID, PX_SDP_AUDIO_GER ));  var AccessTypeType v_AccessType1 := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));  var AccessTypeType v_AccessType2 := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_2_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_Version, </pre>	=	<pre> var SdpFragment v_Sdp1 := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_AUDIO_ENG ));  var SdpFragment v_Sdp2 := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_2_REF_ID, PX_SDP_AUDIO_GER ));  var AccessTypeType v_AccessType1 := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));  var AccessTypeType v_AccessType2 := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_2_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_Version, </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre>                 "TvChannel",                 c_basicTv_ServiceType             ));              var ContentType v_Content := » valueof(m_def_Content(                 PX_SGDU_CONTENT_ID_PROG_1,                 v_Version,                 "Programme",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule1 := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_Version,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_startTime,                 v_endTime             )); </pre>		<pre>                 "TvChannel",                 c_basicTv_ServiceType             ));              var ContentType v_Content := » valueof(m_def_Content(                 PX_SGDU_CONTENT_ID_PROG_1,                 v_Version,                 "Programme",                 PX_SGDU_SERVICE_ID,                 v_DateTime_StartTime,                 v_DateTime_EndTime             ));              var ScheduleType v_Schedule1 := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_1,                 v_Version,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_1,                 v_NTP_StartTime,                 v_NTP_EndTime             )); </pre>
<pre> » // change 01 (WK 6): global time definition                 v_endTime » // change 01 (WK 6): global time definition             )); </pre>	<>	<pre>                 v_NTP_StartTime,                 v_NTP_EndTime             )); </pre>
<pre>             var ScheduleType v_Schedule2 := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_2,                 v_Version,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_2,                 v_startTime,                 v_endTime             )); </pre>	=	<pre>             var ScheduleType v_Schedule2 := » valueof(m_def_Schedule(                 PX_SGDU_SCHEDULE_ID_PROG_2,                 v_Version,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_CONTENT_ID_PROG_2,                 v_NTP_StartTime,                 v_NTP_EndTime             )); </pre>
<pre> » // change 01 (WK 6): global time definition                 v_endTime » // change 01 (WK 6): global time definition             )); </pre>	<>	<pre>                 v_NTP_StartTime,                 v_NTP_EndTime             )); </pre>
<pre>             var AccessType v_Access1 := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_Version,                 v_AccessTyp1,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             )); </pre>	=	<pre>             var AccessType v_Access1 := » valueof(m_def_Access(                 PX_SGDU_ACCESS_ID_PROG_1,                 v_Version,                 v_AccessTyp1,                 PX_SGDU_SERVICE_ID,                 PX_SGDU_SCHEDULE_ID_PROG_1,                 c_class_SG             )); </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

    ));

    var AccessType v_Access2 :=

»   valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_2,
        v_Version,
        v_AccessType2,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_SCHEDULE_ID_PROG_2,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addScheduleFragment(v_SGDU, v_Schedule1);
    f_addAccessFragment(v_SGDU, v_Access1);
    f_addScheduleFragment(v_SGDU, v_Schedule2);
    f_addAccessFragment(v_SGDU, v_Access2);
    f_addSdpFragment(v_SGDU, v_Version, v_Sdp1);
    f_addSdpFragment(v_SGDU, v_Version, v_Sdp2);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,

»   omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
        f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("TvChannel");
    f_main_ut_SelectService("TvChannel");
    f_main_ut_CheckContent("Programme",

»   v_DateTime_StartTime, v_DateTime_EndTime);

    f_main_ut_SelectLanguage(e_audio, "English");
    f_main_ut_UseService("TvChannel");
    f_main_ut_CheckLanguage("Programme", e_audio,

»   "English");

    f_main_ut_SelectLanguage(e_audio, "German");
    f_main_ut_UseService("TvChannel");
    f_main_ut_CheckLanguage("Programme", e_audio,

»   "German");

    // postamble
    f_main_ut_PowerOff();
    f_main_data_stopStreaming(0);
    f_main_data_stopStreaming(1);

```

```

    ));

    var AccessType v_Access2 :=

»   valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_2,
        v_Version,
        v_AccessType2,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_SCHEDULE_ID_PROG_2,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addScheduleFragment(v_SGDU, v_Schedule1);
    f_addAccessFragment(v_SGDU, v_Access1);
    f_addScheduleFragment(v_SGDU, v_Schedule2);
    f_addAccessFragment(v_SGDU, v_Access2);
    f_addSdpFragment(v_SGDU, v_Version, v_Sdp1);
    f_addSdpFragment(v_SGDU, v_Version, v_Sdp2);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,

»   omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
        f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("TvChannel");
    f_main_ut_SelectService("TvChannel");
    f_main_ut_CheckContent("Programme",

»   v_DateTime_StartTime, v_DateTime_EndTime);

    f_main_ut_SelectLanguage(e_audio, "English");
    f_main_ut_UseService("TvChannel");
    f_main_ut_CheckLanguage("Programme", e_audio,

»   "English");

    f_main_ut_SelectLanguage(e_audio, "German");
    f_main_ut_UseService("TvChannel");
    f_main_ut_CheckLanguage("Programme", e_audio,

»   "German");

    // postamble
    f_main_ut_PowerOff();
    f_main_data_stopStreaming(0);
    f_main_data_stopStreaming(1);

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> f_main_data_stopStreaming(2);  f_cf_Broadcast_down(); f_cf_UpperTester_down(); } // end test case TC_BCAST_ESG_CONF_107  /**  *  * @desc  * &lt;p&gt;Service guide with purchase information&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt;  * &lt;p&gt;with a terminal listening to the broadcast » Service Guide Delivery Channel.&lt;/p&gt;  * &lt;p&gt;when the terminal receives Service Guide » containing purchase item, purchase channel and purchase data fragments&lt;/p&gt;  * &lt;p&gt;then the terminal presents the information » of the Service Guide&lt;/p&gt;  * &lt;p&gt;Specification Reference&lt;/p&gt; </pre>		<pre> f_main_data_stopStreaming(2);  f_cf_Broadcast_down(); f_cf_UpperTester_down(); } // end test case TC_BCAST_ESG_CONF_107  /**  *  * @desc  * &lt;p&gt;Service guide with purchase information&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt;  * &lt;p&gt;with a terminal listening to the broadcast » Service Guide Delivery Channel.&lt;/p&gt;  * &lt;p&gt;when the terminal receives Service Guide » containing purchase item, purchase channel and purchase data fragments&lt;/p&gt;  * &lt;p&gt;then the terminal presents the information » of the Service Guide&lt;/p&gt;  * &lt;p&gt;Specification Reference&lt;/p&gt; </pre>
<pre> * &lt;p&gt;[BCAST10-ESG] Section 5.1.2.6, 5.1.2.7, » 5.1.2.8, 5.5.1, 6.1.1&lt;br&gt;[BCAST10-sServices]&lt;/p&gt; </pre>	<>	<pre> * &lt;p&gt;[BCAST10 -ESG] Section 5.1.2.6, 5.1.2.7, » 5.1.2.8, 5.5.1, 6.1.1&lt;br&gt;[BCAST10-Services]&lt;/p&gt; </pre>
<pre> * &lt;p&gt;Preconditions&lt;/p&gt; * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix: * &lt;ul&gt; * &lt;li&gt;Service fragment with » Name="PayTvChannel"&lt;/li&gt; * &lt;li&gt;Content fragment with » Name="Programme", and StartTime and EndTime elements indicating values » after the time of test&lt;/li&gt; * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt; * &lt;li&gt;Access fragment for schedule » fragment with BroadcastServiceDelivery, KeyManagement, and EncryptionType » elements and a reference to the session description fragment&lt;/li&gt; * &lt;li&gt;Session Description fragment » containing SDP for "Programme"&lt;/li&gt; * &lt;li&gt;PurchaseItem fragment for service » fragment&lt;/li&gt; * &lt;li&gt;PurchaseData fragment for » purchaseItem fragment with PriceInfo and Description "Discount" </pre>	=	<pre> * &lt;p&gt;Preconditions&lt;/p&gt; * &lt;p&gt;Service guide cache of terminal is » erased&lt;/p&gt; * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel&lt;/p&gt; * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt; * &lt;p&gt;This test cases uses the following SG » fragment instantiations found in the Appendix: * &lt;ul&gt; * &lt;li&gt;Service fragment with » Name="PayTvChannel"&lt;/li&gt; * &lt;li&gt;Content fragment with » Name="Programme", and StartTime and EndTime elements indicating values » after the time of test&lt;/li&gt; * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt; * &lt;li&gt;Access fragment for schedule » fragment with BroadcastServiceDelivery, KeyManagement, and EncryptionType » elements and a reference to the session description fragment&lt;/li&gt; * &lt;li&gt;Session Description fragment » containing SDP for "Programme"&lt;/li&gt; * &lt;li&gt;PurchaseItem fragment for service » fragment&lt;/li&gt; * &lt;li&gt;PurchaseData fragment for » purchaseItem fragment with PriceInfo and Description "Discount" </pre>

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

» elements</li>
    *           <li>PurchaseChannel fragment for
» purchaseItem fragment with PurchaseURL pointing to a subscription site</li>
    *           </ul>
    *           </p>
    *           <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
    *           <p>Test Procedure</p>
    *           <p>
    *           <ul>
    *           <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
    *           <li>Activate the BCAST application of
» the terminal</li>
    *           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
    *           <li>Browse the SG in the terminal</li>
    *           </ul>
    *           </p>
    *           <p>Pass-Criteria</p>
    *           <p>The following should be visible on terminal
» to the end user after the delivery of the SG
    *           <ul>
    *           <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information </li>
    *           </ul>
    *           </p>
    *           @verdict
    *           pass in case the client pass the conformance
» test.
    *           @verdict
    *           fail in case the client does not pass the
» conformance test.
    *           @verdict
    *           inconc in case a guard timer expires.
    */
    testcase TC_BCAST_ESG_CONF_108() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();

```

```

» elements</li>
    *           <li>PurchaseChannel fragment for
» purchaseItem fragment with PurchaseURL pointing to a subscription site</li>
    *           </ul>
    *           </p>
    *           <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
    *           <p>Test Procedure</p>
    *           <p>
    *           <ul>
    *           <li>Set up the test tool to produce
» BCAST service guide announcement and delivery using broadcast channel.</li>
    *           <li>Activate the BCAST application of
» the terminal</li>
    *           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
    *           <li>Browse the SG in the terminal</li>
    *           </ul>
    *           </p>
    *           <p>Pass-Criteria</p>
    *           <p>The following should be visible on terminal
» to the end user after the delivery of the SG
    *           <ul>
    *           <li>The service "PayTvChannel" is
» displayed with "Programme" schedule from start to end time, available as
» "Discount" purchase, and price information </li>
    *           </ul>
    *           </p>
    *           @verdict
    *           pass in case the client pass the conformance
» test.
    *           @verdict
    *           fail in case the client does not pass the
» conformance test.
    *           @verdict
    *           inconc in case a guard timer expires.
    */
    testcase TC_BCAST_ESG_CONF_108() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();

```

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>		<pre> // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>
<pre> » f_main_ctrl_InitStartEndTime(c_one_hour,c_one » _hour); » definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartEndTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); » definition // change 01 (WK 6): global time  var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime); » time definition // change 01 (WK 6): global </pre>	=	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));  var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo( 1, PX_MONETARY_PRICE, PX_CURRENCY ));  var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_FragmentVersion, "PayTvChannel", c_basicTv_ServiceType )); </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment( PX_SDP_VIDEO_PROG_1_REF_ID, PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER), PX_SDP_VIDEO_PROG_1_REF_ID ));  var PriceInfoType v_PriceInfo := » valueof(m_PriceInfo( 1, PX_MONETARY_PRICE, PX_CURRENCY ));  var ServiceType v_Service := » valueof(m_def_Service( PX_SGDU_SERVICE_ID, v_FragmentVersion, "PayTvChannel", c_basicTv_ServiceType )); </pre>

Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

<pre> var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_FragmentVersion,     "Programme",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_FragmentVersion,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_startTime,     v_endTime )); </pre>		<pre> var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_FragmentVersion,     "Programme",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_FragmentVersion,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_NTP_StartTime,     v_NTP_EndTime )); </pre>
<pre> » // change 01 (WK 6): global time definition v_endTime » // change 01 (WK 6): global time definition </pre>	<>	<pre> v_NTP_StartTime, v_NTP_EndTime </pre>
<pre> ));  // TODO: Omitted here is setting of KMS value » (see open issues) but omission may be ok in this test case var AccessType v_Access := » valueof(m_def_Access(     PX_SGDU_ACCESS_ID_PROG_1,     v_FragmentVersion,     v_AccessType,     PX_SGDU_SERVICE_ID,     PX_SGDU_SCHEDULE_ID_PROG_1,     c_class_SG ));  var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(     PX_SGDU_PURCHASE_ITEM_ID,     v_FragmentVersion,     PX_SGDU_PURCHASE_DATA_ID,     PX_SGDU_SERVICE_ID,     "PurchaseItem1" ));  var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(     PX_SGDU_PURCHASE_CHANNEL_ID,     v_FragmentVersion, </pre>	=	<pre> ));  // TODO: Omitted here is setting of KMS value » (see open issues) but omission may be ok in this test case var AccessType v_Access := » valueof(m_def_Access(     PX_SGDU_ACCESS_ID_PROG_1,     v_FragmentVersion,     v_AccessType,     PX_SGDU_SERVICE_ID,     PX_SGDU_SCHEDULE_ID_PROG_1,     c_class_SG ));  var PurchaseItemType v_PurchaseItem := » valueof(m_def_PurchaseItem(     PX_SGDU_PURCHASE_ITEM_ID,     v_FragmentVersion,     PX_SGDU_PURCHASE_DATA_ID,     PX_SGDU_SERVICE_ID,     "PurchaseItem1" ));  var PurchaseChannelType v_PurchaseChannel := » valueof(m_def_PurchaseChannel(     PX_SGDU_PURCHASE_CHANNEL_ID,     v_FragmentVersion, </pre>



Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

        "PurchaseChannel",
        PX_PURCHASE_URL
    ));

    var PurchaseDataType v_PurchaseData :=
» valueof(m_def_PurchaseData(
        PX_SGDU_PURCHASE_DATA_ID,
        v_FragmentVersion,
        {{omit,"Discount"}}},
        PX_SGDU_PURCHASE_ITEM_ID,
        PX_SGDU_PURCHASE_CHANNEL_ID,
        v_PriceInfo
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addPurchaseItemFragment(v_SGDU,
» v_PurchaseItem);
    f_addPurchaseChannelFragment(v_SGDU,
» v_PurchaseChannel);
    f_addPurchaseDataFragment(v_SGDU,
» v_PurchaseData);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("PayTvChannel");
        f_main_ut_SelectService("PayTvChannel");
        f_main_ut_CheckContent("Programme",
» v_DateTime_StartTime, v_DateTime_EndTime);
        f_main_ut_CheckPurchaseInfo( "Discount" );

        // postamble
        f_main_ut_PowerOff();

        f_cf_Broadcast_down();
        f_cf_UpperTester_down();
    }// end test case TC_BCAST_ESG_CONF_108

```

```

        "PurchaseChannel",
        PX_PURCHASE_URL
    ));

    var PurchaseDataType v_PurchaseData :=
» valueof(m_def_PurchaseData(
        PX_SGDU_PURCHASE_DATA_ID,
        v_FragmentVersion,
        {{omit,"Discount"}}},
        PX_SGDU_PURCHASE_ITEM_ID,
        PX_SGDU_PURCHASE_CHANNEL_ID,
        v_PriceInfo
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addPurchaseItemFragment(v_SGDU,
» v_PurchaseItem);
    f_addPurchaseChannelFragment(v_SGDU,
» v_PurchaseChannel);
    f_addPurchaseDataFragment(v_SGDU,
» v_PurchaseData);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("PayTvChannel");
        f_main_ut_SelectService("PayTvChannel");
        f_main_ut_CheckContent("Programme",
» v_DateTime_StartTime, v_DateTime_EndTime);
        f_main_ut_CheckPurchaseInfo( "Discount" );

        // postamble
        f_main_ut_PowerOff();

        f_cf_Broadcast_down();
        f_cf_UpperTester_down();
    }// end test case TC_BCAST_ESG_CONF_108

```

## Datei: AtsBCast\_ServiceGuideTests.ttcn (Fortsetzung)

```

        } // end group clientConformanceTestCases
    } // end group bcastConformanceTestCases
}

```

```

        } // end group clientConformanceTestCases
    } // end group bcastConformanceTestCases
}

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn

```

/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies pilot tests for BCast service interaction.
 */
module AtsBCast_ServiceInteractionTests {
    import from LibBCast_Interface all;
    import from LibBCast_ServicePrimitives_TypesAndValues all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_BCastNetworkData_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibBCast_Common_Templates all;
    import from LibCommon_BasicTypesAndValues all;
    import from AtsBCast_ModuleParameters all;
    import from LibBCast_ModuleParameters all;
    import from AtsBCast_TestConfiguration_Functions all;
    import from AtsBCast_TestSystem all;

    group bcastConformanceTestCases {
        group clientConformanceTestCases {

            /**
             *
             * @desc
             *   <p>Service guide with XHTML interactivity
            » document via broadcast channel (optional)</p>
             *   <p>Test Case Description</p>
             *   <p>with a terminal listening to the broadcast
            » Service Guide Delivery Channel.</p>
             *   <p>when the terminal receives Service Guide
            » containing interactivity data for XHTML interactivity</p>
             *   <p>then terminal starts an application with
            » XHTML data </p>
             *   <p>Specification Reference</p>
             *   <p>[BCAST10-Services] Section 5.3.6, 5.3.6.1.2,

```

```

= /**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies pilot tests for BCast service interaction.
 */
module AtsBCast_ServiceInteractionTests {
    import from LibBCast_Interface all;
    import from LibBCast_ServicePrimitives_TypesAndValues all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_BCastNetworkData_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibBCast_Common_Templates all;
    import from LibCommon_BasicTypesAndValues all;
    import from AtsBCast_ModuleParameters all;
    import from LibBCast_ModuleParameters all;
    import from AtsBCast_TestConfiguration_Functions all;
    import from AtsBCast_TestSystem all;

    group bcastConformanceTestCases {
        group clientConformanceTestCases {

            /**
             *
             * @desc
             *   <p>Service guide with XHTML interactivity
            » document via broadcast channel (optional)</p>
             *   <p>Test Case Description</p>
             *   <p>with a terminal listening to the broadcast
            » Service Guide Delivery Channel.</p>
             *   <p>when the terminal receives Service Guide
            » containing interactivity data for XHTML interactivity</p>
             *   <p>then terminal starts an application with
            » XHTML data </p>
             *   <p>Specification Reference</p>
             *   <p>[BCAST10-Services] Section 5.3.6, 5.3.6.1.2,

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

» 5.3.6.1.5.</p>
* <p>Preconditions</p>
* <p>Service guide cache of terminal is
» erased</p>
* <p>Terminal is configured to listen to BCAST
» service guide announcements and delivery on the broadcast channel</p>
* <p>Access point information for service guide
» entry point is configured in test tool</p>
* <p>Set up the test tool for Service Guide
» delivery to provide stream delivery information for "Programm1".</p>
* <p>Configure terminal to start a, e.g.,
» browser, application with XHTML interactivity data</p>
* <p>This test cases uses the following SG
» fragment instantiations found in the Appendix:
* <ul>
* <li>Service fragment with
» Name="TvChannel"</li>
* <li>Content fragment with
» Name="Programm1"</li>
* <li>InteractivityData fragment for
» content fragment with interactivityType set to "vote-xhtml"</li>
* <li>Schedule fragment for interactivity
» data fragment</li>
* <li>Access fragment for schedule
» fragment and referencing the session description fragment</li>
* <li>SDP for file delivery session of
» interactivity media document and XHTML file </li>
* </ul>
* </p>
* <p>Set the FLUTE session to transport the
» InteractivityMediaDocument "iamdl.xml" with TOI="1" and the file
» 'Vote.xhtml' with TOI="2".</p>
* <p>Set the FLUTE session to contain an
» FDT-Instance with two 'File'-elements;
* <ul>
* <li>one with attributes
» 'Content-Location="iamdl.xml"', 'Content-Type=
» "application/vnd.oma.bcast.imd+xml"' and 'TOI="1"'</li>
* <li>and another with attributes
» 'Content-Location="Vote.xhtml"', 'Content-Type="text/html"' and
» 'TOI="2"'</li>
* </ul>
* </p>
* <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide delivery.
* <ul>
* <li>XHTML file "Vote.xhtml"</li>

```

```

» 5.3.6.1.5.</p>
* <p>Preconditions</p>
* <p>Service guide cache of terminal is
» erased</p>
* <p>Terminal is configured to listen to BCAST
» service guide announcements and delivery on the broadcast channel</p>
* <p>Access point information for service guide
» entry point is configured in test tool</p>
* <p>Set up the test tool for Service Guide
» delivery to provide stream delivery information for "Programm1".</p>
* <p>Configure terminal to start a, e.g.,
» browser, application with XHTML interactivity data</p>
* <p>This test cases uses the following SG
» fragment instantiations found in the Appendix:
* <ul>
* <li>Service fragment with
» Name="TvChannel"</li>
* <li>Content fragment with
» Name="Programm1"</li>
* <li>InteractivityData fragment for
» content fragment with interactivityType set to "vote-xhtml"</li>
* <li>Schedule fragment for interactivity
» data fragment</li>
* <li>Access fragment for schedule
» fragment and referencing the session description fragment</li>
* <li>SDP for file delivery session of
» interactivity media document and XHTML file </li>
* </ul>
* </p>
* <p>Set the FLUTE session to transport the
» InteractivityMediaDocument "iamdl.xml" with TOI="1" and the file
» 'Vote.xhtml' with TOI="2".</p>
* <p>Set the FLUTE session to contain an
» FDT-Instance with two 'File'-elements;
* <ul>
* <li>one with attributes
» 'Content-Location="iamdl.xml"', 'Content-Type=
» "application/vnd.oma.bcast.imd+xml"' and 'TOI="1"'</li>
* <li>and another with attributes
» 'Content-Location="Vote.xhtml"', 'Content-Type="text/html"' and
» 'TOI="2"'</li>
* </ul>
* </p>
* <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide delivery.
* <ul>
* <li>XHTML file "Vote.xhtml"</li>

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

*          <li>InteractivityMediaDocument with
» MediaObject Set element pointing to XHTML file</li>
*          </ul>
*          </p>
*          <p></p>
*          <p>Test Procedure</p>
*          <p>
*          <ul>
*          <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*          <li>Setup the test tool to deliver the
» interactivity media document and xhtml file "Vote.xhtml" using a file
» delivery session as described by SDP.</li>
*          <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*          <li>Browse the SG on the terminal</li>
*          <li>Select "TvChannel" and interactivity
» "vote-xhtml" </li>
*          </ul>
*          </p>
*          <p>Pass-Criteria</p>
*          <p>The following should be visible to the end
» user after the delivery of the SG
*          <ul>
*          <li>The terminal should associate the
» service "TvChannel" with "Programmel" and interactivity "vote-xhtml"</li>
*          <li>The application presents the XHTML
» content to end user</li>
*          </ul>
*          </p>
*          @verdict
*          pass in case the client pass the conformance
» test.
*          @verdict
*          fail in case the client does not pass the
» conformance test.
*          @verdict
*          inconc in case a guard timer expires.
*/
testcase TC_BCAST_INTER_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();

```

```

*          <li>InteractivityMediaDocument with
» MediaObject Set element pointing to XHTML file</li>
*          </ul>
*          </p>
*          <p></p>
*          <p>Test Procedure</p>
*          <p>
*          <ul>
*          <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*          <li>Setup the test tool to deliver the
» interactivity media document and xhtml file "Vote.xhtml" using a file
» delivery session as described by SDP.</li>
*          <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*          <li>Browse the SG on the terminal</li>
*          <li>Select "TvChannel" and interactivity
» "vote-xhtml" </li>
*          </ul>
*          </p>
*          <p>Pass-Criteria</p>
*          <p>The following should be visible to the end
» user after the delivery of the SG
*          <ul>
*          <li>The terminal should associate the
» service "TvChannel" with "Programmel" and interactivity "vote-xhtml"</li>
*          <li>The application presents the XHTML
» content to end user</li>
*          </ul>
*          </p>
*          @verdict
*          pass in case the client pass the conformance
» test.
*          @verdict
*          fail in case the client does not pass the
» conformance test.
*          @verdict
*          inconc in case a guard timer expires.
*/
testcase TC_BCAST_INTER_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();

```

Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

<pre> var ObjectType v_obj := » valueof(m_def_Object(PX_XHTML_FILE_ID, PX_XHTML_CONTENT_TYPE, omit, omit)); //var MediaObjectSetType v_objSet := » valueof(m_def_mediaObjectSet("application/x-gzip", PX_MEDIA_OBJECT_SET_URI, » {v_obj})); // Change No. 4 from OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review var MediaObjectSetType v_objSet := » valueof(m_def_mediaObjectSet("application/xhtml+xml", » PX_MEDIA_OBJECT_SET_URI, {v_obj})); var MediaObjectGroupType v_objGroup := » valueof(m_def_MediaObjectGroup(PX_MEDIA_DOCUMENT_GROUP_ID, true, » {v_objSet})); var InteractivityMediaDocumentType v_doc := » valueof(m_def_InteractivityMediaDocument(PX_MEDIA_DOCUMENT_GROUP_ID, 0, » PX_MEDIA_DOCUMENT_ID, 0, {v_objGroup}));  var TransferFileList v_FileList :={ { InteractivityMediaDocument := » v_doc }, { fileRef := PX_XHTML_FILE_ID } };  // preamble var UInt v_TransferId := 1; f_main_data_startFileTransfer(v_TransferId, » PX_SDP_FILE_DELIVERY, v_FileList); f_startCommonUtPreamble();  // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>	<>	<pre> var ObjectType v_obj := » valueof(m_def_Object(PX_XHTML_FILE_ID, PX_XHTML_CONTENT_TYPE, omit, omit)); //var MediaObjectSetType v_objSet := » valueof(m_def_mediaObjectSet("application/x-gzip", PX_MEDIA_OBJECT_SET_URI, » {v_obj})); // Change No. 4 from OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review var MediaObjectSetType v_objSet := » valueof(m_def_mediaObjectSet("application/xhtml+xml", » PX_MEDIA_OBJECT_SET_URI, {v_obj})); var MediaObjectGroupType v_objGroup := » valueof(m_def_MediaObjectGroup(PX_MEDIA_DOCUMENT_GROUP_ID, true, » {v_objSet})); var InteractivityMediaDocumentType v_doc := » valueof(m_def_InteractivityMediaDocument(PX_MEDIA_DOCUMENT_GROUP_ID, 0, » PX_MEDIA_DOCUMENT_ID, 0, {v_objGroup}));  var TransferFileList v_FileList :={ { InteractivityMediaDocument := » v_doc }, { fileRef := PX_XHTML_FILE_ID } };  // preamble var UInt v_TransferId := 1; f_main_data_startFileTransfer(v_TransferId, » PX_SDP_FILE_DELIVERY, v_FileList); f_startCommonUtPreamble();  // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>
<pre> f_main_ctrl_InitStartTime(c_one_hour,c_one » _hour); // change 01 (WK 6): global time » definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); // change 01 (WK 6): global » time definition var DateTime v_DateTime_EndTime := </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := </pre>

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```
» f_getDateTime(v_endTime);           // change 01 (WK 6): global
» time definition
```

```

                                var SdpFragment v_Sdp :=
» valueof(m_def_SdpFragment(
                                PX_SDP_VIDEO_PROG_1_REF_ID,
                                PX_SDP_FILE_DELIVERY
                                ));

                                var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                                PX_SDP_VIDEO_PROG_1_REF_ID
                                ));

                                var ServiceType v_Service :=
» valueof(m_def_Service(
                                PX_SGDU_SERVICE_ID,
                                v_FragmentVersion,
                                "TvChannel",
                                c_basicTv_ServiceType
                                ));

                                var ContentType v_Content :=
» valueof(m_def_Content(
                                PX_SGDU_CONTENT_ID_PROG_1,
                                v_FragmentVersion,
                                "Programmel",
                                PX_SGDU_SERVICE_ID,
                                v_DateTime_StartTime,
                                v_DateTime_EndTime
                                ));

                                var InteractivityDataType v_InterData :=
» valueof(m_def_InteractivityData(
                                PX_SGDU_INTERACTIVITY_DATA_ID,
                                v_FragmentVersion,
                                PX_PRELISTEN_INDICATOR,
                                PX_MEDIA_DOCUMENT_GROUP_ID,
                                PX_SGDU_SERVICE_ID
                                ));

                                var ScheduleType v_Schedule :=
» valueof(m_def_inter_Schedule(
                                PX_SGDU_SCHEDULE_ID_PROG_1,
                                v_FragmentVersion,
```

```
» f_getDateTime(v_NTP_EndTime);
```

```

                                var SdpFragment v_Sdp :=
» valueof(m_def_SdpFragment(
                                PX_SDP_VIDEO_PROG_1_REF_ID,
                                PX_SDP_FILE_DELIVERY
                                ));

                                var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                                PX_SDP_VIDEO_PROG_1_REF_ID
                                ));

                                var ServiceType v_Service :=
» valueof(m_def_Service(
                                PX_SGDU_SERVICE_ID,
                                v_FragmentVersion,
                                "TvChannel",
                                c_basicTv_ServiceType
                                ));

                                var ContentType v_Content :=
» valueof(m_def_Content(
                                PX_SGDU_CONTENT_ID_PROG_1,
                                v_FragmentVersion,
                                "Programmel",
                                PX_SGDU_SERVICE_ID,
                                v_DateTime_StartTime,
                                v_DateTime_EndTime
                                ));

                                var InteractivityDataType v_InterData :=
» valueof(m_def_InteractivityData(
                                PX_SGDU_INTERACTIVITY_DATA_ID,
                                v_FragmentVersion,
                                PX_PRELISTEN_INDICATOR,
                                PX_MEDIA_DOCUMENT_GROUP_ID,
                                PX_SGDU_SERVICE_ID
                                ));

                                var ScheduleType v_Schedule :=
» valueof(m_def_inter_Schedule(
                                PX_SGDU_SCHEDULE_ID_PROG_1,
                                v_FragmentVersion,
```

Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

        PX_SGDU_SERVICE_ID,
        PX_SGDU_INTERACTIVITY_DATA_ID
    ));

    var AccessType v_Access :=
» valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_1,
        v_FragmentVersion,
        v_AccessType,
        PX_SGDU_SERVICE_ID,
        omit,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addInteractivityDataFragment(v_SGDU,
» v_InterData);

    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("TvChannel");
    f_main_ut_SelectService("TvChannel");
    f_main_ut_CheckInteractivity("vote-xhtml");
    f_main_ut_SelectInteractivity("vote-xhtml");
    f_main_ut_CheckBrowser( PX_XHTML_FILE_ID );

    // postamble
    f_main_ut_PowerOff();
    f_main_data_stopFileTransfer(v_TransferId);

    f_cf_Broadcast_down();
    f_cf_Data_down();
    f_cf_UpperTester_down();
} // end test case TC_BCAST_INTER_CONF_101

/** @desc

```

```

        PX_SGDU_SERVICE_ID,
        PX_SGDU_INTERACTIVITY_DATA_ID
    ));

    var AccessType v_Access :=
» valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_1,
        v_FragmentVersion,
        v_AccessType,
        PX_SGDU_SERVICE_ID,
        omit,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addInteractivityDataFragment(v_SGDU,
» v_InterData);

    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
    }
    f_main_ut_CheckService("TvChannel");
    f_main_ut_SelectService("TvChannel");
    f_main_ut_CheckInteractivity("vote-xhtml");
    f_main_ut_SelectInteractivity("vote-xhtml");
    f_main_ut_CheckBrowser( PX_XHTML_FILE_ID );

    // postamble
    f_main_ut_PowerOff();
    f_main_data_stopFileTransfer(v_TransferId);

    f_cf_Broadcast_down();
    f_cf_Data_down();
    f_cf_UpperTester_down();
} // end test case TC_BCAST_INTER_CONF_101

/** @desc

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

*      <p>Service guide with SMS interactivity
» document via broadcast channel (optional)</p>
*      <p>Test Case Description</p>
*      <p>with a terminal listening to the Service
» Guide Delivery Channel.</p>
*      <p>when the terminal receives Service Guide
» containing interactivity data for SMS interactivity</p>
*      <p>then the terminal sends an SMS</p>
*      <p>Specification Reference</p>
*      <p>[BCAST10-Services] Section 5.3.6,
» 5.3.6.1.6.</p>
*      <p>Preconditions</p>
*      <p>Service guide cache of terminal is
» erased</p>
*      <p>Terminal is configured to listen to BCAST
» service guide announcements and delivery on the broadcast channel</p>
*      <p>Access point information for service guide
» entry point is configured in test tool</p>
*      <p>This test cases uses the following SG
» fragment instantiations found in the Appendix:
*      <ul>
*          <li>Service fragment with
» Name="TvChannel"</li>
*          <li>Content fragment with
» Name="Programme1" for service fragment</li>
*          <li>InteractivityData fragment for
» content fragment with interactivityType set to "vote-sms"</li>
*          <li>Schedule fragment for interactivity
» data fragment</li>
*          <li>Access fragment for schedule
» fragment and referencing session description fragment</li>
*          <li>SDP for file delivery session of
» interactivity media document</li>
*      </ul>
*      <p>
*      <p>Set the FLUTE session to transport the
» InteractivityMediaDocument "iamdl.xml" with TOI="1".</p>
*      <p>Set the FLUTE session to contain an
» FDT-Instance with a 'File'-element having attributes
» 'Content-Location="iamdl.xml"',
» 'Content-Type="application/vnd.oma.bcast.imd+xml"' and 'TOI="1"'.</p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide delivery.
*      <ul>
*          <li>InteractivityMediaDocument with
» SMSTemplate SelectChoice set to "a", "b", and "c" and smsURI pointing to a
» test tool URI</li>

```

```

*      <p>Service guide with SMS interactivity
» document via broadcast channel (optional)</p>
*      <p>Test Case Description</p>
*      <p>with a terminal listening to the Service
» Guide Delivery Channel.</p>
*      <p>when the terminal receives Service Guide
» containing interactivity data for SMS interactivity</p>
*      <p>then the terminal sends an SMS</p>
*      <p>Specification Reference</p>
*      <p>[BCAST10-Services] Section 5.3.6,
» 5.3.6.1.6.</p>
*      <p>Preconditions</p>
*      <p>Service guide cache of terminal is
» erased</p>
*      <p>Terminal is configured to listen to BCAST
» service guide announcements and delivery on the broadcast channel</p>
*      <p>Access point information for service guide
» entry point is configured in test tool</p>
*      <p>This test cases uses the following SG
» fragment instantiations found in the Appendix:
*      <ul>
*          <li>Service fragment with
» Name="TvChannel"</li>
*          <li>Content fragment with
» Name="Programme1" for service fragment</li>
*          <li>InteractivityData fragment for
» content fragment with interactivityType set to "vote-sms"</li>
*          <li>Schedule fragment for interactivity
» data fragment</li>
*          <li>Access fragment for schedule
» fragment and referencing session description fragment</li>
*          <li>SDP for file delivery session of
» interactivity media document</li>
*      </ul>
*      <p>
*      <p>Set the FLUTE session to transport the
» InteractivityMediaDocument "iamdl.xml" with TOI="1".</p>
*      <p>Set the FLUTE session to contain an
» FDT-Instance with a 'File'-element having attributes
» 'Content-Location="iamdl.xml"',
» 'Content-Type="application/vnd.oma.bcast.imd+xml"' and 'TOI="1"'.</p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide delivery.
*      <ul>
*          <li>InteractivityMediaDocument with
» SMSTemplate SelectChoice set to "a", "b", and "c" and smsURI pointing to a
» test tool URI</li>

```



Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

*      </ul>
*      </p>
*      <p>Test Procedure</p>
*      <p>
*          <ul>
*              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*              <li>Browse the SG on the terminal</li>
*              <li>Select "TvChannel" and interactivity
» "vote-sms" on the terminal </li>
*              <li>Select choice "b" on terminal.</li>
*          </ul>
*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the delivery of the SG
*      <ul>
*          <li>The terminal should associate the
» service "TvChannel" with "Programm1" and interactivity "vote-sms"</li>
*          <li>The terminal displays "a", "b", and
» "c" as available choices .</li>
*      </ul>
*      </p>
*      <p>The following should be observable for the
» test tool
*      <ul>
*          <li>The test tool receives an SMS from
» the terminal with the for choice "b"</li>
*      </ul>
*      </p>
*      @verdict
*          pass in case the client pass the conformance
» test.
*      @verdict
*          fail in case the client does not pass the
» conformance test.
*      @verdict
*          inconc in case a guard timer expires.
*/
testcase TC_BCAST_INTER_CONF_102() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();

```

```

*      </ul>
*      </p>
*      <p>Test Procedure</p>
*      <p>
*          <ul>
*              <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*              <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*              <li>Browse the SG on the terminal</li>
*              <li>Select "TvChannel" and interactivity
» "vote-sms" on the terminal </li>
*              <li>Select choice "b" on terminal.</li>
*          </ul>
*      </p>
*      <p>Pass-Criteria</p>
*      <p>The following should be visible to the end
» user after the delivery of the SG
*      <ul>
*          <li>The terminal should associate the
» service "TvChannel" with "Programm1" and interactivity "vote-sms"</li>
*          <li>The terminal displays "a", "b", and
» "c" as available choices .</li>
*      </ul>
*      </p>
*      <p>The following should be observable for the
» test tool
*      <ul>
*          <li>The test tool receives an SMS from
» the terminal with the for choice "b"</li>
*      </ul>
*      </p>
*      @verdict
*          pass in case the client pass the conformance
» test.
*      @verdict
*          fail in case the client does not pass the
» conformance test.
*      @verdict
*          inconc in case a guard timer expires.
*/
testcase TC_BCAST_INTER_CONF_102() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();

```

Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

<pre> f_cf_Data_up(); f_cf_UpperTester_up(); f_cf_SMS_up();  var SelectChoiceType v_SelectChoice_A := { » smsURI := PX_SMS_URI, ChoiceTexts := {{ text_ := "a" }}; var SelectChoiceType v_SelectChoice_B := { » smsURI := PX_SMS_URI, ChoiceTexts := {{ text_ := "b" }}; var SelectChoiceType v_SelectChoice_C := { » smsURI := PX_SMS_URI, ChoiceTexts := {{ text_ := "c" }};  var SMSTemplateType v_sms := » valueof(m_def_SmsTemplate( {v_SelectChoice_A, v_SelectChoice_B, » v_SelectChoice_C} ));  var MediaObjectGroupType v_objGroup := » valueof(m_def_MediaObjectGroup(PX_MEDIA_DOCUMENT_GROUP_ID, true, omit)); v_objGroup.SMSTemplate := v_sms; var InteractivityMediaDocumentType v_doc := » valueof(m_def_InteractivityMediaDocument(PX_MEDIA_DOCUMENT_GROUP_ID, 0, » PX_MEDIA_DOCUMENT_ID, 0, {v_objGroup})); var TransferFileList v_FileList :={ { InteractivityMediaDocument := » v_doc } };  var UInt v_TransferId := 1; f_main_data_startFileTransfer(v_TransferId, » PX_SDP_FILE_DELIVERY, v_FileList);  // preamble f_startCommonUtPreamble();  // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>	<>	<pre> f_cf_Data_up(); f_cf_UpperTester_up(); f_cf_SMS_up();  var SelectChoiceType v_SelectChoice_A := { » smsURI := PX_SMS_URI, ChoiceTexts := {{ text_ := "a" }}; var SelectChoiceType v_SelectChoice_B := { » smsURI := PX_SMS_URI, ChoiceTexts := {{ text_ := "b" }}; var SelectChoiceType v_SelectChoice_C := { » smsURI := PX_SMS_URI, ChoiceTexts := {{ text_ := "c" }};  var SMSTemplateType v_sms := » valueof(m_def_SmsTemplate( {v_SelectChoice_A, v_SelectChoice_B, » v_SelectChoice_C} ));  var MediaObjectGroupType v_objGroup := » valueof(m_def_MediaObjectGroup(PX_MEDIA_DOCUMENT_GROUP_ID, true, omit)); v_objGroup.SMSTemplate := v_sms; var InteractivityMediaDocumentType v_doc := » valueof(m_def_InteractivityMediaDocument(PX_MEDIA_DOCUMENT_GROUP_ID, 0, » PX_MEDIA_DOCUMENT_ID, 0, {v_objGroup})); var TransferFileList v_FileList :={ { InteractivityMediaDocument := » v_doc } };  var UInt v_TransferId := 1; f_main_data_startFileTransfer(v_TransferId, » PX_SDP_FILE_DELIVERY, v_FileList);  // preamble f_startCommonUtPreamble();  // test behavior var ServiceGuideDeliveryUnit v_SGDU := {}; var UInt32 v_FragmentVersion := 1; </pre>
<pre> f_main_ctrl_InitStartEndTime(c_one_hour,c_one » _hour); // change 01 (WK 6): global time » definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartEndTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := </pre>	=	<pre> var DateTime v_DateTime_StartTime := </pre>

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

» f_getDateTime(v_startTime);           // change 01 (WK 6): global
» time definition
    var DateTime v_DateTime_EndTime :=
» f_getDateTime(v_endTime);             // change 01 (WK 6):
» global time definition

```

```

    var SdpFragment v_Sdp :=
» valueof(m_def_SdpFragment(
    PX_SDP_VIDEO_PROG_1_REF_ID,
    PX_SDP_FILE_DELIVERY
));

    var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
    PX_SDP_VIDEO_PROG_1_REF_ID
));

    var InteractivityDataType v_InterData :=
» valueof(m_def_InteractivityData(
    PX_SGDU_INTERACTIVITY_DATA_ID,
    v_FragmentVersion,
    PX_PRELISTEN_INDICATOR,
    PX_MEDIA_DOCUMENT_GROUP_ID,
    PX_SGDU_SERVICE_ID
));
    v_InterData.InteractivityTypes := {
» {omit,"vote-sms"} };

    var ServiceType v_Service :=
» valueof(m_def_Service(
    PX_SGDU_SERVICE_ID,
    v_FragmentVersion,
    "TvChannel",
    c_basicTv_ServiceType
));

    var ContentType v_Content :=
» valueof(m_def_Content(
    PX_SGDU_CONTENT_ID_PROG_1,
    v_FragmentVersion,
    "Programm1",
    PX_SGDU_SERVICE_ID,
    v_DateTime_StartTime,
    v_DateTime_EndTime
));

```

```

» f_getDateTime(v_NTP_StartTime);

    var DateTime v_DateTime_EndTime :=
» f_getDateTime(v_NTP_EndTime);

    var SdpFragment v_Sdp :=
» valueof(m_def_SdpFragment(
    PX_SDP_VIDEO_PROG_1_REF_ID,
    PX_SDP_FILE_DELIVERY
));

    var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
    PX_SDP_VIDEO_PROG_1_REF_ID
));

    var InteractivityDataType v_InterData :=
» valueof(m_def_InteractivityData(
    PX_SGDU_INTERACTIVITY_DATA_ID,
    v_FragmentVersion,
    PX_PRELISTEN_INDICATOR,
    PX_MEDIA_DOCUMENT_GROUP_ID,
    PX_SGDU_SERVICE_ID
));
    v_InterData.InteractivityTypes := {
» {omit,"vote-sms"} };

    var ServiceType v_Service :=
» valueof(m_def_Service(
    PX_SGDU_SERVICE_ID,
    v_FragmentVersion,
    "TvChannel",
    c_basicTv_ServiceType
));

    var ContentType v_Content :=
» valueof(m_def_Content(
    PX_SGDU_CONTENT_ID_PROG_1,
    v_FragmentVersion,
    "Programm1",
    PX_SGDU_SERVICE_ID,
    v_DateTime_StartTime,
    v_DateTime_EndTime
));

```

Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

        var ScheduleType v_Schedule :=
»   valueof(m_def_inter_Schedule(
        PX_SGDU_SCHEDULE_ID_PROG_1,
        v_FragmentVersion,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_INTERACTIVITY_DATA_ID
    ));

    var AccessType v_Access :=
»   valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_1,
        v_FragmentVersion,
        v_AccessType,
        PX_SGDU_SERVICE_ID,
        omit,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addInteractivityDataFragment(v_SGDU,
»   v_InterData);

    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
»   v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
»   omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
»   f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");
        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckInteractivity("vote-sms");
        f_main_ut_SelectInteractivity("vote-sms");
        f_main_ut_CheckInteractivityChoices( { {text_
»   := "a"}, {text_ := "b"}, {text_ := "c" } } );
        f_main_ut_SelectInteractivityChoice("b");

        f_main_ctrl_awaitingSMS("b"); // TODO - This
»   content check needs to be updated based on resolution of open issue

```

```

        var ScheduleType v_Schedule :=
»   valueof(m_def_inter_Schedule(
        PX_SGDU_SCHEDULE_ID_PROG_1,
        v_FragmentVersion,
        PX_SGDU_SERVICE_ID,
        PX_SGDU_INTERACTIVITY_DATA_ID
    ));

    var AccessType v_Access :=
»   valueof(m_def_Access(
        PX_SGDU_ACCESS_ID_PROG_1,
        v_FragmentVersion,
        v_AccessType,
        PX_SGDU_SERVICE_ID,
        omit,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addInteractivityDataFragment(v_SGDU,
»   v_InterData);

    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
»   v_Sdp);

    f_main_ctrl_broadcastServiceGuide(v_SGDU,
»   omit);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {
»   f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");
        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckInteractivity("vote-sms");
        f_main_ut_SelectInteractivity("vote-sms");
        f_main_ut_CheckInteractivityChoices( { {text_
»   := "a"}, {text_ := "b"}, {text_ := "c" } } );
        f_main_ut_SelectInteractivityChoice("b");

        f_main_ctrl_awaitingSMS("b"); // TODO - This
»   content check needs to be updated based on resolution of open issue

```

Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

        // postamble
        f_main_ut_PowerOff();

        f_cf_Broadcast_down();
        f_cf_Data_down();
        f_cf_UpperTester_down();
        f_cf_SMS_down();
    }// end test case TC_BCAST_INTER_CONF_102

    /** @desc
    *   <p>Service guide with MMS interactivity
    » document via broadcast channel (optional)</p>
    *   <p>Test Case Description</p>
    *   <p>with a terminal listening to the Service
    » Guide Delivery Channel.</p>
    *   <p>when the terminal receives Service Guide
    » containing interactivity data for MMS interactivity</p>
    *   <p>then the terminal sends an MMS</p>
    *   <p>Specification Reference</p>
    *   <p>[BCAST10-Services] Section 5.3.6,
    » 5.3.6.1.7.</p>
    *   <p>Preconditions</p>
    *   <p>Service guide cache of terminal is
    » erased</p>
    *   <p>Terminal is configured to listen to BCAST
    » service guide announcements and delivery on the broadcast channel</p>
    *   <p>Access point information for service guide
    » entry point is configured in test tool</p>
    *   <p>This test cases uses the following SG
    » fragment instantiations found in the Appendix:
    *       <ul>
    *           <li>Service fragment with
    » Name="TvChannel"</li>
    *           <li>Content fragment with
    » Name="Programm1" Content fragment with Name="Programm1" for service
    » fragment</li>
    *           <li>InteractivityData fragment for
    » content fragment with interactivityType set to "vote-mms"</li>
    *           <li>Schedule fragment for interactivity
    » data fragment</li>
    *           <li>Access fragment for schedule
    » fragment and referencing the session description fragment</li>
    *           <li>SDP for file delivery session of
    » interactivity media document and MMS files InteractivityMediaDocument with
    » MediaObject Set element pointing to the MMS Message Template.</li>
    *       </ul>
    *   </p>

```

```

        // postamble
        f_main_ut_PowerOff();

        f_cf_Broadcast_down();
        f_cf_Data_down();
        f_cf_UpperTester_down();
        f_cf_SMS_down();
    }// end test case TC_BCAST_INTER_CONF_102

    /** @desc
    *   <p>Service guide with MMS interactivity
    » document via broadcast channel (optional)</p>
    *   <p>Test Case Description</p>
    *   <p>with a terminal listening to the Service
    » Guide Delivery Channel.</p>
    *   <p>when the terminal receives Service Guide
    » containing interactivity data for MMS interactivity</p>
    *   <p>then the terminal sends an MMS</p>
    *   <p>Specification Reference</p>
    *   <p>[BCAST10-Services] Section 5.3.6,
    » 5.3.6.1.7.</p>
    *   <p>Preconditions</p>
    *   <p>Service guide cache of terminal is
    » erased</p>
    *   <p>Terminal is configured to listen to BCAST
    » service guide announcements and delivery on the broadcast channel</p>
    *   <p>Access point information for service guide
    » entry point is configured in test tool</p>
    *   <p>This test cases uses the following SG
    » fragment instantiations found in the Appendix:
    *       <ul>
    *           <li>Service fragment with
    » Name="TvChannel"</li>
    *           <li>Content fragment with
    » Name="Programm1" Content fragment with Name="Programm1" for service
    » fragment</li>
    *           <li>InteractivityData fragment for
    » content fragment with interactivityType set to "vote-mms"</li>
    *           <li>Schedule fragment for interactivity
    » data fragment</li>
    *           <li>Access fragment for schedule
    » fragment and referencing the session description fragment</li>
    *           <li>SDP for file delivery session of
    » interactivity media document and MMS files InteractivityMediaDocument with
    » MediaObject Set element pointing to the MMS Message Template.</li>
    *       </ul>
    *   </p>

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

*      <p>Set the FLUTE session to transport
*      <ul>
*          <li>InteractivityMediaDocument
» "iamdl.xml" with TOI="1" </li>
*          <li>MMS Message Template
» "mmstemplate.gz" with TOI="2".</li>
*      </ul>
*      </p>
*      <p>Set the FLUTE session to contain an
» FDT-Instance with two 'File'-elements;
*      <ul>
*          <li>one with attributes
» Content-Location="iamdl.xml",
» Content-Type="application/vnd.oma.bcast.imd+xml" and 'TOI="1"' and </li>
*          <li>â€¢ another with attributes
» Content-Location="mmstemplate.gz", Content-Type="text/html" and
» TOI="2".</li>
*      </ul>
*      </p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*      <p>The MMS Message Template contains the
» following files concatenated in one GZIP:
*      <ul>
*          <li>Message Template Definition</li>
*          <li>MMS Presentation Part</li>
*          <li>Image file "ChoiceA.jpg"</li>
*          <li>Image file "ChoiceB.jpg"</li>
*          <li>Text file "Instructions.txt"</li>
*      </ul>
*      </p>
*      <p></p>
*      <p>Test Procedure</p>
*      <p>
*      <ul>
*          <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*          <li>Setup the test tool to deliver the
» interactivity media document, MMS file, and images using a file delivery
» session as described by SDP.</li>
*          <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*          <li>Browse the SG in the terminal</li>
*          <li>Access "Interactivity_URL_MMS" to
» retrieve the InteractivityMediaDocuments</li>

```

```

*      <p>Set the FLUTE session to transport
*      <ul>
*          <li>InteractivityMediaDocument
» "iamdl.xml" with TOI="1" </li>
*          <li>MMS Message Template
» "mmstemplate.gz" with TOI="2".</li>
*      </ul>
*      </p>
*      <p>Set the FLUTE session to contain an
» FDT-Instance with two 'File'-elements;
*      <ul>
*          <li>one with attributes
» Content-Location="iamdl.xml",
» Content-Type="application/vnd.oma.bcast.imd+xml" and 'TOI="1"' and </li>
*          <li>â€¢ another with attributes
» Content-Location="mmstemplate.gz", Content-Type="text/html" and
» TOI="2".</li>
*      </ul>
*      </p>
*      <p>Note: All the fragments are associated with
» the same Service fragment and are sent in the same service guide
» delivery.</p>
*      <p>The MMS Message Template contains the
» following files concatenated in one GZIP:
*      <ul>
*          <li>Message Template Definition</li>
*          <li>MMS Presentation Part</li>
*          <li>Image file "ChoiceA.jpg"</li>
*          <li>Image file "ChoiceB.jpg"</li>
*          <li>Text file "Instructions.txt"</li>
*      </ul>
*      </p>
*      <p></p>
*      <p>Test Procedure</p>
*      <p>
*      <ul>
*          <li>Set up the test tool to produce
» initial BCAST service guide announcement and delivery using broadcast
» channel.</li>
*          <li>Setup the test tool to deliver the
» interactivity media document, MMS file, and images using a file delivery
» session as described by SDP.</li>
*          <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*          <li>Browse the SG in the terminal</li>
*          <li>Access "Interactivity_URL_MMS" to
» retrieve the InteractivityMediaDocuments</li>

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

*          <li>Interact with application to
» generate response and deliver response in an MMS.</li>
*          </ul>
*          </p>
*          <p>Pass-Criteria</p>
*          <p>The following should be visible to the end
» user after the delivery of the SG
*          <ul>
*          <li>Select "TvChannel" and interactivity
» "vote-mms" on the terminal </li>
*          <li>Select image "ChoiceA" on
» terminal.</li>
*          </ul>
*          </p>
*          <p>The following should be observable for the
» test tool
*          <ul>
*          <li>The test tool receives an MMS from
» the terminal formatted correctly according to the MMS Template and it
» contains "ChoiceA" of the user. </li>
*          </ul>
*          </p>
*          @verdict
*          pass in case the client pass the conformance
» test.
*          @verdict
*          fail in case the client does not pass the
» conformance test.
*          @verdict
*          inconc in case a guard timer expires.
*/
testcase TC_BCAST_INTER_CONF_103() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();
    f_cf_MMS_up();

    var UInt v_StreamId := 1;

    var ObjectType v_objMMS :=
» valueof(m_def_Object(PX_MMS_TEMPLATE, "application/vnd.omammsg-mtd+xml",
» true, omit));

    var ObjectType v_objA :=
» valueof(m_def_Object(PX_MMS_VOTE_A_PICTURE, "image/png", omit, omit));

```

```

*          <li>Interact with application to
» generate response and deliver response in an MMS.</li>
*          </ul>
*          </p>
*          <p>Pass-Criteria</p>
*          <p>The following should be visible to the end
» user after the delivery of the SG
*          <ul>
*          <li>Select "TvChannel" and interactivity
» "vote-mms" on the terminal </li>
*          <li>Select image "ChoiceA" on
» terminal.</li>
*          </ul>
*          </p>
*          <p>The following should be observable for the
» test tool
*          <ul>
*          <li>The test tool receives an MMS from
» the terminal formatted correctly according to the MMS Template and it
» contains "ChoiceA" of the user. </li>
*          </ul>
*          </p>
*          @verdict
*          pass in case the client pass the conformance
» test.
*          @verdict
*          fail in case the client does not pass the
» conformance test.
*          @verdict
*          inconc in case a guard timer expires.
*/
testcase TC_BCAST_INTER_CONF_103() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_Data_up();
    f_cf_UpperTester_up();
    f_cf_MMS_up();

    var UInt v_StreamId := 1;

    var ObjectType v_objMMS :=
» valueof(m_def_Object(PX_MMS_TEMPLATE, "application/vnd.omammsg-mtd+xml",
» true, omit));

    var ObjectType v_objA :=
» valueof(m_def_Object(PX_MMS_VOTE_A_PICTURE, "image/png", omit, omit));

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

        var ObjectType v_objB :=
» valueof(m_def_Object(PX_MMS_VOTE_PICTURE, "image/png", omit, omit));
        var ObjectType v_objTxt :=
» valueof(m_def_Object(PX_MMS_TEXT_FILE, "text/plain", omit, omit));

        var MediaObjectSetType v_objSet :=
» valueof(m_def_mediaObjectSet("application/x-gzip", PX_MEDIA_OBJECT_SET_URI,
» {v_objMMS, v_objA, v_objB}));

        var MediaObjectGroupType v_objGroup :=
» valueof(m_def_MediaObjectGroup(PX_MEDIA_DOCUMENT_GROUP_ID, true,
» {v_objSet}));

        var InteractivityMediaDocumentType v_doc :=
» valueof(m_def_InteractivityMediaDocument(PX_MEDIA_DOCUMENT_GROUP_ID, 0,
» PX_MEDIA_DOCUMENT_ID, 0, {v_objGroup}));

        var TransferFileList v_FileList :={
            {
                InteractivityMediaDocument :=
» v_doc
            },
            {
                zipFile := valueof(
» m_def_ZipFile(
» PX_MEDIA_OBJECT_SET_URI,
» PX_MMS_TEMPLATE,
» PX_MMS_VOTE_A_PICTURE,
» PX_MMS_VOTE_PICTURE,
» PX_MMS_TEXT_FILE
                )
            }
        };

        // preamble
        var UInt v_TransferId := 1;
        f_main_data_startFileTransfer(v_TransferId,
» PX_SDP_FILE_DELIVERY, v_FileList);
        f_startCommonUtPreamble();

        // test behavior
        var ServiceGuideDeliveryUnit v_SGDU := {};

```

```

        var ObjectType v_objB :=
» valueof(m_def_Object(PX_MMS_VOTE_PICTURE, "image/png", omit, omit));
        var ObjectType v_objTxt :=
» valueof(m_def_Object(PX_MMS_TEXT_FILE, "text/plain", omit, omit));

        var MediaObjectSetType v_objSet :=
» valueof(m_def_mediaObjectSet("application/x-gzip", PX_MEDIA_OBJECT_SET_URI,
» {v_objMMS, v_objA, v_objB}));

        var MediaObjectGroupType v_objGroup :=
» valueof(m_def_MediaObjectGroup(PX_MEDIA_DOCUMENT_GROUP_ID, true,
» {v_objSet}));

        var InteractivityMediaDocumentType v_doc :=
» valueof(m_def_InteractivityMediaDocument(PX_MEDIA_DOCUMENT_GROUP_ID, 0,
» PX_MEDIA_DOCUMENT_ID, 0, {v_objGroup}));

        var TransferFileList v_FileList :={
            {
                InteractivityMediaDocument :=
» v_doc
            },
            {
                zipFile := valueof(
» m_def_ZipFile(
» PX_MEDIA_OBJECT_SET_URI,
» PX_MMS_TEMPLATE,
» PX_MMS_VOTE_A_PICTURE,
» PX_MMS_VOTE_PICTURE,
» PX_MMS_TEXT_FILE
                )
            }
        };

        // preamble
        var UInt v_TransferId := 1;
        f_main_data_startFileTransfer(v_TransferId,
» PX_SDP_FILE_DELIVERY, v_FileList);
        f_startCommonUtPreamble();

        // test behavior
        var ServiceGuideDeliveryUnit v_SGDU := {};

```



Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

<pre> var UInt32 v_FragmentVersion := 1; </pre>		<pre> var UInt32 v_FragmentVersion := 1; </pre>
<pre>     f_main_ctrl_InitStartTime(c_one_hour,c_one     // change 01 (WK 6): global time » _hour); » definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
	=	
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime);           // change 01 (WK 6): global » time definition var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);             // change 01 (WK 6): » global time definition </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime); var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
	=	
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_FILE_DELIVERY ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var InteractivityDataType v_InterData := » valueof(m_def_InteractivityData(     PX_SGDU_INTERACTIVITY_DATA_ID,     v_FragmentVersion,     PX_PRELISTEN_INDICATOR,     PX_MEDIA_DOCUMENT_GROUP_ID,     PX_SGDU_SERVICE_ID )); v_InterData.InteractivityTypes := { » {omit,"vote-mms"} };  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "TvChannel",     c_basicTv_ServiceType )); </pre>		<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_FILE_DELIVERY ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var InteractivityDataType v_InterData := » valueof(m_def_InteractivityData(     PX_SGDU_INTERACTIVITY_DATA_ID,     v_FragmentVersion,     PX_PRELISTEN_INDICATOR,     PX_MEDIA_DOCUMENT_GROUP_ID,     PX_SGDU_SERVICE_ID )); v_InterData.InteractivityTypes := { » {omit,"vote-mms"} };  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_FragmentVersion,     "TvChannel",     c_basicTv_ServiceType )); </pre>

Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

        var ContentType v_Content :=
» valueof(m_def_Content(
            PX_SGDU_CONTENT_ID_PROG_1,
            v_FragmentVersion,
            "Programmel",
            PX_SGDU_SERVICE_ID,
            v_DateTime_StartTime,
            v_DateTime_EndTime
        ));

        var ScheduleType v_Schedule :=
» valueof(m_def_inter_Schedule(
            PX_SGDU_SCHEDULE_ID_PROG_1,
            v_FragmentVersion,
            PX_SGDU_SERVICE_ID,
            PX_SGDU_INTERACTIVITY_DATA_ID
        ));

        var AccessType v_Access :=
» valueof(m_def_Access(
            PX_SGDU_ACCESS_ID_PROG_1,
            v_FragmentVersion,
            v_AccessType,
            PX_SGDU_SERVICE_ID,
            omit,
            c_class_SG
        ));

        f_addServiceFragment(v_SGDU, v_Service);
        f_addContentFragment(v_SGDU, v_Content);
        f_addInteractivityDataFragment(v_SGDU,
» v_InterData);

        f_addScheduleFragment(v_SGDU, v_Schedule);
        f_addAccessFragment(v_SGDU, v_Access);
        f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");

```

```

        var ContentType v_Content :=
» valueof(m_def_Content(
            PX_SGDU_CONTENT_ID_PROG_1,
            v_FragmentVersion,
            "Programmel",
            PX_SGDU_SERVICE_ID,
            v_DateTime_StartTime,
            v_DateTime_EndTime
        ));

        var ScheduleType v_Schedule :=
» valueof(m_def_inter_Schedule(
            PX_SGDU_SCHEDULE_ID_PROG_1,
            v_FragmentVersion,
            PX_SGDU_SERVICE_ID,
            PX_SGDU_INTERACTIVITY_DATA_ID
        ));

        var AccessType v_Access :=
» valueof(m_def_Access(
            PX_SGDU_ACCESS_ID_PROG_1,
            v_FragmentVersion,
            v_AccessType,
            PX_SGDU_SERVICE_ID,
            omit,
            c_class_SG
        ));

        f_addServiceFragment(v_SGDU, v_Service);
        f_addContentFragment(v_SGDU, v_Content);
        f_addInteractivityDataFragment(v_SGDU,
» v_InterData);

        f_addScheduleFragment(v_SGDU, v_Schedule);
        f_addAccessFragment(v_SGDU, v_Access);
        f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

        f_main_ctrl_broadcastServiceGuide(v_SGDU,
» omit);

        f_main_ut_RunBCastApplication();
        if (PX_GET_SG_DISPLAY_MANUALLY) {
» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");

```

## Datei: AtsBCast\_ServiceInteractionTests.ttcn (Fortsetzung)

```

        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckInteractivity("vote-mms");
        f_main_ut_SelectInteractivity("vote-mms");
        f_main_ut_CheckInteractivityChoices( { {
» text_ := "ChoiceA" } });

» f_main_ut_SelectInteractivityChoice("ChoiceA");

        f_main_ctrl_awaitingMMS("ChoiceA"); // TODO -
» This content check needs to be updated based on resolution of open issue

        // postamble
        f_main_ut_PowerOff();
        f_main_data_stopFileTransfer(v_TransferId);

        f_cf_Broadcast_down();
        f_cf_Data_down();
        f_cf_UpperTester_down();
        f_cf_MMS_down();

        } // end test case TC_BCAST_INTER_CONF_103
    } // end group clientConformanceTestCases
} // end group bcastConformanceTestCases
}

```

```

        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckInteractivity("vote-mms");
        f_main_ut_SelectInteractivity("vote-mms");
        f_main_ut_CheckInteractivityChoices( { {
» text_ := "ChoiceA" } });

» f_main_ut_SelectInteractivityChoice("ChoiceA");

        f_main_ctrl_awaitingMMS("ChoiceA"); // TODO -
» This content check needs to be updated based on resolution of open issue

        // postamble
        f_main_ut_PowerOff();
        f_main_data_stopFileTransfer(v_TransferId);

        f_cf_Broadcast_down();
        f_cf_Data_down();
        f_cf_UpperTester_down();
        f_cf_MMS_down();

        } // end test case TC_BCAST_INTER_CONF_103
    } // end group clientConformanceTestCases
} // end group bcastConformanceTestCases
}

```

## Datei: AtsBCast\_ServiceProvisioningTests.ttcn

```

/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies pilot tests for service provisioning.
 */
module AtsBCast_ServiceProvisioningTests {
    import from LibBCast_Interface all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibCommon_BasicTypesAndValues all;
    import from AtsBCast_ModuleParameters all;
    import from LibBCast_ModuleParameters all;
    import from AtsBCast_TestConfiguration_Functions all;
    import from AtsBCast_TestSystem all;
}

```

```

=
/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies pilot tests for service provisioning.
 */
module AtsBCast_ServiceProvisioningTests {
    import from LibBCast_Interface all;
    import from AtsBCast_Main_Functions all;
    import from LibBCast_ServiceGuide_Templates all;
    import from AtsBCast_ServiceGuide_Functions all;
    import from LibBCast_ServiceGuide_TypesAndValues all;
    import from LibBCast_Common_TypesAndValues all;
    import from LibCommon_BasicTypesAndValues all;
    import from AtsBCast_ModuleParameters all;
    import from LibBCast_ModuleParameters all;
    import from AtsBCast_TestConfiguration_Functions all;
    import from AtsBCast_TestSystem all;
}

```

## Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

<pre> group bcastConformanceTestCases {     group clientConformanceTestCases {         /**          *          * @desc          * &lt;p&gt;Service guide discovery via broadcast » channel&lt;/p&gt;          * &lt;p&gt;Test Case Description&lt;/p&gt;          * &lt;p&gt;with a terminal broadcast bearer established » and configured to use the fixed service guide entry point&lt;/p&gt;          * &lt;p&gt;when the terminal receives a service guide » announcement referring to a service guide containing one service, content, » access, and schedule fragment&lt;/p&gt;          * &lt;p&gt;then the terminal presents the service and » content name to the user.&lt;/p&gt;          * &lt;p&gt;Specification Reference&lt;/p&gt;          * &lt;p&gt;[BCAST10-ESG] Section 5.1.2.1, 5.1.2.2, » 5.1.2.3, 5.1.2.4, 5.4.2.1, 6.1.1&lt;/p&gt;          * &lt;p&gt;Preconditions&lt;/p&gt;          * &lt;p&gt;Service guide cache of terminal.is » erased.&lt;/p&gt;          * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel &lt;/p&gt;          * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;          * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:          * &lt;ul&gt;          * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;          * &lt;li&gt;Content fragment for service » fragment with Name="Programm1" , and StartTime and EndTime elements » indicating values after the time of test&lt;/li&gt;          * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt;          * &lt;li&gt;Access fragment for schedule » fragment&lt;/li&gt;          * &lt;/ul&gt;          * &lt;/p&gt;          * &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt;          * &lt;p&gt;Test Procedure&lt;/p&gt;          * &lt;p&gt;          * &lt;ul&gt;          * &lt;li&gt;Set up the test tool to produce </pre>	<pre> group bcastConformanceTestCases {     group clientConformanceTestCases {         /**          *          * @desc          * &lt;p&gt;Service guide discovery via broadcast » channel&lt;/p&gt;          * &lt;p&gt;Test Case Description&lt;/p&gt;          * &lt;p&gt;with a terminal broadcast bearer established » and configured to use the fixed service guide entry point&lt;/p&gt;          * &lt;p&gt;when the terminal receives a service guide » announcement referring to a service guide containing one service, content, » access, and schedule fragment&lt;/p&gt;          * &lt;p&gt;then the terminal presents the service and » content name to the user.&lt;/p&gt;          * &lt;p&gt;Specification Reference&lt;/p&gt;          * &lt;p&gt;[BCAST10-ESG] Section 5.1.2.1, 5.1.2.2, » 5.1.2.3, 5.1.2.4, 5.4.2.1, 6.1.1&lt;/p&gt;          * &lt;p&gt;Preconditions&lt;/p&gt;          * &lt;p&gt;Service guide cache of terminal.is » erased.&lt;/p&gt;          * &lt;p&gt;Terminal is configured to listen to BCAST » service guide announcements and delivery on the broadcast channel &lt;/p&gt;          * &lt;p&gt;Access point information for service guide » entry point is configured in test tool&lt;/p&gt;          * &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix:          * &lt;ul&gt;          * &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt;          * &lt;li&gt;Content fragment for service » fragment with Name="Programm1" , and StartTime and EndTime elements » indicating values after the time of test&lt;/li&gt;          * &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt;          * &lt;li&gt;Access fragment for schedule » fragment&lt;/li&gt;          * &lt;/ul&gt;          * &lt;/p&gt;          * &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt;          * &lt;p&gt;Test Procedure&lt;/p&gt;          * &lt;p&gt;          * &lt;ul&gt;          * &lt;li&gt;Set up the test tool to produce </pre>
--	--

## Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

```

» BCAST service guide announcement and delivery using broadcast channel.
» </li>
*           <li>Activate the BCAST application of
» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible to the
» terminal user
*           <ul>
*           <li>There is a service "TvChannel"
» associated with program "Programm1" as well as start and end time
» values(There is a "TvChannel" that contains "Programm1" scheduled from
» startTime to endTime)</li>
*           </ul>
*           </p>
*           @verdict
*           pass in case the client pass the conformance
» test.
*           @verdict
*           fail in case the client does not pass the
» conformance test.
*           @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_PROV_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();

    // test behavior
    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_Version := 1;

    f_main_ctrl_InitStartEndTime(c_one_hour,c_one
» _hour);
» definition
    // change 01 (WK 6): global time

```

```

» BCAST service guide announcement and delivery using broadcast channel.
» </li>
*           <li>Activate the BCAST application of
» the terminal</li>
*           <li>Request from BCAST application on
» terminal to get the service guide (optional)</li>
*           <li>Browse the SG on the terminal</li>
*           </ul>
*           </p>
*           <p>Pass-Criteria</p>
*           <p>The following should be visible to the
» terminal user
*           <ul>
*           <li>There is a service "TvChannel"
» associated with program "Programm1" as well as start and end time
» values(There is a "TvChannel" that contains "Programm1" scheduled from
» startTime to endTime)</li>
*           </ul>
*           </p>
*           @verdict
*           pass in case the client pass the conformance
» test.
*           @verdict
*           fail in case the client does not pass the
» conformance test.
*           @verdict
*           inconc in case a guard timer expires.
*/
testcase TC_BCAST_PROV_CONF_101() runs on
» BCastMainComponent system BCastPlatformComponent {
    // init components and ports
    f_createComponents();
    f_cf_Broadcast_up();
    f_cf_UpperTester_up();

    // preamble
    f_startCommonUtPreamble();

    // test behavior
    var ServiceGuideDeliveryUnit v_SGDU := {};
    var UInt32 v_Version := 1;

    var UInt v_NTP_StartTime := 0;

    var UInt v_NTP_EndTime := 0;
    f_InitStartEndTime(v_NTP_StartTime,

```

Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

	=	» v_NTP_EndTime);
	=	
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime);           // change 01 (WK 6): global time » definition var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime);             // change 01 (WK 6): global time » definition </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime); var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_Version,     "TvChannel",     c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_Version,     "Programm1",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_Version,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_startTime, </pre>	=	<pre> var SdpFragment v_Sdp := » valueof(m_def_SdpFragment(     PX_SDP_VIDEO_PROG_1_REF_ID,     PX_SDP_VIDEO_PROG_1 ));  var AccessTypeType v_AccessType := » valueof(m_def_AccessType_bc_sdp( » f_getAccessType(PX_BROADCAST_NETWORK_BEARER),     PX_SDP_VIDEO_PROG_1_REF_ID ));  var ServiceType v_Service := » valueof(m_def_Service(     PX_SGDU_SERVICE_ID,     v_Version,     "TvChannel",     c_basicTv_ServiceType ));  var ContentType v_Content := » valueof(m_def_Content(     PX_SGDU_CONTENT_ID_PROG_1,     v_Version,     "Programm1",     PX_SGDU_SERVICE_ID,     v_DateTime_StartTime,     v_DateTime_EndTime ));  var ScheduleType v_Schedule := » valueof(m_def_Schedule(     PX_SGDU_SCHEDULE_ID_PROG_1,     v_Version,     PX_SGDU_SERVICE_ID,     PX_SGDU_CONTENT_ID_PROG_1,     v_NTP_StartTime, </pre>
	<>	

Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

» // change 01 (WK 6): global time definition v_endTime » // change 01 (WK 6): global time definition		v_NTP_EndTime
<pre>         ));          var AccessType v_Access := » valueof(m_def_Access(             PX_SGDU_ACCESS_ID_PROG_1,             v_Version,             v_AccessType,             PX_SGDU_SERVICE_ID,             PX_SGDU_SCHEDULE_ID_PROG_1,             c_class_SG         ));          f_addServiceFragment(v_SGDU, v_Service);         f_addContentFragment(v_SGDU, v_Content);         f_addScheduleFragment(v_SGDU, v_Schedule);         f_addAccessFragment(v_SGDU, v_Access);         f_addSdpFragment(v_SGDU, v_Version, v_Sdp);          f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);          f_main_ut_RunBCastApplication();         if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);             }             f_main_ut_CheckService("TvChannel");             f_main_ut_SelectService("TvChannel");             f_main_ut_CheckContent("Programm1", » v_DateTime_StartTime, v_DateTime_EndTime );              // postamble             f_main_ut_PowerOff();              f_cf_Broadcast_down();             f_cf_UpperTester_down();         } // end test case TC_BCAST_PROV_CONF_101  /**  *  * @desc  * &lt;p&gt;Service guide discovery via interaction » channel (optional)&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt; </pre>	=	<pre>         ));          var AccessType v_Access := » valueof(m_def_Access(             PX_SGDU_ACCESS_ID_PROG_1,             v_Version,             v_AccessType,             PX_SGDU_SERVICE_ID,             PX_SGDU_SCHEDULE_ID_PROG_1,             c_class_SG         ));          f_addServiceFragment(v_SGDU, v_Service);         f_addContentFragment(v_SGDU, v_Content);         f_addScheduleFragment(v_SGDU, v_Schedule);         f_addAccessFragment(v_SGDU, v_Access);         f_addSdpFragment(v_SGDU, v_Version, v_Sdp);          f_main_ctrl_broadcastServiceGuide(v_SGDU, » omit);          f_main_ut_RunBCastApplication();         if (PX_GET_SG_DISPLAY_MANUALLY) {  » f_main_ut_GetServiceGuide(PX_SGDD_ID);             }             f_main_ut_CheckService("TvChannel");             f_main_ut_SelectService("TvChannel");             f_main_ut_CheckContent("Programm1", » v_DateTime_StartTime, v_DateTime_EndTime );              // postamble             f_main_ut_PowerOff();              f_cf_Broadcast_down();             f_cf_UpperTester_down();         } // end test case TC_BCAST_PROV_CONF_101  /**  *  * @desc  * &lt;p&gt;Service guide discovery via interaction » channel (optional)&lt;/p&gt;  * &lt;p&gt;Test Case Description&lt;/p&gt; </pre>

## Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

<pre> *      &lt;p&gt;with a terminal interaction bearer » established and configured to use a fixed service guide entry point&lt;/p&gt; *      &lt;p&gt;when the terminal requests and receives a » service guide announcement referring to a service guide containing one » service, content, access, and schedule fragment&lt;/p&gt; *      &lt;p&gt;then the terminal presents the service and » content name to the user.&lt;/p&gt; *      &lt;p&gt;Specification Reference&lt;/p&gt; *      &lt;p&gt;[BCAST10-ESG] Section 5.1.2.1, 5.1.2.2, » 5.1.2.3, 5.1.2.4,, 5.4.3, 6.1.2, 6.2&lt;/p&gt; </pre>		<pre> *      &lt;p&gt;with a terminal interaction bearer » established and configured to use a fixed service guide entry point&lt;/p&gt; *      &lt;p&gt;when the terminal requests and receives a » service guide announcement referring to a service guide containing one » service, content, access, and schedule fragment&lt;/p&gt; *      &lt;p&gt;then the terminal presents the service and » content name to the user.&lt;/p&gt; *      &lt;p&gt;Specification Reference&lt;/p&gt; *      &lt;p&gt;[BCAST10 -ESG] Section 5.1.2.1, 5.1.2.2, » 5.1.2.3, 5.1.2.4,, 5.4.3, 6.1.2, 6.2&lt;/p&gt; </pre>
<pre> *      &lt;p&gt;Preconditions&lt;/p&gt; *      &lt;p&gt;Erase service guide cache of terminal.&lt;/p&gt; *      &lt;p&gt;Configure terminal to listen to BCAST » service guide announcements and delivery on the interaction channel&lt;/p&gt; *      &lt;p&gt;Configure access point information for » service guide entry point in test tool&lt;/p&gt; *      &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix: *      &lt;ul&gt; *          &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt; *          &lt;li&gt;Content fragment with » Name="Programmel" , and StartTime and EndTime elements indicating values » after the time of test&lt;/li&gt; *          &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt; *          &lt;li&gt;Access fragment for schedule » fragment&lt;/li&gt; *      &lt;/ul&gt; *      &lt;p&gt;&lt;/p&gt; *      &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; *      &lt;p&gt;Test Procedure&lt;/p&gt; *      &lt;p&gt; *      &lt;ul&gt; *          &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using interaction » channel.&lt;/li&gt; *          &lt;li&gt;Activate the BCAST application of » the terminal&lt;/li&gt; *          &lt;li&gt;Request from BCAST application on » terminal to get the service guide (optional).&lt;/li&gt; *          &lt;li&gt;Browse the SG on the terminal.&lt;/li&gt; *      &lt;/ul&gt; *      &lt;p&gt;&lt;/p&gt; </pre>	=	<pre> *      &lt;p&gt;Preconditions&lt;/p&gt; *      &lt;p&gt;Erase service guide cache of terminal.&lt;/p&gt; *      &lt;p&gt;Configure terminal to listen to BCAST » service guide announcements and delivery on the interaction channel&lt;/p&gt; *      &lt;p&gt;Configure access point information for » service guide entry point in test tool&lt;/p&gt; *      &lt;p&gt;This test case uses the following SG » fragment instantiations found in the Appendix: *      &lt;ul&gt; *          &lt;li&gt;Service fragment with » Name="TvChannel"&lt;/li&gt; *          &lt;li&gt;Content fragment with » Name="Programmel" , and StartTime and EndTime elements indicating values » after the time of test&lt;/li&gt; *          &lt;li&gt;Schedule fragment for content » fragment with same values for startTime and endTime in the » presentationWindow element&lt;/li&gt; *          &lt;li&gt;Access fragment for schedule » fragment&lt;/li&gt; *      &lt;/ul&gt; *      &lt;p&gt;&lt;/p&gt; *      &lt;p&gt;Note: All the fragments are associated with » the same Service fragment and are sent in the same service guide » delivery.&lt;/p&gt; *      &lt;p&gt;Test Procedure&lt;/p&gt; *      &lt;p&gt; *      &lt;ul&gt; *          &lt;li&gt;Set up the test tool to produce » BCAST service guide announcement and delivery using interaction » channel.&lt;/li&gt; *          &lt;li&gt;Activate the BCAST application of » the terminal&lt;/li&gt; *          &lt;li&gt;Request from BCAST application on » terminal to get the service guide (optional).&lt;/li&gt; *          &lt;li&gt;Browse the SG on the terminal.&lt;/li&gt; *      &lt;/ul&gt; *      &lt;p&gt;&lt;/p&gt; </pre>



## Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

<pre> *      &lt;p&gt;Pass-Criteria&lt;/p&gt; *      &lt;p&gt;The following should be visible to the » terminal user *      &lt;ul&gt; *          &lt;li&gt;There is a service "TvChannel" » associated with a program "Programm1" as well as start and end time » values(There is a "TvChannel" that contains "Programm1" scheduled from » startTime to endTime)&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; *      @verdict *          pass in case the client pass the conformance » test. *      @verdict *          fail in case the client does not pass the » conformance test. *      @verdict *          inconc in case a guard timer expires. */ testcase TC_BCAST_PROV_CONF_102() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Interaction_up();     f_cf_UpperTester_up();      // preamble     f_startCommonUtPreamble();      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>		<pre> *      &lt;p&gt;Pass-Criteria&lt;/p&gt; *      &lt;p&gt;The following should be visible to the » terminal user *      &lt;ul&gt; *          &lt;li&gt;There is a service "TvChannel" » associated with a program "Programm1" as well as start and end time » values(There is a "TvChannel" that contains "Programm1" scheduled from » startTime to endTime)&lt;/li&gt; *      &lt;/ul&gt; *      &lt;/p&gt; *      @verdict *          pass in case the client pass the conformance » test. *      @verdict *          fail in case the client does not pass the » conformance test. *      @verdict *          inconc in case a guard timer expires. */ testcase TC_BCAST_PROV_CONF_102() runs on » BCastMainComponent system BCastPlatformComponent {     // init components and ports     f_createComponents();     f_cf_Interaction_up();     f_cf_UpperTester_up();      // preamble     f_startCommonUtPreamble();      // test behavior     var ServiceGuideDeliveryUnit v_SGDU := {};     var UInt32 v_FragmentVersion := 1; </pre>
<pre> » _hour);     f_main_ctrl_InitStartTime(c_one_hour,c_one     // change 01 (WK 6): global time definition </pre>	<>	<pre> var UInt v_NTP_StartTime := 0;  var UInt v_NTP_EndTime := 0; f_InitStartTime(v_NTP_StartTime, » v_NTP_EndTime); </pre>
<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_startTime); » definition     // change 01 (WK 6): global time var DateTime v_DateTime_EndTime := » f_getDateTime(v_endTime); » definition     // change 01 (WK 6): global time </pre>	<>	<pre> var DateTime v_DateTime_StartTime := » f_getDateTime(v_NTP_StartTime);  var DateTime v_DateTime_EndTime := » f_getDateTime(v_NTP_EndTime); </pre>
<pre> var SdpFragment v_Sdp := </pre>	=	<pre> var SdpFragment v_Sdp := </pre>

## Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

```

» valueof(m_def_SdpFragment(
                                PX_SDP_VIDEO_PROG_1_REF_ID,
                                PX_SDP_VIDEO_PROG_1
                                ));

                                var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                                PX_SDP_VIDEO_PROG_1_REF_ID
                                ));

                                var ServiceType v_Service :=
» valueof(m_def_Service(
                                PX_SGDU_SERVICE_ID,
                                v_FragmentVersion,
                                "TvChannel",
                                c_basicTv_ServiceType
                                ));

                                var ContentType v_Content :=
» valueof(m_def_Content(
                                PX_SGDU_CONTENT_ID_PROG_1,
                                v_FragmentVersion,
                                "Programm1",
                                PX_SGDU_SERVICE_ID,
                                v_DateTime_StartTime,
                                v_DateTime_EndTime
                                ));

                                var ScheduleType v_Schedule :=
» valueof(m_def_Schedule(
                                PX_SGDU_SCHEDULE_ID_PROG_1,
                                v_FragmentVersion,
                                PX_SGDU_SERVICE_ID,
                                PX_SGDU_CONTENT_ID_PROG_1,
                                v_startTime,
                                v_endTime
                                ));

                                var AccessType v_Access :=
» valueof(m_def_Access(
                                PX_SGDU_ACCESS_ID_PROG_1,
                                v_FragmentVersion,
                                v_AccessType,

```

```

» valueof(m_def_SdpFragment(
                                PX_SDP_VIDEO_PROG_1_REF_ID,
                                PX_SDP_VIDEO_PROG_1
                                ));

                                var AccessTypeType v_AccessType :=
» valueof(m_def_AccessType_bc_sdp(

» f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
                                PX_SDP_VIDEO_PROG_1_REF_ID
                                ));

                                var ServiceType v_Service :=
» valueof(m_def_Service(
                                PX_SGDU_SERVICE_ID,
                                v_FragmentVersion,
                                "TvChannel",
                                c_basicTv_ServiceType
                                ));

                                var ContentType v_Content :=
» valueof(m_def_Content(
                                PX_SGDU_CONTENT_ID_PROG_1,
                                v_FragmentVersion,
                                "Programm1",
                                PX_SGDU_SERVICE_ID,
                                v_DateTime_StartTime,
                                v_DateTime_EndTime
                                ));

                                var ScheduleType v_Schedule :=
» valueof(m_def_Schedule(
                                PX_SGDU_SCHEDULE_ID_PROG_1,
                                v_FragmentVersion,
                                PX_SGDU_SERVICE_ID,
                                PX_SGDU_CONTENT_ID_PROG_1,
                                v_NTP_StartTime,
                                v_NTP_EndTime
                                ));

                                var AccessType v_Access :=
» valueof(m_def_Access(
                                PX_SGDU_ACCESS_ID_PROG_1,
                                v_FragmentVersion,
                                v_AccessType,

```

## Datei: AtsBCast\_ServiceProvisioningTests.ttcn (Fortsetzung)

```

        PX_SGDU_SERVICE_ID,
        PX_SGDU_SCHEDULE_ID_PROG_1,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

    f_main_ctrl_ServiceGuideOnRequest(v_SGDU);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");
        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckContent("Programm1",
» v_DateTime_StartTime, v_DateTime_EndTime);

        // postamble
        f_main_ut_PowerOff();

        f_cf_Interaction_down();
        f_cf_UpperTester_down();
        } // end test case TC_BCAST_PROV_CONF_102
    } // end group clientConformanceTestCases
} // end group bcastConformanceTestCases
}

```

```

        PX_SGDU_SERVICE_ID,
        PX_SGDU_SCHEDULE_ID_PROG_1,
        c_class_SG
    ));

    f_addServiceFragment(v_SGDU, v_Service);
    f_addContentFragment(v_SGDU, v_Content);
    f_addScheduleFragment(v_SGDU, v_Schedule);
    f_addAccessFragment(v_SGDU, v_Access);
    f_addSdpFragment(v_SGDU, v_FragmentVersion,
» v_Sdp);

    f_main_ctrl_ServiceGuideOnRequest(v_SGDU);

    f_main_ut_RunBCastApplication();
    if (PX_GET_SG_DISPLAY_MANUALLY) {

» f_main_ut_GetServiceGuide(PX_SGDD_ID);
        }
        f_main_ut_CheckService("TvChannel");
        f_main_ut_SelectService("TvChannel");
        f_main_ut_CheckContent("Programm1",
» v_DateTime_StartTime, v_DateTime_EndTime);

        // postamble
        f_main_ut_PowerOff();

        f_cf_Interaction_down();
        f_cf_UpperTester_down();
        } // end test case TC_BCAST_PROV_CONF_102
    } // end group clientConformanceTestCases
} // end group bcastConformanceTestCases
}

```

## Datei: AtsBCast\_TestControl.ttcn

```

/**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies a fairly static test execution control. Via
 *   module parameters for test area or specific test case identifiers
 *   execution can be controlled. The BCAST test suite can also be
 *   compiled without this module. In that case test case execution must
 *   be implemented via the TCI-TM interface.

```

```

= /**
 *
 * @author
 *   ETSI CTI BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies a fairly static test execution control. Via
 *   module parameters for test area or specific test case identifiers
 *   execution can be controlled. The BCAST test suite can also be
 *   compiled without this module. In that case test case execution must
 *   be implemented via the TCI-TM interface.

```

Datei: AtsBCast\_TestControl.ttcn (Fortsetzung)

```

*/
module AtsBCast_TestControl {
    import from AtsBCast_ModuleParameters all;
    import from AtsBCast_ContentProtectionTests all;
    import from AtsBCast_FileAndStreamDistributionTests all;
    import from AtsBCast_ServiceGuideTests all;
    import from AtsBCast_ServiceInteractionTests all;
    import from AtsBCast_ServiceProvisioningTests all;

    // test case selection switches
    control {
        log ("Control part started...");
        if (PX_ALL_TCS) {
            log ("Running all test cases");
        }
        if (PX_ALL_TCS or PX_ALL_SP_TCS) {
            log ("Running all service provisoring test cases");
            execute (TC_BCAST_PROV_CONF_101());
            execute (TC_BCAST_PROV_CONF_101());
        }
        if (PX_ALL_TCS or PX_ALL_SG_TCS) {
            log ("Running all service guide test cases");
            execute (TC_BCAST_ESG_CONF_101());
            execute (TC_BCAST_ESG_CONF_102());
            execute (TC_BCAST_ESG_CONF_103());
            execute (TC_BCAST_ESG_CONF_104());
            execute (TC_BCAST_ESG_CONF_105());
            execute (TC_BCAST_ESG_CONF_106());
            execute (TC_BCAST_ESG_CONF_107());
            execute (TC_BCAST_ESG_CONF_108());
        }
        if (PX_ALL_TCS or PX_ALL_FD_TCS) {
            log ("Running all file distribution test cases");
            execute (TC_BCAST_DIST_CONF_102());
        }
        if (PX_ALL_TCS or PX_ALL_SI_TCS) {
            log ("Running all service interaction test cases");
            execute (TC_BCAST_INTER_CONF_101());
            execute (TC_BCAST_INTER_CONF_102());
            execute (TC_BCAST_INTER_CONF_103());
        }
        if (PX_ALL_TCS or PX_ALL_CP_TCS) {
            log ("Running all content protection test cases");
            execute (TC_BCAST_CONTPROT_CONF_101());
            execute (TC_BCAST_CONTPROT_CONF_102());
            execute (TC_BCAST_CONTPROT_CONF_103());
        }
    }
}

```

```

*/
module AtsBCast_TestControl {
    import from AtsBCast_ModuleParameters all;
    import from AtsBCast_ContentProtectionTests all;
    import from AtsBCast_FileAndStreamDistributionTests all;
    import from AtsBCast_ServiceGuideTests all;
    import from AtsBCast_ServiceInteractionTests all;
    import from AtsBCast_ServiceProvisioningTests all;

    // test case selection switches
    control {
        log ("Control part started...");
        if (PX_ALL_TCS) {
            log ("Running all test cases");
        }
        if (PX_ALL_TCS or PX_ALL_SP_TCS) {
            log ("Running all service provisoring test cases");
            execute (TC_BCAST_PROV_CONF_101());
            execute (TC_BCAST_PROV_CONF_102());
        }
        if (PX_ALL_TCS or PX_ALL_SG_TCS) {
            log ("Running all service guide test cases");
            execute (TC_BCAST_ESG_CONF_101());
            execute (TC_BCAST_ESG_CONF_102());
            execute (TC_BCAST_ESG_CONF_103());
            execute (TC_BCAST_ESG_CONF_104());
            execute (TC_BCAST_ESG_CONF_105());
            execute (TC_BCAST_ESG_CONF_106());
            execute (TC_BCAST_ESG_CONF_107());
            execute (TC_BCAST_ESG_CONF_108());
        }
        if (PX_ALL_TCS or PX_ALL_FD_TCS) {
            log ("Running all file distribution test cases");
            execute (TC_BCAST_DIST_CONF_102());
        }
        if (PX_ALL_TCS or PX_ALL_SI_TCS) {
            log ("Running all service interaction test cases");
            execute (TC_BCAST_INTER_CONF_101());
            execute (TC_BCAST_INTER_CONF_102());
            execute (TC_BCAST_INTER_CONF_103());
        }
        if (PX_ALL_TCS or PX_ALL_CP_TCS) {
            log ("Running all content protection test cases");
            execute (TC_BCAST_CONTPROT_CONF_101());
            execute (TC_BCAST_CONTPROT_CONF_102());
            execute (TC_BCAST_CONTPROT_CONF_103());
        }
    }
}

```

Datei: AtsBCast\_TestControl.ttcn (Fortsetzung)

<pre>         }     } // end module AtsBCast_TestControl </pre>	<pre>         }     } // end module AtsBCast_TestControl </pre>
---	---

Datei: AtsBCast\_TestSystem.ttcn

<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies component types for test components used in the  *   BCAST test suite. The system component types is defined in the BCAST  *   platform interface module  * @see  *   LibBCast_Interface.BCastPlatformComponent  */ module AtsBCast_TestSystem {     import from LibBCast_Interface all;     import from LibCommon_BasicTypesAndValues {         type UInt32;         type UInt;     }     » change 01 (WK 6): global time definition      group atsComponentTypes {         /**          *          * @desc          *   This component type is used for the Main Test         » Component (MTC)          *   which coordinates the creation and execution of BCAST         » control,          *   BCAST data and Upper Tester components.         */         type component BCastMainComponent {             var BCastNetworkControlComponent vc_CTRL;             var BCastNetworkDataComponent vc_DATA;             var UpperTesterComponent vc_UTC;             var UInt32 vc_BCASTTransportID := 1;         }         » change 05 (WK 6): renaming to vc_BCASTTransportID, as SG fragments are         » distinguished with transport IDs         var UInt v_startTime := 0;         » change 01 (WK 6): global time definition         var UInt v_endTime := 0;         » change 01 (WK 6): global time definition     } } </pre>	=	<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies component types for test components used in the  *   BCAST test suite. The system component types is defined in the BCAST  *   platform interface module  * @see  *   LibBCast_Interface.BCastPlatformComponent  */ module AtsBCast_TestSystem {     import from LibBCast_Interface all;     import from LibCommon_BasicTypesAndValues {         type UInt32;     }     » change 01 (WK 6): global time definition      group atsComponentTypes {         /**          *          * @desc          *   This component type is used for the Main Test         » Component (MTC)          *   which coordinates the creation and execution of BCAST         » control,          *   BCAST data and Upper Tester components.         */         type component BCastMainComponent {             var BCastNetworkControlComponent vc_CTRL;             var BCastNetworkDataComponent vc_DATA;             var UpperTesterComponent vc_UTC;             var UInt32 vc_TOI := 1;         }     } } </pre>
---	---	---

Datei: AtsBCast\_TestSystem.ttcn (Fortsetzung)

<pre>     }      group testSystemComponents {         /**          * @desc          *   This component type defines the interface for BCAST » tests to the          *   SUT Adapter. It shall be used as the system component » in any BCAST          *   test case statement         */         type component BCastPlatformComponent {             port UtpPort utp;             port BroadcastPort ac;             port BroadcastPort dc;             port InteractionPort ic;             port StreamServerCtrlPort ssc;             port SmsPort sms;             port MmsPort mms;             port FileServerControlPort fsc;         }     }  } // end module AtsBCast_TestSystem </pre>	<pre>     }      group testSystemComponents {         /**          * @desc          *   This component type defines the interface for BCAST » tests to the          *   SUT Adapter. It shall be used as the system component » in any BCAST          *   test case statement         */         type component BCastPlatformComponent {             port UtpPort utp;             port BroadcastPort ac;             port BroadcastPort dc;             port InteractionPort ic;             port StreamServerCtrlPort ssc;             port SmsPort sms;             port MmsPort mms;             port FileServerControlPort fsc;         }     }  } // end module AtsBCast_TestSystem </pre>
--	--

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn

<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies some function which can be used by the » BCastNetworkControl component.  */ module LibBCast_BCastNetworkControl_Functions {     import from LibBCast_Common_TypesAndValues all;     import from LibBCast_ServiceGuide_TypesAndValues all;     import from AtsBCast_ServiceGuide_Functions all;     import from LibBCast_Common_Templates all;     import from LibBCast_ServicePrimitives_TypesAndValues all;     import from LibBCast_UpperTester_Functions all;     import from LibBCast_Interface all;     import from LibBCast_ModuleParameters all;     import from LibCommon_Time all;     import from LibCommon_BasicTypesAndValues all; </pre>	=	<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies some function which can be used by the » BCastNetworkControl component.  */ module LibBCast_BCastNetworkControl_Functions {     import from LibBCast_Common_TypesAndValues all;     import from LibBCast_ServiceGuide_TypesAndValues all;     import from AtsBCast_ServiceGuide_Functions all;     import from LibBCast_Common_Templates all;     import from LibBCast_ServicePrimitives_TypesAndValues all;     import from LibBCast_UpperTester_Functions all;     import from LibBCast_Interface all;     import from LibBCast_ModuleParameters all;     import from LibCommon_Time all;     import from LibCommon_BasicTypesAndValues all; </pre>
---	---	---

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre> import from AtsBCast_ModuleParameters all;           // change » 16 (WK 6): usage of PIXIT parameter for flute session import from LibBCast_ServiceGuide_Templates {       // change » 13 (WK 6): send DVB-H bootstrap files via FLUTE     group DVBH_Bootstrap; }; </pre>	+ -	
<pre> group subFunctions {     group behaviorFunctions {          /**          *          * @desc          *     This function waits for a http subscription » indicaton.           * @param          *     serviceName the name of the service          * @verdict          *     pass The substription was successfull received » for the           *     given service.          * @verdict          *     fail A message was received which is not » expected.           * @verdict          *     inconc A guard timer expires.         */          function » f_ctrl_awaitingHttpSubscriptionIndication(charstring serviceName) runs on » BCastNetworkControlComponent {             t_wait.start(PX_TWAIT);              alt {                  [] ic.receive » (HttpSubscriptionRequestIndication:{?, serviceName}) {                     t_wait.stop;                     setverdict(pass);                 }                  [] ic.receive {                     t_wait.stop;                     setverdict(fail);                 }             }              log("***f_ctrl_awaitingHttpSubscriptionIndication*** wrong message             » received");         }          [] t_wait.timeout { </pre>	=	<pre> group subFunctions {     group behaviorFunctions {          /**          *          * @desc          *     This function waits for a http subscription » indicaton.           * @param          *     serviceName the name of the service          * @verdict          *     pass The substription was successfull received » for the           *     given service.          * @verdict          *     fail A message was received which is not » expected.           * @verdict          *     inconc A guard timer expires.         */          function » f_ctrl_awaitingHttpSubscriptionIndication(charstring serviceName) runs on » BCastNetworkControlComponent {             t_wait.start(PX_TWAIT);              alt {                  [] ic.receive » (HttpSubscriptionRequestIndication:{?, serviceName}) {                     t_wait.stop;                     setverdict(pass);                 }                  [] ic.receive {                     t_wait.stop;                     setverdict(fail);                 }             }              log("***f_ctrl_awaitingHttpSubscriptionIndication*** wrong message             » received");         }          [] t_wait.timeout { </pre>

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

```

                                setverdict(inconc);
» log("***f_ctrl_awaitingHttpSubscriptionIndication*** timer expired");
                                }
                                }
                                }

                                /**
                                 *
                                 * @desc
                                 *   This function waits for a mms indication.
                                 * @param
                                 *   content the content which should be inside of
» the mms.
                                 * @verdict
                                 *   pass The indication was successfull received
» for the
                                 *   given service.
                                 * @verdict
                                 *   fail A message was received which is not
» expected.
                                 * @verdict
                                 *   inconc A guard timer expires.
                                 */
                                function f_ctrl_awaitingMMS(charstring content) runs
» on BCastNetworkControlComponent {
                                    t_wait.start(PX_TWAIT);

                                    alt {
                                        []
» mms.receive(MmsGetIndication:{content}) {
                                            t_wait.stop;
                                            setverdict(pass);
                                        }
                                        [] mms.receive {
                                            t_wait.stop;
                                            setverdict(fail);
                                            log("***f_ctrl_awaitingMMS***
» wrong message received");
                                        }
                                        [] t_wait.timeout {
                                            setverdict(inconc);
                                            log("***f_ctrl_awaitingMMS***
» time expired");
                                        }
                                    }
                                }

```

```

                                setverdict(inconc);
» log("***f_ctrl_awaitingHttpSubscriptionIndication*** timer expired");
                                }
                                }
                                }

                                /**
                                 *
                                 * @desc
                                 *   This function waits for a mms indication.
                                 * @param
                                 *   content the content which should be inside of
» the mms.
                                 * @verdict
                                 *   pass The indication was successfull received
» for the
                                 *   given service.
                                 * @verdict
                                 *   fail A message was received which is not
» expected.
                                 * @verdict
                                 *   inconc A guard timer expires.
                                 */
                                function f_ctrl_awaitingMMS(charstring content) runs
» on BCastNetworkControlComponent {
                                    t_wait.start(PX_TWAIT);

                                    alt {
                                        []
» mms.receive(MmsGetIndication:{content}) {
                                            t_wait.stop;
                                            setverdict(pass);
                                        }
                                        [] mms.receive {
                                            t_wait.stop;
                                            setverdict(fail);
                                            log("***f_ctrl_awaitingMMS***
» wrong message received");
                                        }
                                        [] t_wait.timeout {
                                            setverdict(inconc);
                                            log("***f_ctrl_awaitingMMS***
» time expired");
                                        }
                                    }
                                }

```



Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

```

    }

    /**
     *
     * @desc
     *   This function waits for a sms indication.
     * @param
     *   content the content which should be inside of
» the mms.
     * @verdict
     *   pass The indication was successfull received
» for the
     *   given service.
     * @verdict
     *   fail A message was received which is not
» expected.
     * @verdict
     *   inconc A guard timer expires.
     */
    function f_ctrl_awaitingSMS(charstring content) runs
» on BCastNetworkControlComponent {
        t_wait.start(PX_TWAIT);

        alt {
            []
» sms.receive(SmsGetIndication:{content}) {
                t_wait.stop;
                setverdict(pass);
            }
            [] sms.receive {
                t_wait.stop;
                setverdict(fail);
                log("****f_ctrl_awaitingSMS***
» wrong message received");
            }
            [] t_wait.timeout {
                setverdict(inconc);
                log("****f_ctrl_awaitingSMS***
» time expired");
            }
        }
    }

    /**

```

```

    }

    /**
     *
     * @desc
     *   This function waits for a sms indication.
     * @param
     *   content the content which should be inside of
» the mms.
     * @verdict
     *   pass The indication was successfull received
» for the
     *   given service.
     * @verdict
     *   fail A message was received which is not
» expected.
     * @verdict
     *   inconc A guard timer expires.
     */
    function f_ctrl_awaitingSMS(charstring content) runs
» on BCastNetworkControlComponent {
        t_wait.start(PX_TWAIT);

        alt {
            []
» sms.receive(SmsGetIndication:{content}) {
                t_wait.stop;
                setverdict(pass);
            }
            [] sms.receive {
                t_wait.stop;
                setverdict(fail);
                log("****f_ctrl_awaitingSMS***
» wrong message received");
            }
            [] t_wait.timeout {
                setverdict(inconc);
                log("****f_ctrl_awaitingSMS***
» time expired");
            }
        }
    }

    /**

```

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

```

*
* @desc
*   This function waits for a http post request
» and send the
*
*   related response including the service guide.
* @param
*   p_SGDU The ServiceGuideDeliveryUnit to send
* @param
*   p_SGDU_Id The ServiceGuideDeliveryUnit id
* @param
*   p_SGDD_Id The SGDD id
* @verdict
*   pass in case of that the service guide could
» successful
*
*   send
* @verdict
*   fail in case of tthe service guide could not
» successful
*
*   send
* @verdict
*   inconc a guard timer runs out of time
*/
function f_ctrl_ServiceGuideViaHttp(
ServiceGuideDeliveryUnit p_SGDU,
charstring p_SGDU_Id,
charstring p_SGDD_Id) runs on
» BCastNetworkControlComponent {
    var HttpPostIndication v_PostRequest;

    // awaiting HTTP POST request
    t_wait.start(PX_TWAIT);
    alt{
        [] ic.receive
» (mw_def_HttpPostIndication) -> value v_PostRequest {
            t_wait.stop;
            setverdict(pass);

» f_sendServiceGuideHttpResponse(
                                p_SGDU,
                                PX_SGDU_ID,
                                PX_SGDD_ID,

» v_PostRequest.connectionId,

» v_PostRequest.serviceGuideRequestType
    );

```

```

*
* @desc
*   This function waits for a http post request
» and send the
*
*   related response including the service guide.
* @param
*   p_SGDU The ServiceGuideDeliveryUnit to send
* @param
*   p_SGDU_Id The ServiceGuideDeliveryUnit id
* @param
*   p_SGDD_Id The SGDD id
* @verdict
*   pass in case of that the service guide could
» successful
*
*   send
* @verdict
*   fail in case of tthe service guide could not
» successful
*
*   send
* @verdict
*   inconc a guard timer runs out of time
*/
function f_ctrl_ServiceGuideViaHttp(
ServiceGuideDeliveryUnit p_SGDU,
charstring p_SGDU_Id,
charstring p_SGDD_Id) runs on
» BCastNetworkControlComponent {
    var HttpPostIndication v_PostRequest;

    // awaiting HTTP POST request
    t_wait.start(PX_TWAIT);
    alt{
        [] ic.receive
» (mw_def_HttpPostIndication) -> value v_PostRequest {
            t_wait.stop;
            setverdict(pass);

» f_sendServiceGuideHttpResponse(
                                p_SGDU,
                                PX_SGDU_ID,
                                PX_SGDD_ID,

» v_PostRequest.connectionId,

» v_PostRequest.serviceGuideRequestType
    );

```

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

```

    }
    [] ic.receive {
        t_wait.stop;
        setverdict(fail);
    }
    [] t_wait.timeout {
        setverdict (inconc);
    }
}

/**
 *
 * @desc
 *   This function broadcast a service guide over
» the flute
 *   protocol
 * @param
 *   p_SGDU The ServiceGuideDeliveryUnit
 * @param
 *   p_SGDU_Id The ServiceGuideDeliveryUnit id
 * @param
 *   p_SGDD_Id The id of the
» ServiceGuideDeliveryDescriptor
 * @param
 *   p_Encoding The encoding type for compression
» in case that
 *   the service guide should be compressed
» otherwise this
 *   parameter should be omitted.
 * @verdict
 *   pass in case the broadcast of the
 *   ServiceGuideDeliveryUnit was successful
 * @verdict
 *   fail in case that the broadcast was not
» successful
 * @verdict
 *   inconc in case of the timer runs out of time
 */
function f_ctrl_broadcastServiceGuide(
    ServiceGuideDeliveryUnit p_SGDU,
    charstring p_SGDU_Id,
    charstring p_SGDD_Id,
    template charstring p_Encoding) runs on
» BCastNetworkControlComponent {

```

```

    }
    [] ic.receive {
        t_wait.stop;
        setverdict(fail);
    }
    [] t_wait.timeout {
        setverdict (inconc);
    }
}

/**
 *
 * @desc
 *   This function broadcast a service guide over
» the flute
 *   protocol
 * @param
 *   p_SGDU The ServiceGuideDeliveryUnit
 * @param
 *   p_SGDU_Id The ServiceGuideDeliveryUnit id
 * @param
 *   p_SGDD_Id The id of the
» ServiceGuideDeliveryDescriptor
 * @param
 *   p_Encoding The encoding type for compression
» in case that
 *   the service guide should be compressed
» otherwise this
 *   parameter should be omitted.
 * @verdict
 *   pass in case the broadcast of the
 *   ServiceGuideDeliveryUnit was successful
 * @verdict
 *   fail in case that the broadcast was not
» successful
 * @verdict
 *   inconc in case of the timer runs out of time
 */
function f_ctrl_broadcastServiceGuide(
    ServiceGuideDeliveryUnit p_SGDU,
    charstring p_SGDU_Id,
    charstring p_SGDD_Id,
    template charstring p_Encoding) runs on
» BCastNetworkControlComponent {

```

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre> // create SGDD var ServiceGuideDeliveryDescriptorType v_SGDD  » :=     f_createSGDD4SGDU(p_SGDU, p_SGDU_Id, » p_SGDD_Id, v_startTime, v_endTime); // change 01 (WK 6): global time » definition      if (PX_BROADCAST_NETWORK_BEARER == e_dvb_h) { » // change 13 (WK 6): send DVB-H bootstrap files via FLUTE         // DVB-H ESG Bootstrapping         f_broadcast_dvbh_bootstrap(0);     } </pre>	<>	<pre> // create SGDD var ServiceGuideDeliveryDescriptorType v_SGDD  » :=     f_createSGDD4SGDU(p_SGDU, p_SGDU_Id, » p_SGDD_Id); </pre>
<pre> // broadcast SGDD f_broadcast_sgdd(v_SGDD, 1, omit);  // broadcast ServiceGuideDeliveryUnit f_broadcast_sgdu(p_SGDU, p_SGDU_Id, 2, » p_Encoding);     } }  group commonFunctions { </pre>	=	<pre> // broadcast SGDD f_broadcast_sgdd(v_SGDD, 1, omit);  // broadcast ServiceGuideDeliveryUnit f_broadcast_sgdu(p_SGDU, p_SGDU_Id, 2, » p_Encoding);     } }  group commonFunctions { </pre>
<pre> // change 14 (WK 6): moved function from » 'AtsBCast_ServiceGuide_Funtions' /**  *  * @desc  * initialize the start and end time of the » BCastNetworkControlComponent.  * @param  * p_startTime The Start Time which is one hour after test » execution  * @param  * p_endTime The End Time which is set to a value one our » after the start time  */ function f_ctrl_InitStartEndTime(UInt p_startTime, UInt » p_endTime) runs on BCastNetworkControlComponent {     v_startTime := p_startTime;     v_endTime := p_endTime; } </pre>	+ -	
/**	=	/**

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre> * * @desc *   This function sends a http response including the » service *   guide. * @param *   p_SGDU The ServiceGuideDeliveryUnit to send * @param *   p_SGDU_Id The ServiceGuideDeliveryUnit id * @param *   p_SGDD_Id The SGDD id * @param *   p_ConnectionId The http connection id * @param *   p_type The http request type * @verdict *   pass in case of that the response was successful send * @verdict *   fail in case of the response was not successfule send * @verdict *   inconc a guard timer runs out of time */ function f_sendServiceGuideHttpResponse( ServiceGuideDeliveryUnit p_SGDU, charstring p_SGDU_Id, charstring p_SGDD_Id, UInt p_ConnectionId, template ServiceGuideRequestType p_type) runs on » BCastNetworkControlComponent {     // create SGDD     var ServiceGuideDeliveryDescriptorType v_SGDD := </pre>	<pre> * * @desc *   This function sends a http response including the » service *   guide. * @param *   p_SGDU The ServiceGuideDeliveryUnit to send * @param *   p_SGDU_Id The ServiceGuideDeliveryUnit id * @param *   p_SGDD_Id The SGDD id * @param *   p_ConnectionId The http connection id * @param *   p_type The http request type * @verdict *   pass in case of that the response was successful send * @verdict *   fail in case of the response was not successfule send * @verdict *   inconc a guard timer runs out of time */ function f_sendServiceGuideHttpResponse( ServiceGuideDeliveryUnit p_SGDU, charstring p_SGDU_Id, charstring p_SGDD_Id, UInt p_ConnectionId, template ServiceGuideRequestType p_type) runs on » BCastNetworkControlComponent {     // create SGDD     var ServiceGuideDeliveryDescriptorType v_SGDD := </pre>
<pre>         f_createSGDD4SGDU(p_SGDU, p_SGDU_Id, » p_SGDD_Id, v_startTime, v_endTime); // change 01 (WK 6): global time » definition </pre>	<pre>         f_createSGDD4SGDU(p_SGDU, p_SGDU_Id, » p_SGDD_Id); </pre>
<pre>         // note: we assume in case the service guide request » type is omitted we should send both sgdu and sgdd         var ServiceGuidePayload v_Payload;          if(p_type == e_sgdd) {             v_Payload := » valueof(m_def_ServiceGuidePayload_SGDD(v_SGDD));         }         else if (p_type == e_sgdu) {             v_Payload := » valueof(m_def_ServiceGuidePayload_SGDU(p_SGDU)); </pre>	<pre>         // note: we assume in case the service guide request » type is omitted we should send both sgdu and sgdd         var ServiceGuidePayload v_Payload;          if(p_type == e_sgdd) {             v_Payload := » valueof(m_def_ServiceGuidePayload_SGDD(v_SGDD));         }         else if (p_type == e_sgdu) {             v_Payload := » valueof(m_def_ServiceGuidePayload_SGDU(p_SGDU)); </pre>

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre>         }         else {             v_Payload := » valueof(m_def_ServiceGuidePayload_SGDU_SGDD(p_SGDU, v_SGDD));         }          ic.send(m_def_HttpResponseRequest(p_ConnectionId, » v_Payload));          t_wait.start(PX_TWAIT);         alt {             [] ic.receive(HttpResponseResponse:{0, *}) {                 t_wait.stop;                 setverdict(pass);             }             [] ic.receive {                 t_wait.stop;                 setverdict(fail);             }             [] t_wait.timeout {                 setverdict(inconc);             }         }     } } </pre>		<pre>         }         else {             v_Payload := » valueof(m_def_ServiceGuidePayload_SGDU_SGDD(p_SGDU, v_SGDD));         }          ic.send(m_def_HttpResponseRequest(p_ConnectionId, » v_Payload));          t_wait.start(PX_TWAIT);         alt {             [] ic.receive(HttpResponseResponse:{0, *}) {                 t_wait.stop;                 setverdict(pass);             }             [] ic.receive {                 t_wait.stop;                 setverdict(fail);             }             [] t_wait.timeout {                 setverdict(inconc);             }         }     } } </pre>
<pre> // change 13 (WK 6): send DVB-H bootstrap files via FLUTE /**  *  * @desc  *   this function sends the DVB-H bootstrapping over the  *   flute protocoal and expects a successful flute » response back.  *  * @param  *   p_Id The Flute session id  * @verdict  *   pass in case the DVB-H bootstrap could be successful » send  *  * @verdict  *   fail in case the DVB-H bootstrap could not be » successful send  *  * @verdict  *   inconc in case a guard timer expires  */ function f_broadcast_dvbh_bootstrap(UInt p_Id) runs on » BCastNetworkControlComponent {     var StartFluteSessionRequest v_SessionRequest; </pre>	+-	

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn

(Fortsetzung)

```

        var ESGProviderDiscoveryType v_epdd :=
» valueof(m_def_esg_prov_disc(
        PX_DVBH_ESG_PROV_DISC_URI,
        PX_DVBH_ESG_PROV_DISC_NAME,
        PX_DVBH_ESG_PROV_DISC_ID));
        var ESGAccessType v_ead :=
» valueof(m_def_esg_access_ipv4(
        PX_SRC_IP,
        PX_SGDD_FLUTE_SESSION_IP,
        PX_SGDD_FLUTE_SESSION_PORT));

        var FDT_InstanceType v_fdt :=
» f_createFDT4DVBH(int2str(v_endTime));

        v_SessionRequest :=
» valueof(m_def_StartFluteSessionRequest(
        p_Id,
        PX_DVB_H_FLUTE_SESSION_IP,
        PX_DVB_H_FLUTE_SESSION_PORT,
        v_fdt,

» m_DVBH_ESG_Descriptor_FluteSessionPayload(v_epdd, v_ead)));

        ac.send(v_SessionRequest);
        t_wait.start(PX_TWAIT);
        alt {
» *)} {
            [] ac.receive(StartFluteSessionResponse: {0,
                t_wait.stop;
                setverdict(pass);
            }
            [] ac.receive {
                t_wait.stop;
                setverdict(fail);
            }
            [] t_wait.timeout {
                setverdict(inconc);
            }
        }
    }
}

```

```

/**
 *
 * @desc
 *     this function sends a ServiceGuideDeliveryDescriptor
» over the

```

=

```

/**
 *
 * @desc
 *     this function sends a ServiceGuideDeliveryDescriptor
» over the

```

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre> *      flute protocol and expects a successful flute » response back. * @param *      p_SGDD The SGDD which should be send * @param *      p_Id The Flute session id * @param *      p_Encoding The encoding attribute of the *      ServiceGuideDeliveryDescriptor * @verdict *      pass in case the SGDD could be successful send * @verdict *      fail in case the SGDD could not be successful send * @verdict *      inconc in case a guard timer expires */ function f_broadcast_sgdd(     ServiceGuideDeliveryDescriptorType p_SGDD,     UInt p_Id,     template charstring p_Encoding) runs on » BCastNetworkControlComponent {     var StartFluteSessionRequest v_SessionRequest; </pre>		<pre> *      flute protocol and expects a successful flute » response back. * @param *      p_SGDD The SGDD which should be send * @param *      p_Id The Flute session id * @param *      p_Encoding The encoding attribute of the *      ServiceGuideDeliveryDescriptor * @verdict *      pass in case the SGDD could be successful send * @verdict *      fail in case the SGDD could not be successful send * @verdict *      inconc in case a guard timer expires */ function f_broadcast_sgdd(     ServiceGuideDeliveryDescriptorType p_SGDD,     UInt p_Id,     template charstring p_Encoding) runs on » BCastNetworkControlComponent {     var StartFluteSessionRequest v_SessionRequest; </pre>
<pre>     var FDT_InstanceType v_fdt := f_createFDT4SGDD("urn:o » ma:bcast:sgdd:&amp;int2str(p_Id), p_Encoding, int2str(v_endTime)); » // change 15 (WK 6): dynamic setting of content location and expiry time </pre>	<>	<pre>     var FDT_InstanceType v_fdt := f_createFDT4SGDD(PX_SGD » D_ID, p_Encoding, "set the expire time here"); </pre>
<pre>     v_SessionRequest :=         valueof(m_def_StartFluteSessionRequest(             p_Id, </pre>	=	<pre>     v_SessionRequest :=         valueof(m_def_StartFluteSessionRequest(             p_Id, </pre>
<pre>             PX_SGDD_FLUTE_SESSION_IP, » // change 16 (WK 6): usage of PIXIT parameter for flute session             PX_SGDD_FLUTE_SESSION_PORT, » // change 16 (WK 6): usage of PIXIT parameter for flute session </pre>	+-	
<pre>             v_fdt,             m_SGDD_FluteSessionPayload(p_SGDD)));         ac.send (v_SessionRequest);         t_wait.start(PX_TWAIT);         alt {             [] ac.receive (StartFluteSessionResponse: {0, » *}) {                 t_wait.stop;                 setverdict (pass);             }             [] ac.receive { </pre>	=	<pre>             v_fdt,             m_SGDD_FluteSessionPayload(p_SGDD)));         ac.send (v_SessionRequest);         t_wait.start(PX_TWAIT);         alt {             [] ac.receive (StartFluteSessionResponse: {0, » *}) {                 t_wait.stop;                 setverdict (pass);             }             [] ac.receive { </pre>



Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre>                                 t_wait.stop;                                 setverdict (fail);                             }                             [] t_wait.timeout {                                 setverdict (inconc);                             }                         }                     }  /**  *  * @desc  *   this function sends a ServiceGuideDeliveryUnit over » the flute  *   protocoal and expect a successful flute response back.  * @param  *   p_SGDU The ServiceGuideDeliveryUnit which should be » send  * @param  *   p_SGDU_Id the id of the ServiceGuideDeliveryUnit  * @param  *   p_Id The flute session id  * @param  *   p_Encoding the encoding of the » ServiceGuideDeliveryUnit  * @verdict  *   pass in case the SGDD could be successful send  * @verdict  *   fail in case the SGDD could not be successful  * @verdict  *   inconc in case a guard timer expires  */ function f_broadcast_sgdu(     ServiceGuideDeliveryUnit p_SGDU,     AnyUri p_SGDU_Id,     UInt p_Id,     template charstring p_Encoding) runs on » BCastNetworkControlComponent { </pre>		<pre>                                 t_wait.stop;                                 setverdict (fail);                             }                             [] t_wait.timeout {                                 setverdict (inconc);                             }                         }                     }  /**  *  * @desc  *   this function sends a ServiceGuideDeliveryUnit over » the flute  *   protocoal and expect a successful flute response back.  * @param  *   p_SGDU The ServiceGuideDeliveryUnit which should be » send  * @param  *   p_SGDU_Id the id of the ServiceGuideDeliveryUnit  * @param  *   p_Id The flute session id  * @param  *   p_Encoding the encoding of the » ServiceGuideDeliveryUnit  * @verdict  *   pass in case the SGDD could be successful send  * @verdict  *   fail in case the SGDD could not be successful  * @verdict  *   inconc in case a guard timer expires  */ function f_broadcast_sgdu(     ServiceGuideDeliveryUnit p_SGDU,     AnyUri p_SGDU_Id,     UInt p_Id,     template charstring p_Encoding) runs on » BCastNetworkControlComponent { </pre>
<pre>                                 var FDT_InstanceType v_fdt := f_createFDT4SGDU("urn:o » ma:bcast:sgdu:&amp;int2str(p_Id), p_Encoding, int2str(v_endTime)); // » change 15 (WK 6): dynamic setting of content location and expiry time </pre>	<>	<pre>                                 var FDT_InstanceType v_fdt := f_createFDT4SGDU(p_SGDU » _Id, p_Encoding, "set the expire time here"); </pre>
<pre>                                 var StartFluteSessionRequest v_SessionRequest;                                 v_SessionRequest :=                                     valueof(m_def_StartFluteSessionRequest( </pre>	=	<pre>                                 var StartFluteSessionRequest v_SessionRequest;                                 v_SessionRequest :=                                     valueof(m_def_StartFluteSessionRequest( </pre>

Datei: LibBCast\_BCastNetworkControl\_Functions.ttcn  
(Fortsetzung)

<pre>                 p_Id,                 PX_SGDU_FLUTE_SESSION_IP, » // change 16 (WK 6): usage of PIXIT parameter for flute session                 PX_SGDU_FLUTE_SESSION_PORT, » // change 16 (WK 6): usage of PIXIT parameter for flute session             v_fdt,             m_SGDU_FluteSessionPayload(p_SGDU));         dc.send (v_SessionRequest);         t_wait.start(PX_TWAIT);         alt {             [] dc.receive (StartFluteSessionResponse: {0, » *)) {                 t_wait.stop;                 setverdict (pass);             }             [] dc.receive {                 t_wait.stop;                 setverdict (fail);             }             [] t_wait.timeout {                 setverdict (inconc);             }         }     } } </pre>	<pre>                 p_Id,                 PX_SGDU_FLUTE_SESSION_IP, » // change 16 (WK 6): usage of PIXIT parameter for flute session                 PX_SGDU_FLUTE_SESSION_PORT, » // change 16 (WK 6): usage of PIXIT parameter for flute session             v_fdt,             m_SGDU_FluteSessionPayload(p_SGDU));         dc.send (v_SessionRequest);         t_wait.start(PX_TWAIT);         alt {             [] dc.receive (StartFluteSessionResponse: {0, » *)) {                 t_wait.stop;                 setverdict (pass);             }             [] dc.receive {                 t_wait.stop;                 setverdict (fail);             }             [] t_wait.timeout {                 setverdict (inconc);             }         }     } } </pre>
---	---

Datei: LibBCast\_Common\_Templates.ttcn

<pre> /**  *  * @author  *   ETSI CTI OMA BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module provides common templates.  */ module LibBCast_Common_Templates {     import from LibBCast_ServicePrimitives_TypesAndValues all;     import from LibBCast_ServiceGuide_TypesAndValues all;     import from LibBCast_ServicePrimitives_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_BasicTypesAndValues all;     import from LibCommon_TextStrings all; </pre>	<pre> /**  *  * @author  *   ETSI CTI OMA BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module provides common templates.  */ module LibBCast_Common_Templates {     import from LibBCast_ServicePrimitives_TypesAndValues all;     import from LibBCast_ServiceGuide_TypesAndValues all;     import from LibBCast_ServicePrimitives_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_BasicTypesAndValues all; </pre>
--	---

Datei: LibBCast\_Common\_Templates.ttcn (Fortsetzung)

» change 17 (WK 6): additional parameter necessary for flute session setup		
<pre> group flute {     /**      *      * @desc      *   This template creates a StartFluteSessionRequest.      * @param      *   p_Id The flute session id      * @param      *   p_FDT The File Delivery Table (FDT)      * @param      *   p_Payload The payload which should be send      */     template StartFluteSessionRequest » m_def_StartFluteSessionRequest(         UInt p_Id, </pre>	=	<pre> group flute {     /**      *      * @desc      *   This template creates a StartFluteSessionRequest.      * @param      *   p_Id The flute session id      * @param      *   p_FDT The File Delivery Table (FDT)      * @param      *   p_Payload The payload which should be send      */     template StartFluteSessionRequest » m_def_StartFluteSessionRequest(         UInt p_Id, </pre>
<pre>         charstring p_IP, » // change 17 (WK 6): additional parameter necessary for flute session setup         UInt16 p_Port, » // change 17 (WK 6): additional parameter necessary for flute session setup </pre>	+-	
<pre>         template FDT_InstanceType p_FDT,         template FluteSessionPayloadType p_Payload) := { </pre>	=	<pre>         template FDT_InstanceType p_FDT,         template FluteSessionPayloadType p_Payload) := { </pre>
<pre>         FluteSessionId := p_Id,         FluteSessionIp := p_IP, » // change 17 (WK 6): additional parameter necessary for flute session setup         FluteSessionIpType := IPv4, » // change 17 (WK 6): additional parameter necessary for flute session setup         FluteSessionPort := p_Port, » // change 17 (WK 6): additional parameter necessary for flute session setup </pre>	<>	<pre>         FluteSessionId := p_Id, </pre>
<pre>         FDT_Instance := p_FDT,         FluteSessionPayload := p_Payload     }      /**      *      * @desc      *   This template creates a Flute Session Payload      * @param      *   p_SGDD The SGDD which should be included      */     template FluteSessionPayloadType » m_SGDD_FluteSessionPayload(ServiceGuideDeliveryDescriptorType p_SGDD) := {         ServiceGuideDeliveryDescriptors := { p_SGDD }     }      /** </pre>	=	<pre>         FDT_Instance := p_FDT,         FluteSessionPayload := p_Payload     }      /**      *      * @desc      *   This template creates a Flute Session Payload      * @param      *   p_SGDD The SGDD which should be included      */     template FluteSessionPayloadType » m_SGDD_FluteSessionPayload(ServiceGuideDeliveryDescriptorType p_SGDD) := {         ServiceGuideDeliveryDescriptors := { p_SGDD }     }      /** </pre>

Datei: LibBCast\_Common\_Templates.ttcn (Fortsetzung)

<pre> * * @desc *   This template creates a Flute Session Payload. * @param *   p_SGDU The SGDU which should be included */ template FluteSessionPayloadType » m_SGDU_FluteSessionPayload(ServiceGuideDeliveryUnit p_SGDU) := {     ServiceGuideDeliveryUnits := { p_SGDU } } </pre>	<>	<pre> * * @desc *   This template creates a Flute Session Payload. * @param *   p_SGDU The SGDU wich should be included */ template FluteSessionPayloadType » m_SGDU_FluteSessionPayload(ServiceGuideDeliveryUnit p_SGDU) := {     ServiceGuideDeliveryUnits := { p_SGDU } } </pre>
<pre> // change 13 (WK 6): send DVB-H bootstrap files via FLUTE /** * * @desc *   This template creates a Flute Session Payload. * @param *   p_EPDD The ESG Provider Discovery Descriptor which » should be included * @param *   p_EAD The ESG Access Descriptor which should be » included */ template FluteSessionPayloadType » m_DVBH_ESG_Descriptor_FluteSessionPayload(ESGProviderDiscoveryType p_EPDD, » ESGAccessType p_EAD) := {     ESGDescriptor := {         p_EPDD,         p_EAD     } } </pre>	+ -	
<pre> }  group http { /** * * @desc This is a default HttpPostIndication template wich » can be used as recieve template. The mandatory fields are set to any and » the optional fields are set to any or omit. */ template HttpPostIndication mw_def_HttpPostIndication := {     connectionId := ?,     serviceGuideRequestType := *,     keyValuePairs := omit,     StreamingReceptionReport := omit } </pre>	=	<pre> }  group http { /** * * @desc This is a default HttpPostIndication template wich » can be used as recieve template. The mandatory fields are set to any and » the optional fields are set to any or omit. */ template HttpPostIndication mw_def_HttpPostIndication := {     connectionId := ?,     serviceGuideRequestType := *,     keyValuePairs := omit,     StreamingReceptionReport := omit } </pre>

Datei: LibBCast\_Common\_Templates.ttcn (Fortsetzung)

```

    /**
     *
     * @desc This template creates a HttpResponseRequest.
     * @param p_ConnectionId The http connection id
     * @param p_ServiceGuidePayload The payload which should be
» send.
    */
    template HttpResponseRequest m_def_HttpResponseRequest(
        UInt p_ConnectionId,
        ServiceGuidePayload p_ServiceGuidePayload) := {
        connectionId := p_ConnectionId,
        responseCode := 200,
        responsePayload := {
            ServiceGuidePayload := p_ServiceGuidePayload
        }
    }

    /**
     *
     * @desc Default ServiceGuidePayload template wich includes
» only a SGDU
     * @param p_SGDU the included SGDU
    */
    template ServiceGuidePayload
» m_def_ServiceGuidePayload_SGDU(ServiceGuideDeliveryUnit p_SGDU) := {
        serviceGuideResponse := {
            status := 0,
            ServiceGuideDeliveryDescriptors := omit,
            privateExt := omit },
        serviceGuideDeliveryUnit := p_SGDU
    }

    /**
     *
     * @desc Default ServiceGuidePayload template wich includes
» only a SGDD
     * @param p_SGDD the included SGDD
    */
    template ServiceGuidePayload
» m_def_ServiceGuidePayload_SGDD(ServiceGuideDeliveryDescriptorType p_SGDD)
» := {
        serviceGuideResponse := {
            status := 0,
            ServiceGuideDeliveryDescriptors := { p_SGDD
» },
            privateExt := omit },
        serviceGuideDeliveryUnit := omit
    }

```

```

    /**
     *
     * @desc This template creates a HttpResponseRequest.
     * @param p_ConnectionId The http connection id
     * @param p_ServiceGuidePayload The payload which should be
» send.
    */
    template HttpResponseRequest m_def_HttpResponseRequest(
        UInt p_ConnectionId,
        ServiceGuidePayload p_ServiceGuidePayload) := {
        connectionId := p_ConnectionId,
        responseCode := 200,
        responsePayload := {
            ServiceGuidePayload := p_ServiceGuidePayload
        }
    }

    /**
     *
     * @desc Default ServiceGuidePayload template wich includes
» only a SGDU
     * @param p_SGDU the included SGDU
    */
    template ServiceGuidePayload
» m_def_ServiceGuidePayload_SGDU(ServiceGuideDeliveryUnit p_SGDU) := {
        serviceGuideResponse := {
            status := 0,
            ServiceGuideDeliveryDescriptors := omit,
            privateExt := omit },
        serviceGuideDeliveryUnit := p_SGDU
    }

    /**
     *
     * @desc Default ServiceGuidePayload template wich includes
» only a SGDD
     * @param p_SGDD the included SGDD
    */
    template ServiceGuidePayload
» m_def_ServiceGuidePayload_SGDD(ServiceGuideDeliveryDescriptorType p_SGDD)
» := {
        serviceGuideResponse := {
            status := 0,
            ServiceGuideDeliveryDescriptors := { p_SGDD
» },
            privateExt := omit },
        serviceGuideDeliveryUnit := omit
    }

```

Datei: LibBCast\_Common\_Templates.ttcn (Fortsetzung)

```

    }

    /**
     *
     * @desc Default ServiceGuidePayload template wich includes a
» SGDU and SGDD
     * @param p_SGDU the included SGDU
     * @param p_SGDD the included SGDD
     */
    template ServiceGuidePayload
» m_def_ServiceGuidePayload_SGDU_SGDD(
        ServiceGuideDeliveryUnit p_SGDU,
        ServiceGuideDeliveryDescriptorType p_SGDD) := {
        serviceGuideResponse := {
            status := 0,
            ServiceGuideDeliveryDescriptors := { p_SGDD
» },
            privateExt := omit },
        serviceGuideDeliveryUnit := p_SGDU
    }

}

group streaming {

    /**
     *
     * @desc This is a default template for the
» StartStreamingRequest.
     * @param p_StreamId The stream id
     * @param p_ContentList the list of elemts which should be
» streamed
     * @param p_SDP The used SDP
     */
    /**
     *
     * @desc
     * This is a default template for the
» StartStreamingRequest.
     * @param
     * p_StreamId The stream id
     * @param
     * p_FileName The name of the file which should be
» streamed
     * @param
     * p_SDP The used SDP
     */

```

```

    }

    /**
     *
     * @desc Default ServiceGuidePayload template wich includes a
» SGDU and SGDD
     * @param p_SGDU the included SGDU
     * @param p_SGDD the included SGDD
     */
    template ServiceGuidePayload
» m_def_ServiceGuidePayload_SGDU_SGDD(
        ServiceGuideDeliveryUnit p_SGDU,
        ServiceGuideDeliveryDescriptorType p_SGDD) := {
        serviceGuideResponse := {
            status := 0,
            ServiceGuideDeliveryDescriptors := { p_SGDD
» },
            privateExt := omit },
        serviceGuideDeliveryUnit := p_SGDU
    }

}

group streaming {

    /**
     *
     * @desc This is a default template for the
» StartStreamingRequest.
     * @param p_StreamId The stream id
     * @param p_ContentList the list of elemts which should be
» streamed
     * @param p_SDP The used SDP
     */
    /**
     *
     * @desc
     * This is a default template for the
» StartStreamingRequest.
     * @param
     * p_StreamId The stream id
     * @param
     * p_FileName The name of the file which should be
» streamed
     * @param
     * p_SDP The used SDP
     */

```

## Datei: LibBCast\_Common\_Templates.ttcn (Fortsetzung)

```

        template StartStreamingRequest m_def_StartStreamingRequest(
            UInt p_StreamId,
            charstring p_FileName,
            charstring p_SDP) := {

            streamId := p_StreamId,
            fileName := p_FileName,
            sdp := p_SDP,
            contentProtection := omit,
            serviceProtection := omit

        }

    /**
     *
     * @desc This is a default template for a
    » StopStreamingRequest
     * @param p_StreamId The id of the stream wich should stopped
     */
    template StopStreamingRequest m_def_StopStreamingRequest(UInt
    » p_StreamId) := {
        streamId := p_StreamId
    }

    group transfer {
    /**
     *
     * @desc This is a default template for a
    » StartFileTransferRequest.
     * @param p_TransferId The transfer id
     * @param p_FileList The list of file
     * @param p_Sdp The used SDP
     */
    template StartFileTransferRequest
    » m_def_StartFileTransferRequest(
        UInt p_TransferId,
        TransferFileList p_FileList,
        charstring p_Sdp) := {

        transferId := p_TransferId,
        fileList := p_FileList,
        sdp := p_Sdp,
        contentProtection := omit,
        serviceProtection := omit

    }

    /**

```

```

        template StartStreamingRequest m_def_StartStreamingRequest(
            UInt p_StreamId,
            charstring p_FileName,
            charstring p_SDP) := {

            streamId := p_StreamId,
            fileName := p_FileName,
            sdp := p_SDP,
            contentProtection := omit,
            serviceProtection := omit

        }

    /**
     *
     * @desc This is a default template for a
    » StopStreamingRequest
     * @param p_StreamId The id of the stream wich should stopped
     */
    template StopStreamingRequest m_def_StopStreamingRequest(UInt
    » p_StreamId) := {
        streamId := p_StreamId
    }

    group transfer {
    /**
     *
     * @desc This is a default template for a
    » StartFileTransferRequest.
     * @param p_TransferId The transfer id
     * @param p_FileList The list of file
     * @param p_Sdp The used SDP
     */
    template StartFileTransferRequest
    » m_def_StartFileTransferRequest(
        UInt p_TransferId,
        TransferFileList p_FileList,
        charstring p_Sdp) := {

        transferId := p_TransferId,
        fileList := p_FileList,
        sdp := p_Sdp,
        contentProtection := omit,
        serviceProtection := omit

    }

    /**

```

## Datei: LibBCast\_Common\_Templates.ttcn (Fortsetzung)

```

        *
        * @desc This is a default template for a
» StopFileTransferRequest
        * @param p_TransferId The id of the stream wich should
» stopped
        */
        template StopFileTransferRequest
» m_def_StopFileTransferRequest(UInt p_TransferId) := {
            transferId := p_TransferId
        }

        /**
        *
        * @desc This is a default ZipFile template
        * @param p_Name The name of the zip file
        * @param p_FileList The content file list of the zip file
        */
        template ZipFile m_def_ZipFile(charstring p_Name,
» FileReferenceList p_FileList) := {
            name := p_Name,
            fileList := p_FileList
        }
    }
}

```

```

        *
        * @desc This is a default template for a
» StopFileTransferRequest
        * @param p_TransferId The id of the stream wich should
» stopped
        */
        template StopFileTransferRequest
» m_def_StopFileTransferRequest(UInt p_TransferId) := {
            transferId := p_TransferId
        }

        /**
        *
        * @desc This is a default ZipFile template
        * @param p_Name The name of the zip file
        * @param p_FileList The content file list of the zip file
        */
        template ZipFile m_def_ZipFile(charstring p_Name,
» FileReferenceList p_FileList) := {
            name := p_Name,
            fileList := p_FileList
        }
    }
}

```

## Datei: LibBCast\_Interface.ttcn

```

/**
 *
 * @author
 *   ETSI CTI OMA BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies port and component types that specify the
 *   interface required by BCAST library functions.
 */
module LibBCast_Interface {
    import from LibBCast_UpperTesterPrimitives_TypesAndValues all ;
    import from LibBCast_ServicePrimitives_TypesAndValues all ;
    import from LibCommon_BasicTypesAndValues { // change 01 (WK
» 6): global time definition
        type UInt;
    }

    group libraryComponentTypes {

```

```

= /**
 *
 * @author
 *   ETSI CTI OMA BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies port and component types that specify the
 *   interface required by BCAST library functions.
 */
module LibBCast_Interface {
    import from LibBCast_UpperTesterPrimitives_TypesAndValues all ;
    import from LibBCast_ServicePrimitives_TypesAndValues all ;
    import from LibCommon_BasicTypesAndValues { // change 01 (WK
» 6): global time definition
        type UInt;
    }

    group libraryComponentTypes {

```



Datei: LibBCast\_Interface.ttcn (Fortsetzung)

<pre>       /**        *        * @desc        *   This type of component is used by library functions » which        *   specify BCAST control behavior, i.e., handling of » BCAST        *   SGDDs, SGDUs, and Interactivity Media Documents.        */ type component BCastNetworkControlComponent {     port BroadcastPort ac;     port BroadcastPort dc;     port InteractionPort ic;     port SmsPort sms;     port MmsPort mms;     timer t_wait; </pre>		<pre>       /**        *        * @desc        *   This type of component is used by library functions » which        *   specify BCAST control behavior, i.e., handling of » BCAST        *   SGDDs, SGDUs, and Interactivity Media Documents.        */ type component BCastNetworkControlComponent {     port BroadcastPort ac;     port BroadcastPort dc;     port InteractionPort ic;     port SmsPort sms;     port MmsPort mms;     timer t_wait; </pre>
<pre>       var UInt v_startTime := 0; » change 01 (WK 6): global time definition       var UInt v_endTime := 0; » change 01 (WK 6): global time definition </pre>	+-	
<pre>     }      /**      *      * @desc      *   This component type is used by library functions which      *   control streaming or file servers.      */ type component BCastNetworkDataComponent {     port StreamServerCtrlPort ssc;     port FileServerControlPort fsc;     timer t_wait; }      /**      *      * @desc      *   This component type is used by library functions which      *   specify upper tester behavior.      */ type component UpperTesterComponent {     port UtpPort utp;     timer t_wait; }  } </pre>	=	<pre>     }      /**      *      * @desc      *   This component type is used by library functions which      *   control streaming or file servers.      */ type component BCastNetworkDataComponent {     port StreamServerCtrlPort ssc;     port FileServerControlPort fsc;     timer t_wait; }      /**      *      * @desc      *   This component type is used by library functions which      *   specify upper tester behavior.      */ type component UpperTesterComponent {     port UtpPort utp;     timer t_wait; }  } </pre>

Datei: LibBCast\_Interface.ttcn (Fortsetzung)

```

    group portDefinitions {
        /**
         * @desc
         * This port type is used to configure and interact with the
» broadcast
         * channel.
        */
        type port BroadcastPort message {
            inout StartFluteSessionRequest;
            inout StartFluteSessionResponse;
        }

        /**
         * @desc
         * This port type is used to configure and interact with the
» interaction
         * channel.
        */
        type port InteractionPort message {
            inout HttpResponseRequest;
            inout HttpResponseResponse;
            inout HttpSubscriptionRequestIndication;
            inout HttpPostIndication;
        }

        /**
         * @desc
         * This port type is used to configure streaming servers.
        */
        type port StreamServerCtrlPort message {
            inout StartStreamingRequest;
            inout StartStreamingResponse;
            inout StopStreamingRequest;
            inout StopStreamingResponse;
        }

        /**
         * @desc
         * This port type is used to interact with BCAST upper
» testers.
        */
        type port UtpPort message {
            inout PowerOnTerminalRequest;
            inout PowerOffTerminalRequest;
            inout RunBCastApplicationRequest;
            inout GetServiceGuideRequest;

```

```

    group portDefinitions {
        /**
         * @desc
         * This port type is used to configure and interact with the
» broadcast
         * channel.
        */
        type port BroadcastPort message {
            inout StartFluteSessionRequest;
            inout StartFluteSessionResponse;
        }

        /**
         * @desc
         * This port type is used to configure and interact with the
» interaction
         * channel.
        */
        type port InteractionPort message {
            inout HttpResponseRequest;
            inout HttpResponseResponse;
            inout HttpSubscriptionRequestIndication;
            inout HttpPostIndication;
        }

        /**
         * @desc
         * This port type is used to configure streaming servers.
        */
        type port StreamServerCtrlPort message {
            inout StartStreamingRequest;
            inout StartStreamingResponse;
            inout StopStreamingRequest;
            inout StopStreamingResponse;
        }

        /**
         * @desc
         * This port type is used to interact with BCAST upper
» testers.
        */
        type port UtpPort message {
            inout PowerOnTerminalRequest;
            inout PowerOffTerminalRequest;
            inout RunBCastApplicationRequest;
            inout GetServiceGuideRequest;

```

Datei: LibBCast\_Interface.ttcn (Fortsetzung)

```

        inout UpdateServiceGuideRequest;
        inout CheckServiceRequest;
        inout SelectServiceRequest;
        inout CheckContentPresenceRequest;
        inout SelectContentRequest;
        inout SelectLanguageRequest;
        inout ClearServiceGuideCacheRequest;
        inout GenericUtsResponse;
        inout CheckBrowserRequest;
        inout CheckPreviewRequest;
        inout CheckFileRequest;
        inout CheckVideoRequest;
        inout CheckPurchaseInfoRequest;
        inout CheckLanguageRequest;
        inout CheckInteractivityRequest;
        inout CheckInteractivityChoicesRequest;
        inout PurchaseServiceRequest;
        inout SelectFileRequest;
        inout SelectInteractivityRequest;
        inout CheckFileReceivedRequest;
        inout SelectInteractivityChoiceRequest;
        inout UseServiceRequest;
    }

    /**
     *
     * @desc This port type is used to interact with sms centers
     */
    type port SmsPort message {
        inout SmsGetIndication;
    }

    /**
     * @desc This port type is used to interact with mms centers
     */
    type port MmsPort message {
        inout MmsGetIndication;
    }

    /**
     *
     * @desc This port type is used to interact with the file
    » servers in order
     * to deliver files to a terminal
     */
    type port FileServerControlPort message {
        inout StartFileTransferRequest;

```

```

        inout UpdateServiceGuideRequest;
        inout CheckServiceRequest;
        inout SelectServiceRequest;
        inout CheckContentPresenceRequest;
        inout SelectContentRequest;
        inout SelectLanguageRequest;
        inout ClearServiceGuideCacheRequest;
        inout GenericUtsResponse;
        inout CheckBrowserRequest;
        inout CheckPreviewRequest;
        inout CheckFileRequest;
        inout CheckVideoRequest;
        inout CheckPurchaseInfoRequest;
        inout CheckLanguageRequest;
        inout CheckInteractivityRequest;
        inout CheckInteractivityChoicesRequest;
        inout PurchaseServiceRequest;
        inout SelectFileRequest;
        inout SelectInteractivityRequest;
        inout CheckFileReceivedRequest;
        inout SelectInteractivityChoiceRequest;
        inout UseServiceRequest;
    }

    /**
     *
     * @desc This port type is used to interact with sms centers
     */
    type port SmsPort message {
        inout SmsGetIndication;
    }

    /**
     * @desc This port type is used to interact with mms centers
     */
    type port MmsPort message {
        inout MmsGetIndication;
    }

    /**
     *
     * @desc This port type is used to interact with the file
    » servers in order
     * to deliver files to a terminal
     */
    type port FileServerControlPort message {
        inout StartFileTransferRequest;

```

## Datei: LibBCast\_Interface.ttcn (Fortsetzung)

<pre>         inout StartFileTransferResponse;         inout StopFileTransferRequest;         inout StopFileTransferResponse;     } } } // end module LibBCast_Interface </pre>	<pre>         inout StartFileTransferResponse;         inout StopFileTransferRequest;         inout StopFileTransferResponse;     } } } // end module LibBCast_Interface </pre>
---	---

## Datei: LibBCast\_ModuleParameters.ttcn

<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies BCAST library specific module parameters  *   and their default values which can be still modified in their value as  *   late as just prior to test case execution.  *   The parameters in this file OMA BCAST specific.  *   Module parameters realize so called PIXIT.  */ module LibBCast_ModuleParameters {     import from LibBCast_UpperTesterPrimitives_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_TextStrings {         const c_CRLF     }      group GlobalConfiguration {         /**          *          * @desc          *   Specifies the broadcast network bearer to be used by » the test case, i.e., DVB-H, MBMS or BMCMS          */         modulepar BroadcastBearerType PX_BROADCAST_NETWORK_BEARER := » e_dvb_h;          /**          *          * @desc          *   Specifies the interaction network bearer to be used by » the test case, i.e., GPRS/GSM, GPRS/CDMA, GPRS/UMTS          */         modulepar InteractiveBearerType PX_INTERACTION_NETWORK_BEARER » :=             e_gsm_gprs; </pre>	=	<pre> /**  *  * @author  *   ETSI CTI BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module specifies BCAST library specific module parameters  *   and their default values which can be still modified in their value as  *   late as just prior to test case execution.  *   The parameters in this file OMA BCAST specific.  *   Module parameters realize so called PIXIT.  */ module LibBCast_ModuleParameters {     import from LibBCast_UpperTesterPrimitives_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_TextStrings {         const c_CRLF     }      group GlobalConfiguration {         /**          *          * @desc          *   Specifies the broadcast network bearer to be used by » the test case, i.e., DVB-H, MBMS or BMCMS          */         modulepar BroadcastBearerType PX_BROADCAST_NETWORK_BEARER := » e_dvb_h;          /**          *          * @desc          *   Specifies the interaction network bearer to be used by » the test case, i.e., GPRS/GSM, GPRS/CDMA, GPRS/UMTS          */         modulepar InteractiveBearerType PX_INTERACTION_NETWORK_BEARER » :=             e_gsm_gprs; </pre>
--	---	--

Datei: LibBCast\_ModuleParameters.ttcn (Fortsetzung)

<pre>     }      group ServiceGuideIds {         group ServiceGuideDeliveryDescriptor {             /**              *              * @desc              *     Specifies the URI that uniquely identifies the » SGDD within one specific SG              */             modulepar charstring PX_SGDD_ID :=                 "bcast://bcast.openmobilealliance.org/SGDD/sg » dd_id";           // change 18 (WK 6): string descriptors changed to » openmobilealliance.org         }     }      group ServiceGuideDeliveryUnit {         /**          *          * @desc          *     Specifies the ID of the SGDU.          */         modulepar charstring PX_SGDU_ID :=             "bcast://bcast.openmobilealliance.org/SGDU/sg » du_id";           // change 18 (WK 6): string descriptors changed to » openmobilealliance.org     } } </pre>		<pre>     }      group ServiceGuideIds {         group ServiceGuideDeliveryDescriptor {             /**              *              * @desc              *     Specifies the URI that uniquely identifies the » SGDD within one specific SG              */             modulepar charstring PX_SGDD_ID :=                 "bcast://bcast.testingtech.com/SGDD/sgdd_id";         }     }      group ServiceGuideDeliveryUnit {         /**          *          * @desc          *     Specifies the ID of the SGDU.          */         modulepar charstring PX_SGDU_ID :=             "bcast://bcast.testingtech.com/SGDU/sgdu_id";     } } </pre>
<pre> // change 19 (WK 6): default values for DVB-H parameters as PIXIT group DvbhBootstrapConfiguration {     /** @desc Specifies the DVB-H Bootstrap ESG Provider » Discovery URI */     modulepar charstring PX_DVBH_ESG_PROV_DISC_URI := » "http://www.openmobilealliance.org/bcast";     /** @desc Specifies the DVB-H Bootstrap ESG Provider » Discovery Name */     modulepar charstring PX_DVBH_ESG_PROV_DISC_NAME := "IPDC";     /** @desc Specifies the DVB-H Bootstrap ESG Provider » Discovery ID */     modulepar charstring PX_DVBH_ESG_PROV_DISC_ID := "1";      /** @desc Specifies the source IPv4 Address */     modulepar charstring PX_SRC_IP := "10.10.0.1"; } </pre>		
<pre> } // end module LibBCast_ModuleParameters </pre>		<pre> } // end module LibBCast_ModuleParameters </pre>

Datei: LibBCast\_ServiceGuide\_Templates.ttcn

<pre> /**  *  * @author  *   ETSI CTI OMA BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module provides service guide specific templates.  */ module LibBCast_ServiceGuide_Templates {     import from LibBCast_ServiceGuide_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_BasicTypesAndValues all;     import from LibBCast_ServicePrimitives_TypesAndValues all; </pre>	=	<pre> /**  *  * @author  *   ETSI CTI OMA BCAST CON Project  * @version  *   \$Id\$  * @desc  *   This module provides service guide specific templates.  */ module LibBCast_ServiceGuide_Templates {     import from LibBCast_ServiceGuide_TypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibCommon_BasicTypesAndValues all;     import from LibBCast_ServicePrimitives_TypesAndValues all; </pre>
<pre> import from AtsBCast_ModuleParameters all; import from LibBCast_ModuleParameters {     group DvbhBootstrapConfiguration }; </pre>	<>	
<pre>     group FDT {         /**          *          * @desc          *   This template creates a file elment which is used in » File          *   Delivery Table Instance (FDT).          * @param          *   p_ContentLocation The contnent location for the file » element          * @param          *   p_TOI The index of the file element in the FDT          * @see          * » LibBCast_Common_TypesAndValues.FileDeliveryTableInstance          * @see          *   LibBCast_Common_TypesAndValues.FileElement         */          template LibBCast_ServicePrimitives_TypesAndValues.FileType » m_def_FileType(             AnyUri p_Content_Location,             UInt p_TOI,             template charstring p_ContentType,             template charstring p_ContentEncoding) := {                 MBMS_Session_Identitys := omit,                 Content_Location := p_Content_Location,                 TOI := p_TOI, </pre>	=	<pre>     group FDT {         /**          *          * @desc          *   This template creates a file elment which is used in » File          *   Delivery Table Instance (FDT).          * @param          *   p_ContentLocation The contnent location for the file » element          * @param          *   p_TOI The index of the file element in the FDT          * @see          * » LibBCast_Common_TypesAndValues.FileDeliveryTableInstance          * @see          *   LibBCast_Common_TypesAndValues.FileElement         */          template LibBCast_ServicePrimitives_TypesAndValues.FileType » m_def_FileType(             AnyUri p_Content_Location,             UInt p_TOI,             template charstring p_ContentType,             template charstring p_ContentEncoding) := {                 MBMS_Session_Identitys := omit,                 Content_Location := p_Content_Location,                 TOI := p_TOI, </pre>

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

<pre> Content_Length := omit, Transfer_Length := omit, Content_Type := p_ContentType, Content_Encoding := p_ContentEncoding, Content_MD5 := omit, FEC_OTI_FEC_Encoding_ID := omit, FEC_OTI_FEC_Instance_ID := omit, FEC_OTI_Maximum_Source_Block_Length := omit, FEC_OTI_Encoding_Symbol_Length := omit, FEC_OTI_Max_Number_of_Encoding_Symbols := omit, FEC_OTI_Scheme_Specific_Info := omit, Version_ID_Length := omit  }  /**  *  * @desc This template creates a File Delivery Table Instance » (FDT)  * @param p_FileList The file list wich should be included  * @param p_Expires The expires time for the FDT  */  template FDT_InstanceType m_def_FDT_Instance(   LibBCast_ServicePrimitives_TypesAndValues.FileList » p_FileList,   charstring p_Expires,   template charstring p_Content_Type,   template UInt8 p_FEC_OTI_FEC_Encoding_ID) := {   Files := p_FileList,   MBMS_Session_Identity_Expirys := omit,   Expires := p_Expires,   Complete := omit,   Content_Type := p_Content_Type, » // necessary for xsd: FDT-InstanceType-BdsDvb   Content_Encoding := omit,   FEC_OTI_FEC_Encoding_ID := p_FEC_OTI_FEC_Encoding_ID, » // necessary for xsd: FDT-InstanceType-BdsDvb   FEC_OTI_FEC_Instance_ID := omit,   FEC_OTI_Maximum_Source_Block_Length := omit,   FEC_OTI_Encoding_Symbol_Length := omit,   FEC_OTI_Max_Number_of_Encoding_Symbols := omit,   FullFDT := omit,   Version_ID_Length := omit  }  } </pre>	<>=	<pre> Content_Length := omit, Transfer_Length := omit, Content_Type := p_ContentType, Content_Encoding := p_ContentEncoding, Content_MD5 := omit, FEC_OTI_FEC_Encoding_ID := omit, FEC_OTI_FEC_Instance_ID := omit, FEC_OTI_Maximum_Source_Block_Length := omit, FEC_OTI_Encoding_Symbol_Length := omit, FEC_OTI_Max_Number_of_Encoding_Symbols := omit, FEC_OTI_Scheme_Specific_Info := omit, Version_ID_Length := omit  }  /**  *  * @desc This template creates a File Delivery Table Instance » (FDT)  * @param p_FileList The file list wich should be included  * @param p_Expires The expires time for the FDT  */  template FDT_InstanceType m_def_FDT_Instance(   LibBCast_ServicePrimitives_TypesAndValues.FileList » p_FileList,   charstring p_Expires,   template charstring p_Content_Type,   template UInt8 p_FEC_OTI_FEC_Encoding_ID) := {   Files := p_FileList,   MBMS_Session_Identity_Expirys := omit,   Expires := p_Expires,   Complete := omit,   Content_Type := p_Content_Type, » // necessary for xsd: FDT-InstanceType-BdsDvb   Content_Encoding := omit,   FEC_OTI_FEC_Encoding_ID := p_FEC_OTI_FEC_Encoding_ID, » // necessary for xsd: FDT-InstanceType-BdsDvb   FEC_OTI_FEC_Instance_ID := omit,   FEC_OTI_Maximum_Source_Block_Length := omit,   FEC_OTI_Encoding_Symbol_Length := omit,   FEC_OTI_Max_Number_of_Encoding_Symbols := omit,   FullFDT := omit,   Version_ID_Length := omit  }  } </pre>
--	-----	--

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

<pre> group SGDD {     /**      *      * @desc      *   This templates creates a Service Guide Delivery » Descriptor      *   (SGDD) Fragment.      * @param      *   p_TOI The fragment index in the FDT      * @param      *   p_Id The fragment id      * @param      *   p_Encoding The encoding of the fragment      * @param      *   p_Type The fragment type      * @see      * @see      *   LibBCast_ServiceGuide_TypesAndValues.Fragment     */     template FragmentType m_def_Fragment(         UInt p_transportObjectId, </pre>		<pre> group SGDD {     /**      *      * @desc      *   This templates creates a Service Guide Delivery » Descriptor      *   (SGDD) Fragment.      * @param      *   p_TOI The fragment index in the FDT      * @param      *   p_Id The fragment id      * @param      *   p_Encoding The encoding of the fragment      * @param      *   p_Type The fragment type      * @see      * @see      *   LibBCast_ServiceGuide_TypesAndValues.Fragment     */     template FragmentType m_def_Fragment(         UInt p_transportObjectId, </pre>
<pre>         UInt p_fragmentVersion, // change 06 » (WK 6): dynamic setting of fragment version number </pre>	+ -	
<pre>         AnyUri p_Id,         UInt8 p_Encoding,         template UInt8 p_Type) := {             transportID := p_transportObjectId,             id := p_Id, </pre>	=	<pre>         AnyUri p_Id,         UInt8 p_Encoding,         template UInt8 p_Type) := {             transportID := p_transportObjectId,             id := p_Id, </pre>
<pre>         version := p_fragmentVersion, // change 06 » (WK 6): dynamic setting of fragment version number </pre>	< >	<pre>         version := 1, </pre>
<pre>         validFrom := omit,         validTo := omit,         fragmentEncoding := p_Encoding,         fragmentType := p_Type,         GroupingCriteria := omit     }      /**      *      * @desc      *   This template creates a ServiceGuideDeliveryUnitField » which      *   is used by the Service Guide Delivery Descriptor » (SGDD).      * @param      *   p_TOI The Service Guide Delivery Unit (SGDU) index in » the FDT </pre>	=	<pre>         validFrom := omit,         validTo := omit,         fragmentEncoding := p_Encoding,         fragmentType := p_Type,         GroupingCriteria := omit     }      /**      *      * @desc      *   This template creates a ServiceGuideDeliveryUnitField » which      *   is used by the Service Guide Delivery Descriptor » (SGDD).      * @param      *   p_TOI The Service Guide Delivery Unit (SGDU) index in » the FDT </pre>



Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

<pre> * @param *   p_location the location of the Service Guide Delivery » Unit *   (SGDU) * @param *   p_FragmentList The fragment list of the *   ServiceGuideDeliveryUnitField (SGDU) */ template ServiceGuideDeliveryUnitType » m_def_ServiceGuideDeliveryUnitField(     UInt p_TOI,     AnyUri p_location,     FragmentList p_FragmentList) := {     transportObjectID := p_TOI,     contentLocation := p_location,     validFrom := omit,     validTo := omit,     Fragments := p_FragmentList }  /** * * @desc *   This template creates a ServiceGuideDeliveryDescriptor *   (SGDD). * @param *   p_Id The id of the SGDD * @param *   p_ServiceGuideDeliveryUnitField The SGDU field of the » SGDD */ template ServiceGuideDeliveryDescriptorType » m_def_ServiceGuideDeliveryDescriptor(     AnyUri p_Id,     ServiceGuideDeliveryUnitType » p_ServiceGuideDeliveryUnitField,     UInt p_startTime, » change 01 (WK 6): global time definition     UInt p_endTime) := { » change 01 (WK 6): global time definition     id := p_Id,     version := 1,     NotificationReception := omit,     BSMLList := omit,     DescriptorEntrys := {     {     GroupingCriteria := { </pre>	<>	<pre> * @param *   p_location the location of the Service Guide Delivery » Unit *   (SGDU) * @param *   p_FragmentList The fragment list of the *   ServiceGuideDeliveryUnitField (SGDU) */ template ServiceGuideDeliveryUnitType » m_def_ServiceGuideDeliveryUnitField(     UInt p_TOI,     AnyUri p_location,     FragmentList p_FragmentList) := {     transportObjectID := p_TOI,     contentLocation := p_location,     validFrom := omit,     validTo := omit,     Fragments := p_FragmentList }  /** * * @desc *   This template creates a ServiceGuideDeliveryDescriptor *   (SGDD). * @param *   p_Id The id of the SGDD * @param *   p_ServiceGuideDeliveryUnitField The SGDU field of the » SGDD */ template ServiceGuideDeliveryDescriptorType » m_def_ServiceGuideDeliveryDescriptor(     AnyUri p_Id,     ServiceGuideDeliveryUnitType » p_ServiceGuideDeliveryUnitField) := {     id := p_Id,     version := 1,     NotificationReception := omit,     BSMLList := omit,     DescriptorEntrys := {     { </pre>
<pre>     {     GroupingCriteria := { </pre>	=	<pre>     {     { </pre>

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

» // change 20 (WK 6): for broadcasting, SGDD must contain grouping criteria
» and tranport element

                                TimeGroupingCriteria :=

» {p_startTime,p_endTime},

                                GenreGroupingCriteria := omit,
                                BSMSectors := omit,
                                ServiceCriteria := omit },
                                Transport := {

» PX_SGDU_FLUTE_SESSION_IP,PX_SGDU_FLUTE_SESSION_PORT,          // change 20
» (WK 6): for broadcasting, SGDD must contain grouping criteria and tranport
» element

```

```

                                TimeGroupingCriteria :=

                                GenreGroupingCriteria := omit,
                                BSMSectors := omit,
                                ServiceCriteria := omit },
                                Transport := {

```

```

                                PX_SRC IP, 1, omit, omit },

                                AlternativeAccessURLs := omit,
                                ServiceGuideDeliveryUnits := {
                                    p_ServiceGuideDeliveryUnitField }
                                }
                                },
                                PrivateExt := omit
                                }
                                }

```

```

group SGDU {

    group sdpFragments {

        /**
         *
         * @desc This is a default template for the
» SdpFragment.

         * @param p_fragmentId The id of the SDP fragment
         * @param p_content The content of the SDP fragment
         */
        template SdpFragment m_def_SdpFragment(
            charstring p_fragmentId,
            charstring p_content) := {
            validFrom := 0,
            validTo := 0,
            fragmentId := p_fragmentId,
            sdpFragment := p_content
        }

    }

    group xmlFragments {

        /**
         *

```

```

GroupingCriteria := omit,

```

```

Transport := omit,

```

```

=
                                AlternativeAccessURLs := omit,
                                ServiceGuideDeliveryUnits := {
                                    p_ServiceGuideDeliveryUnitField }
                                }
                                },
                                PrivateExt := omit
                                }
                                }

group SGDU {

    group sdpFragments {

        /**
         *
         * @desc This is a default template for the
» SdpFragment.

         * @param p_fragmentId The id of the SDP fragment
         * @param p_content The content of the SDP fragment
         */
        template SdpFragment m_def_SdpFragment(
            charstring p_fragmentId,
            charstring p_content) := {
            validFrom := 0,
            validTo := 0,
            fragmentId := p_fragmentId,
            sdpFragment := p_content
        }

    }

    group xmlFragments {

        /**
         *

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

    * @desc
    *   This template create a default PreviewData
» which includes a
    *   picture reference.
    * @param
    *   p_Id The Id of the PreviewData
    * @param
    *   p_Version The version of the PreviewData
    * @param
    *   p_PictureId The picture reference id
    */
template PreviewDataType m_def_picture_PreviewData(
    AnyUri p_Id,
    UInt p_Version,
    AnyUri p_PictureId,
    LanguageString p_txt) := {
    id := p_Id,
    version := p_Version,
    validFrom := omit,
    validTo := omit,
    SMIL := omit,
    Video := omit,
    Audio := omit,
    Picture := {
        choice := { PictureURI :=
            MIMEType := omit,
            AlternativeTexts := {p_txt} },
        Texts := omit,
        AccessReference := omit,
        PrivateExt := omit
    }
}

/**
 *
 * @desc This template create a Service Fragment
» which is used in a SGDU.
 * @param p_Id The service id
 * @param p_Name The service name
 * @param p_version The version of the service
» fragment
 * @param p_ServiceType The service type of the
» service
 */
template ServiceType m_def_Service(
    AnyUri p_Id,
    UInt p_version,

```

```

    * @desc
    *   This template create a default PreviewData
» which includes a
    *   picture reference.
    * @param
    *   p_Id The Id of the PreviewData
    * @param
    *   p_Version The version of the PreviewData
    * @param
    *   p_PictureId The picture reference id
    */
template PreviewDataType m_def_picture_PreviewData(
    AnyUri p_Id,
    UInt p_Version,
    AnyUri p_PictureId,
    LanguageString p_txt) := {
    id := p_Id,
    version := p_Version,
    validFrom := omit,
    validTo := omit,
    SMIL := omit,
    Video := omit,
    Audio := omit,
    Picture := {
        choice := { PictureURI :=
            MIMEType := omit,
            AlternativeTexts := {p_txt} },
        Texts := omit,
        AccessReference := omit,
        PrivateExt := omit
    }
}

/**
 *
 * @desc This template create a Service Fragment
» which is used in a SGDU.
 * @param p_Id The service id
 * @param p_Name The service name
 * @param p_version The version of the service
» fragment
 * @param p_ServiceType The service type of the
» service
 */
template ServiceType m_def_Service(
    AnyUri p_Id,
    UInt p_version,

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

charstring p_Name,
XmlElementUnsignedByteValue p_ServiceType) :=
» {

    id := p_Id,
    version := p_version,
    validFrom := omit,
    validTo := omit,
    globalServiceID := omit,
    weight := omit,
    //serviceContentProtection := omit,
    baseCID := omit,
    emergency := omit,
    ProtectionKeyIDs := omit,
    ServiceTypes := { {text_ := p_ServiceType }},
    Names := { {lang := omit, text_:=p_Name} },
    Descriptions := omit,
    AudioLanguages := omit,
    TextLanguages := omit,
    ParentalRatings := omit,
    TargetUserProfiles := omit,
    Genres := omit,
    Extensions := omit,
    PreviewDataReferences := omit,
    BroadcastArea := omit,
    TermsOfUses := omit,
    PrivateExt := omit
}

/**
 *
 * @desc
 *   This template create a Content Fragment which
» is used in a
 *
 *   SGDU.
 * @param
 *   p_Id the content id
 * @param
 *   p_Name the content name
 * @param
 *   p_ServiceId the referenced service id
 */
template ContentType m_def_Content(
    AnyUri p_Id,
    UInt p_version,
    charstring p_Name,
    AnyUri p_ServiceId,
    DateTime p_startTime,

```

```

charstring p_Name,
XmlElementUnsignedByteValue p_ServiceType) :=
» {

    id := p_Id,
    version := p_version,
    validFrom := omit,
    validTo := omit,
    globalServiceID := omit,
    weight := omit,
    //serviceContentProtection := omit,
    baseCID := omit,
    emergency := omit,
    ProtectionKeyIDs := omit,
    ServiceTypes := { {text_ := p_ServiceType }},
    Names := { {lang := omit, text_:=p_Name} },
    Descriptions := omit,
    AudioLanguages := omit,
    TextLanguages := omit,
    ParentalRatings := omit,
    TargetUserProfiles := omit,
    Genres := omit,
    Extensions := omit,
    PreviewDataReferences := omit,
    BroadcastArea := omit,
    TermsOfUses := omit,
    PrivateExt := omit
}

/**
 *
 * @desc
 *   This template create a Content Fragment which
» is used in a
 *
 *   SGDU.
 * @param
 *   p_Id the content id
 * @param
 *   p_Name the content name
 * @param
 *   p_ServiceId the referenced service id
 */
template ContentType m_def_Content(
    AnyUri p_Id,
    UInt p_version,
    charstring p_Name,
    AnyUri p_ServiceId,
    DateTime p_startTime,

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

        DateTime p_stopTime) := {
        id := p_Id,
        version := p_version,
        validFrom := omit,
        validTo := omit,
        globalContentID := omit,
        emergency := omit,
        //serviceContentProtection := omit,
        baseCID := omit,
        ServiceReferences := { { idRef :=
» p_ServiceId, weight := omit } },
        ProtectionKeyIDs := omit,
        Names := { {lang := omit, text_:=p_Name} },
        Descriptions := omit,
        StartTime := { text_ := p_startTime },
        EndTime := { text_ := p_stopTime },
        AudioLanguages := omit,
        TextLanguages := omit,
        ParentalRatings := omit,
        TargetUserProfiles := omit,
        Genres := omit,
        Extensions := omit,
        PreviewDataReferences := omit,
        BroadcastArea := omit,
        TermsOfUses := omit,
        PrivateExt := omit
    }

/**
 *
 * @desc This template creates a SGDU Schedule

» Fragment

 * @param p_Id the schedule id
 * @param p_ServiceId the referenced service id
 * @param p_ContentId the referenced content id
 * @param p_StartTime the start time
 * @param p_EndTime the end time
 */
template ScheduleType m_def_Schedule(
    AnyUri p_Id,
    UInt p_version,
    AnyUri p_ServiceId,
    AnyUri p_ContentId,
    UInt p_StartTime,
    UInt p_EndTime) := {
    id := p_Id,
    version := p_version,

```

```

        DateTime p_stopTime) := {
        id := p_Id,
        version := p_version,
        validFrom := omit,
        validTo := omit,
        globalContentID := omit,
        emergency := omit,
        //serviceContentProtection := omit,
        baseCID := omit,
        ServiceReferences := { { idRef :=
» p_ServiceId, weight := omit } },
        ProtectionKeyIDs := omit,
        Names := { {lang := omit, text_:=p_Name} },
        Descriptions := omit,
        StartTime := { text_ := p_startTime },
        EndTime := { text_ := p_stopTime },
        AudioLanguages := omit,
        TextLanguages := omit,
        ParentalRatings := omit,
        TargetUserProfiles := omit,
        Genres := omit,
        Extensions := omit,
        PreviewDataReferences := omit,
        BroadcastArea := omit,
        TermsOfUses := omit,
        PrivateExt := omit
    }

/**
 *
 * @desc This template creates a SGDU Schedule

» Fragment

 * @param p_Id the schedule id
 * @param p_ServiceId the referenced service id
 * @param p_ContentId the referenced content id
 * @param p_StartTime the start time
 * @param p_EndTime the end time
 */
template ScheduleType m_def_Schedule(
    AnyUri p_Id,
    UInt p_version,
    AnyUri p_ServiceId,
    AnyUri p_ContentId,
    UInt p_StartTime,
    UInt p_EndTime) := {
    id := p_Id,
    version := p_version,

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

        defaultSchedule := omit,
        onDemand := omit,
        validFrom := omit,
        validTo := omit,
        ServiceReference := {
            idRef := p_ServiceId,
            audioLanguageIDRef := omit,
            textLanguageIDRef := omit },
        InteractivityDataReferences := omit,
        ContentReferences := {
            {
                idRef := p_ContentId,
                //contentLocation := p_ContentId,
» // Change No. 11 from OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review
                contentLocation := omit,
                audioLanguageIDRef := omit,
                textLanguageIDRef := omit,
                repeatPlayback := omit,
                AutoStarts := omit,
                DistributionWindows := omit,
                PresentationWindows := {
                    {
                        startTime := p_StartTime,
                        endTime := p_EndTime,
                        duration := omit,
                        id := omit }
                    }
                },
                PreviewDataReferences := omit,
                TermsOfUses := omit,
                PrivateExt := omit
            }
        },
        /**
         *
         * @desc This is a default template for a Schedule
» fragment which refers a interactivity fragment.
         * @param p_Id The id of the schedule fragment
         * @param p_version The fragment version
         * @param p_ServiceId The refered service id
         * @param p_InteractivityDataId the referd
» interactivity data
         * @param p_StartTime the start time
         * @param p_EndTime the end time
         */
template ScheduleType m_def_inter_Schedule(

```

```

        defaultSchedule := omit,
        onDemand := omit,
        validFrom := omit,
        validTo := omit,
        ServiceReference := {
            idRef := p_ServiceId,
            audioLanguageIDRef := omit,
            textLanguageIDRef := omit },
        InteractivityDataReferences := omit,
        ContentReferences := {
            {
                idRef := p_ContentId,
                //contentLocation := p_ContentId,
» // Change No. 11 from OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review
                contentLocation := omit,
                audioLanguageIDRef := omit,
                textLanguageIDRef := omit,
                repeatPlayback := omit,
                AutoStarts := omit,
                DistributionWindows := omit,
                PresentationWindows := {
                    {
                        startTime := p_StartTime,
                        endTime := p_EndTime,
                        duration := omit,
                        id := omit }
                    }
                },
                PreviewDataReferences := omit,
                TermsOfUses := omit,
                PrivateExt := omit
            }
        },
        /**
         *
         * @desc This is a default template for a Schedule
» fragment which refers a interactivity fragment.
         * @param p_Id The id of the schedule fragment
         * @param p_version The fragment version
         * @param p_ServiceId The refered service id
         * @param p_InteractivityDataId the referd
» interactivity data
         * @param p_StartTime the start time
         * @param p_EndTime the end time
         */
template ScheduleType m_def_inter_Schedule(

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

        AnyUri p_Id,
        UInt p_version,
        AnyUri p_ServiceId,
        AnyUri p_InteractivityDataId) := {
    id := p_Id,
    version := p_version,
    defaultSchedule := omit,
    onDemand := omit,
    validFrom := omit,
    validTo := omit,
    ServiceReference := {
        idRef := p_ServiceId,
        audioLanguageIDRef := omit,
        textLanguageIDRef := omit },
    InteractivityDataReferences := {
    {
        idRef := p_InteractivityDataId,
        AutoStarts := omit,
        DistributionWindows := omit }
    },
    ContentReferences := omit,
    PreviewDataReferences := omit,
    TermsOfUses := omit,
    PrivateExt := omit
}

/**
 *
 * @desc This template create a AccessType for
» broadcast (SDP) which is used in the access fragment of the SGDU.
 * @param p_bcType the broadcast type
 * @param p_SDP the SDP which describes the session
 */
template AccessTypeType m_def_AccessType_bc_sdp(
    UInt8 p_bcType,
    charstring p_sdpId) := {
    choice := {
        BroadcastServiceDelivery := {
            BDSType := {
                Type := {p_bcType},
                Versions := omit
            },
            SessionDescription := {
                // choice := {SDPRef :=
» {p_sdpId,p_sdpId}}, // Change No. 13 from
» OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review
                choice := {SDPRef := {uri :=

```

```

        AnyUri p_Id,
        UInt p_version,
        AnyUri p_ServiceId,
        AnyUri p_InteractivityDataId) := {
    id := p_Id,
    version := p_version,
    defaultSchedule := omit,
    onDemand := omit,
    validFrom := omit,
    validTo := omit,
    ServiceReference := {
        idRef := p_ServiceId,
        audioLanguageIDRef := omit,
        textLanguageIDRef := omit },
    InteractivityDataReferences := {
    {
        idRef := p_InteractivityDataId,
        AutoStarts := omit,
        DistributionWindows := omit }
    },
    ContentReferences := omit,
    PreviewDataReferences := omit,
    TermsOfUses := omit,
    PrivateExt := omit
}

/**
 *
 * @desc This template create a AccessType for
» broadcast (SDP) which is used in the access fragment of the SGDU.
 * @param p_bcType the broadcast type
 * @param p_SDP the SDP which describes the session
 */
template AccessTypeType m_def_AccessType_bc_sdp(
    UInt8 p_bcType,
    charstring p_sdpId) := {
    choice := {
        BroadcastServiceDelivery := {
            BDSType := {
                Type := {p_bcType},
                Versions := omit
            },
            SessionDescription := {
                // choice := {SDPRef :=
» {p_sdpId,p_sdpId}}, // Change No. 13 from
» OMA-IOP-TTCN-2007-0026-CR_TTCN_code_review
                choice := {SDPRef := {uri :=

```

## Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

» omit, idRef := p_sdpId}}, // editorial modification, when merging 0001R01
» and 0039R01 to permanent document

        USBDRef := omit,
        ADPRef := omit },
        FileDescription := omit }

    }

/**
 *
 * @desc This template create a SGDU Access Fragment.
 * @param p_Id the access id
 * @param p_AccessType the access type
 * @param p_ServiceId the referenced service id
 * @param p_ScheduleId the referenced schedule id
 * @param p_ServiceClass the service class
 */
template AccessType m_def_Access(
    AnyUri p_Id,
    UInt p_version,
    AccessTypeType p_AccessType,
    AnyUri p_ServiceId,
    template AnyUri p_ScheduleId,
    charstring p_ServiceClass) := {
    id := p_Id,
    version := p_version,
    validFrom := omit,
    validTo := omit,
    AccessType := p_AccessType,
    KeyManagementSystems := omit,
    EncryptionTypes := omit,
    //choice := {ServiceReferences := { { idRef
    := p_ServiceId } } }, // change 21 (WK 6):
    » ETS requires, that access fragment references schedule fragment
    choice := {ScheduleReferences := { { idRef :=
    » p_ScheduleId, DistributionWindowIDs := omit } } }, // change 21 (WK 6):
    » ETS requires, that access fragment references schedule fragment

    TerminalCapabilityRequirement := omit,
    BandwidthRequirement := omit,
    ServiceClasss := { {p_ServiceClass} },
    PreviewDataReferences := omit,
    NotificationReception := omit,
    PrivateExt := omit

    }

/**

```

```

» omit, idRef := p_sdpId}}, // editorial modification, when merging 0001R01
» and 0039R01 to permanent document

        USBDRef := omit,
        ADPRef := omit },
        FileDescription := omit }

    }

/**
 *
 * @desc This template create a SGDU Access Fragment.
 * @param p_Id the access id
 * @param p_AccessType the access type
 * @param p_ServiceId the referenced service id
 * @param p_ScheduleId the referenced schedule id
 * @param p_ServiceClass the service class
 */
template AccessType m_def_Access(
    AnyUri p_Id,
    UInt p_version,
    AccessTypeType p_AccessType,
    AnyUri p_ServiceId,
    template AnyUri p_ScheduleId,
    charstring p_ServiceClass) := {
    id := p_Id,
    version := p_version,
    validFrom := omit,
    validTo := omit,
    AccessType := p_AccessType,
    KeyManagementSystems := omit,
    EncryptionTypes := omit,
    choice := {ServiceReferences := { { idRef :=
    » p_ServiceId } } },

    TerminalCapabilityRequirement := omit,
    BandwidthRequirement := omit,
    ServiceClasss := { {p_ServiceClass} },
    PreviewDataReferences := omit,
    NotificationReception := omit,
    PrivateExt := omit

    }

/**

```



Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

*
* @desc This template is a default template for
» PurchaseItem fragment.
* @param p_Id The id of the PurchaseItem
* @param p_Version The Version of the PurchaseItem
* @param p_GlobalPurchaseId The global purchase id
* @param p_ServiceID The serive id of the referenced
» service fragment

* @param p_Name The name of the PurchaseItem
*/
template PurchaseItemType m_def_PurchaseItem(
    AnyUri p_Id,
    UInt p_Version,
    AnyUri p_GlobalPurchaseId,
    AnyUri p_ServiceID,
    charstring p_Name) := {
    id := p_Id,
    version := p_Version,
    validFrom := omit,
    validTo := omit,
    globalPurchaseItemID := p_GlobalPurchaseId,
    binaryPurchaseItemID := omit,
    weight := omit,
    closed := omit,
    choice := {ServiceReferences := { { idRef :=
» p_ServiceID } } },

    ProtectionKeyIDs := omit,
    Names := { {lang := omit, text_:=p_Name} },
    Descriptions := omit,
    StartTime := omit,
    EndTime := omit,
    ParentalRatings := omit,
    Extensions := omit,
    DependencyReferences := omit,
    ExclusionReferences := omit,
    PrivateExt := omit
}

/**
*
* @desc This template is a default template for
» PurchaseData.
* @param p_Id The id of the PurchaseData
* @param p_Version The version of the PurchaseData
* @param p_Description the description for the
» PurchaseData fragment
* @param p_PurchaseItemId The id of the referenced

```

```

*
* @desc This template is a default template for
» PurchaseItem fragment.
* @param p_Id The id of the PurchaseItem
* @param p_Version The Version of the PurchaseItem
* @param p_GlobalPurchaseId The global purchase id
* @param p_ServiceID The serive id of the referenced
» service fragment

* @param p_Name The name of the PurchaseItem
*/
template PurchaseItemType m_def_PurchaseItem(
    AnyUri p_Id,
    UInt p_Version,
    AnyUri p_GlobalPurchaseId,
    AnyUri p_ServiceID,
    charstring p_Name) := {
    id := p_Id,
    version := p_Version,
    validFrom := omit,
    validTo := omit,
    globalPurchaseItemID := p_GlobalPurchaseId,
    binaryPurchaseItemID := omit,
    weight := omit,
    closed := omit,
    choice := {ServiceReferences := { { idRef :=
» p_ServiceID } } },

    ProtectionKeyIDs := omit,
    Names := { {lang := omit, text_:=p_Name} },
    Descriptions := omit,
    StartTime := omit,
    EndTime := omit,
    ParentalRatings := omit,
    Extensions := omit,
    DependencyReferences := omit,
    ExclusionReferences := omit,
    PrivateExt := omit
}

/**
*
* @desc This template is a default template for
» PurchaseData.
* @param p_Id The id of the PurchaseData
* @param p_Version The version of the PurchaseData
* @param p_Description the description for the
» PurchaseData fragment
* @param p_PurchaseItemId The id of the referenced

```

## Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

» of the PurchaseItem
    * @param p_PurchaseChannelId The id of the
» referenced of the PurchaseChannel
    * @param p_PriceInfo The PriceInfo field. It could
» be omit

    */
    template PurchaseDataType m_def_PurchaseData(
        AnyUri p_Id,
        UInt p_Version,
        template LanguageStringList p_Descriptions,
        AnyUri p_PurchaseItemId,
        AnyUri p_PurchaseChannelId,
        template PriceInfoType p_PriceInfo
    ) := {
        id := p_Id,
        version := p_Version,
        validFrom := omit,
        validTo := omit,
        Descriptions := p_Descriptions,
        PriceInfo := p_PriceInfo,
        PromotionInfos := omit,
        Extensions := omit,
        PurchaseItemReference := { idRef :=
» p_PurchaseItemId },
        PurchaseChannelReferences := { { idRef :=
» p_PurchaseChannelId } },
        PreviewDataReferences := omit,
        TermsOfUses := omit,
        PrivateExt := omit
    }
    /**
    *
    * @desc Specifies the price information of the
» purchase item associated with this PurchaseData fragment.
    * @param p_subscriptionType
    * @param p_monetaryPrice
    * @param p_currency
    */
    template PriceInfoType m_PriceInfo(
        UInt p_subscriptionType,
        float p_monetaryPrice,
        charstring p_currency) := {
        subscriptionType := p_subscriptionType,
        MonetaryPrices := {
            { text_ := p_monetaryPrice, currency
» := p_currency }
        },

```

```

» of the PurchaseItem
    * @param p_PurchaseChannelId The id of the
» referenced of the PurchaseChannel
    * @param p_PriceInfo The PriceInfo field. It could
» be omit

    */
    template PurchaseDataType m_def_PurchaseData(
        AnyUri p_Id,
        UInt p_Version,
        template LanguageStringList p_Descriptions,
        AnyUri p_PurchaseItemId,
        AnyUri p_PurchaseChannelId,
        template PriceInfoType p_PriceInfo
    ) := {
        id := p_Id,
        version := p_Version,
        validFrom := omit,
        validTo := omit,
        Descriptions := p_Descriptions,
        PriceInfo := p_PriceInfo,
        PromotionInfos := omit,
        Extensions := omit,
        PurchaseItemReference := { idRef :=
» p_PurchaseItemId },
        PurchaseChannelReferences := { { idRef :=
» p_PurchaseChannelId } },
        PreviewDataReferences := omit,
        TermsOfUses := omit,
        PrivateExt := omit
    }
    /**
    *
    * @desc Specifies the price information of the
» purchase item associated with this PurchaseData fragment.
    * @param p_subscriptionType
    * @param p_monetaryPrice
    * @param p_currency
    */
    template PriceInfoType m_PriceInfo(
        UInt p_subscriptionType,
        float p_monetaryPrice,
        charstring p_currency) := {
        subscriptionType := p_subscriptionType,
        MonetaryPrices := {
            { text_ := p_monetaryPrice, currency
» := p_currency }
        },

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

        TotalNumberToken := omit,
        SubscriptionPeriod := omit
    }

    /**
     *
     * @desc This is a default template for the
» PurchaseChannel fragment.
     * @param p_Id The id of the PurchaseChannel
     * @param p_Version The version of the
» PurchaseChannel

     * @param p_Name The name for PurchaseChannel
     * @param p_PurchaseUrl and the url of the
» subscription site

    */
    template PurchaseChannelType m_def_PurchaseChannel(
        AnyUri p_Id,
        UInt p_Version,
        charstring p_Name,
        AnyUri p_PurchaseUrl) := {
        id := p_Id,
        version := p_Version,
        validFrom := omit,
        validTo := omit,
        rightsIssuerURI := omit,
        Names := { {lang := omit, text_:=p_Name} },
        Descriptions := omit,
        ContactInfo := omit,
        PortalURL := omit,
        PurchaseURLs := { {p_PurchaseUrl} },
        Extensions := omit,
        PrivateExt := omit
    }

    /**
     *
     * @desc This this a default InteractivityData
» template.
     * @param p_Id The id of the InteractivityData
     * @param p_Version The version of the
» InteractivityData

     * @param p_PreListenIndicator a boolean to indecate
» how the terminal should handle interactivity media objects
     * @param p_InteractivityMediaDocumentGroupId The id
» of a media document group
     * @param p_ServiceId the referenced service fragment

```

```

        TotalNumberToken := omit,
        SubscriptionPeriod := omit
    }

    /**
     *
     * @desc This is a default template for the
» PurchaseChannel fragment.
     * @param p_Id The id of the PurchaseChannel
     * @param p_Version The version of the
» PurchaseChannel

     * @param p_Name The name for PurchaseChannel
     * @param p_PurchaseUrl and the url of the
» subscription site

    */
    template PurchaseChannelType m_def_PurchaseChannel(
        AnyUri p_Id,
        UInt p_Version,
        charstring p_Name,
        AnyUri p_PurchaseUrl) := {
        id := p_Id,
        version := p_Version,
        validFrom := omit,
        validTo := omit,
        rightsIssuerURI := omit,
        Names := { {lang := omit, text_:=p_Name} },
        Descriptions := omit,
        ContactInfo := omit,
        PortalURL := omit,
        PurchaseURLs := { {p_PurchaseUrl} },
        Extensions := omit,
        PrivateExt := omit
    }

    /**
     *
     * @desc This this a default InteractivityData
» template.
     * @param p_Id The id of the InteractivityData
     * @param p_Version The version of the
» InteractivityData

     * @param p_PreListenIndicator a boolean to indecate
» how the terminal should handle interactivity media objects
     * @param p_InteractivityMediaDocumentGroupId The id
» of a media document group
     * @param p_ServiceId the referenced service fragment

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

» id
    */
    template InteractivityDataType
» m_def_InteractivityData(
    AnyUri p_Id,
    UInt p_Version,
    boolean p_PreListenIndicator,
    AnyUri p_InteractivityMediaDocumentGroupId,
    AnyUri p_ServiceId) := {
    id := p_Id,
    version := p_Version,
    validFrom := omit,
    validTo := omit,
    preListenIndicator := p_PreListenIndicator,
    interactivityMediaDocumentPointer :=
        p_InteractivityMediaDocumentGroupId,
    InteractivityTypes := omit,
    ServiceReference := { idRef := p_ServiceId },
    choice := omit,
    InteractivityDelivery := omit,
    Extension := omit,
    BackOffTiming := omit,
    TermsOfUses := omit,
    PrivateExt := omit
    }

    group ServiceGuideFragments {
        /**
         *
         * @desc This template is a default template for a
» xml ServiceGuideFragment.
         * @param p_fragmentTransportId The Transport ID of
» the fragment
         * @param p_fragmentVersion The fragment version
         * @param p_xmlFragment the the xml fragment which
» should be included
         */
        template ServiceGuideFragment
» m_xml_ServiceGuideFragment(
            UInt32 p_fragmentTransportId,
            UInt32 p_fragmentVersion,
            XmlFragment p_xmlFragment) := {
            fragmentTransportId := p_fragmentTransportId,
            fragmentVersion := p_fragmentVersion,
            fragmentType := { xmlFragment :=
» p_xmlFragment }

```

```

» id
    */
    template InteractivityDataType
» m_def_InteractivityData(
    AnyUri p_Id,
    UInt p_Version,
    boolean p_PreListenIndicator,
    AnyUri p_InteractivityMediaDocumentGroupId,
    AnyUri p_ServiceId) := {
    id := p_Id,
    version := p_Version,
    validFrom := omit,
    validTo := omit,
    preListenIndicator := p_PreListenIndicator,
    interactivityMediaDocumentPointer :=
        p_InteractivityMediaDocumentGroupId,
    InteractivityTypes := omit,
    ServiceReference := { idRef := p_ServiceId },
    choice := omit,
    InteractivityDelivery := omit,
    Extension := omit,
    BackOffTiming := omit,
    TermsOfUses := omit,
    PrivateExt := omit
    }

    group ServiceGuideFragments {
        /**
         *
         * @desc This template is a default template for a
» xml ServiceGuideFragment.
         * @param p_fragmentTransportId The Transport ID of
» the fragment
         * @param p_fragmentVersion The fragment version
         * @param p_xmlFragment the the xml fragment which
» should be included
         */
        template ServiceGuideFragment
» m_xml_ServiceGuideFragment(
            UInt32 p_fragmentTransportId,
            UInt32 p_fragmentVersion,
            XmlFragment p_xmlFragment) := {
            fragmentTransportId := p_fragmentTransportId,
            fragmentVersion := p_fragmentVersion,
            fragmentType := { xmlFragment :=
» p_xmlFragment }

```

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

    }

    /**
     *
     * @desc This template is a default template for a
    » sdp ServiceGuideFragment.
     * @param p_fragmentTransportId The Transport ID of
    » the fragment
     * @param p_fragmentVersion The fragment version
     * @param p_sdpFragment the the sdp fragment which
    » should be included
     */
    template ServiceGuideFragment
    » m_sdp_ServiceGuideFragment(
        UInt32 p_fragmentTransportId,
        UInt32 p_fragmentVersion,
        SdpFragment p_sdpFragment) := {
        fragmentTransportId := p_fragmentTransportId,
        fragmentVersion := p_fragmentVersion,
        fragmentType := { sdpFragment :=
    » p_sdpFragment }
    }

    /**
     *
     * @desc This template is a default template for a
    » mbms ServiceGuideFragment.
     * @param p_fragmentTransportId The Transport ID of
    » the fragment
     * @param p_fragmentVersion The fragment version
     * @param p_mbmsUsbdFragment the the mbms fragment
    » which should be included
     */
    template ServiceGuideFragment
    » m_mbmsUsbd_ServiceGuideFragment(
        UInt32 p_fragmentTransportId,
        UInt32 p_fragmentVersion,
        MbmsUsbdFragment p_mbmsUsbdFragment) := {
        fragmentTransportId := p_fragmentTransportId,
        fragmentVersion := p_fragmentVersion,
        fragmentType := { mbmsUsbdFragment :=
    » p_mbmsUsbdFragment }
    }

    /**
     *
     * @desc This template is a default template for a

```

```

    }

    /**
     *
     * @desc This template is a default template for a
    » sdp ServiceGuideFragment.
     * @param p_fragmentTransportId The Transport ID of
    » the fragment
     * @param p_fragmentVersion The fragment version
     * @param p_sdpFragment the the sdp fragment which
    » should be included
     */
    template ServiceGuideFragment
    » m_sdp_ServiceGuideFragment(
        UInt32 p_fragmentTransportId,
        UInt32 p_fragmentVersion,
        SdpFragment p_sdpFragment) := {
        fragmentTransportId := p_fragmentTransportId,
        fragmentVersion := p_fragmentVersion,
        fragmentType := { sdpFragment :=
    » p_sdpFragment }
    }

    /**
     *
     * @desc This template is a default template for a
    » mbms ServiceGuideFragment.
     * @param p_fragmentTransportId The Transport ID of
    » the fragment
     * @param p_fragmentVersion The fragment version
     * @param p_mbmsUsbdFragment the the mbms fragment
    » which should be included
     */
    template ServiceGuideFragment
    » m_mbmsUsbd_ServiceGuideFragment(
        UInt32 p_fragmentTransportId,
        UInt32 p_fragmentVersion,
        MbmsUsbdFragment p_mbmsUsbdFragment) := {
        fragmentTransportId := p_fragmentTransportId,
        fragmentVersion := p_fragmentVersion,
        fragmentType := { mbmsUsbdFragment :=
    » p_mbmsUsbdFragment }
    }

    /**
     *
     * @desc This template is a default template for a

```

## Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

» apd ServiceGuideFragment.
    * @param p_fragmentTransportId The Transport ID of
» the fragment
    * @param p_fragmentVersion The fragment version
    * @param p_adpFragment the the apd fragment which
» should be included
    */
    template ServiceGuideFragment
» m_adp_ServiceGuideFragment(
        UInt32 p_fragmentTransportId,
        UInt32 p_fragmentVersion,
        FileAdp p_adpFragment) := {
    fragmentTransportId := p_fragmentTransportId,
    fragmentVersion := p_fragmentVersion,
    fragmentType := { adpFragment :=
» p_adpFragment }
    }
}

```

```

» apd ServiceGuideFragment.
    * @param p_fragmentTransportId The Transport ID of
» the fragment
    * @param p_fragmentVersion The fragment version
    * @param p_adpFragment the the apd fragment which
» should be included
    */
    template ServiceGuideFragment
» m_adp_ServiceGuideFragment(
        UInt32 p_fragmentTransportId,
        UInt32 p_fragmentVersion,
        FileAdp p_adpFragment) := {
    fragmentTransportId := p_fragmentTransportId,
    fragmentVersion := p_fragmentVersion,
    fragmentType := { adpFragment :=
» p_adpFragment }
    }
}

```

```

// change 13 (WK 6): send DVB-H bootstrap files via FLUTE
group DVBH_Bootstrap {
    /**
    *
    * @desc
    * This is a default template for the
» ESGProviderDiscoveryType
    * @param
    * p_providerURI The provider URI
    * @param
    * p_providerName The provider name
    * @param
    * p_providerID The provider ID
    */
    template ESGProviderDiscoveryType
» m_def_esg_prov_disc(charstring p_providerURI, charstring p_providerName,
» charstring p_providerID) := {
        ServiceProviders := {
            {
                format :=
» "urn:oma:xml:bcast:sg:fragments:1.0",
                ProviderURI := {text_ :=
» p_providerURI},
                ProviderName := {text_ :=
» p_providerName},
                ProviderID := {text_ := p_providerID}
            }
        }
    }
}

```

&lt;&gt;

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

<pre>         }     }      /**      *      * @desc      *   This is a default template for the ESGAccessType      * @param      *   p_srcIpAddress The source IPv4 address      * @param      *   p_dstIpAddress The destination IPv4 address      * @param      *   p_dstIpPort The destination port      */     template ESGAccessType m_def_esg_access_ipv4(charstring » p_srcIpAddress, charstring p_dstIpAddress, UInt16 p_dstIpPort) := {         ESGEntryVersion := 1,         MultipleStreamTransport := false,         IPVersion6 := false,         ProviderID := 1,         SrcIpAddress := p_srcIpAddress,         DstIpAddress := p_dstIpAddress,         DstIpPort := p_dstIpPort,         TSI := 0     } } </pre>		
<pre> group IMD {     /**      *      * @desc      *   This is a default template for the » InteractivityMediaDocumentType      * @param </pre>	=	<pre> group IMD {     /**      *      * @desc      *   This is a default template for the » InteractivityMediaDocumentType      * @param </pre>
<pre>      *   p_GroupID The id of the group // » change 99 (WK 6): Clerical </pre>	<>	<pre>      *   p_GroupID The id of the grouo </pre>
<pre>      * @param      *   p_GroupPosition The position of the group      * @param </pre>	=	<pre>      * @param      *   p_GroupPosition The position of the group      * @param </pre>
<pre>      *   p_id The media document id // » change 99 (WK 6): Clerical </pre>	<>	<pre>      *   p_id the media document id </pre>
<pre>      * @param      *   p_MediaObjectGroupList the media object group list      */     template InteractivityMediaDocumentType » m_def_InteractivityMediaDocument( </pre>	=	<pre>      * @param      *   p_MediaObjectGroupList the media object group list      */     template InteractivityMediaDocumentType » m_def_InteractivityMediaDocument( </pre>

Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

        AnyUri p_GroupID,
        UInt16 p_GroupPosition,
        AnyUri p_id,
        UInt    p_version,
        MediaObjectGroupList p_MediaObjectGroups) := {
    groupID := p_GroupID,
    groupPosition := p_GroupPosition,
    id := p_id,
    version := p_version,
    vaildFrom := omit,
    valudTo := omit,
    MediaObjectGroups := p_MediaObjectGroups,
    PrivateExt := omit
}

/**
 *
 * @desc
 *   This is a default MediaObjectGroupType template
 * @param
 *   p_id The id of the group
 * @param
 *   p_StartMediaFlag the start media flag
 * @param
 *   p_MediaObjectSetList the media object set list
 */
template MediaObjectGroupType m_def_MediaObjectGroup(
    AnyUri p_id,
    boolean p_startMediaFlag,
    template MediaObjectSetList p_MediaObjectSets) := {
    id := p_id,
    startMediaFlag := p_startMediaFlag,
    ActionDescriptor := omit,
    BackOffTiming := omit,
    MediaObjectSets := p_MediaObjectSets,
    SMSTemplate := omit,
    EmailTemplate := omit,
    VoiceCall := omit,
    WebLink := omit,
    AlternativeText := omit
}

/**
 *
 * @desc
 *   This is a default MediaObjectSetType template
 * @param

```

```

        AnyUri p_GroupID,
        UInt16 p_GroupPosition,
        AnyUri p_id,
        UInt    p_version,
        MediaObjectGroupList p_MediaObjectGroups) := {
    groupID := p_GroupID,
    groupPosition := p_GroupPosition,
    id := p_id,
    version := p_version,
    vaildFrom := omit,
    valudTo := omit,
    MediaObjectGroups := p_MediaObjectGroups,
    PrivateExt := omit
}

/**
 *
 * @desc
 *   This is a default MediaObjectGroupType template
 * @param
 *   p_id The id of the group
 * @param
 *   p_StartMediaFlag the start media flag
 * @param
 *   p_MediaObjectSetList the media object set list
 */
template MediaObjectGroupType m_def_MediaObjectGroup(
    AnyUri p_id,
    boolean p_startMediaFlag,
    template MediaObjectSetList p_MediaObjectSets) := {
    id := p_id,
    startMediaFlag := p_startMediaFlag,
    ActionDescriptor := omit,
    BackOffTiming := omit,
    MediaObjectSets := p_MediaObjectSets,
    SMSTemplate := omit,
    EmailTemplate := omit,
    VoiceCall := omit,
    WebLink := omit,
    AlternativeText := omit
}

/**
 *
 * @desc
 *   This is a default MediaObjectSetType template
 * @param

```



Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

```

        *      p_content_Type The content type of the
» MediaObjectSetElement
        * @param
        *      p_content_Location The location of the content
        * @param
        *      p_objects The list of media objects or omit
        */
template MediaObjectSetType m_def_mediaObjectSet(
    charstring p_Content_Type,
    AnyUri p_Content_Location,
    template ObjectList p_Objects) := {
    relativePreference := omit,
    Content_Type := p_Content_Type,
    Content_Location := p_Content_Location,
    Descriptions := omit,
    Objects := p_Objects,
    File := omit
}

/**
 *
 * @desc
 *      This is a default ObjectType template
 * @param
 *      p_Content_Location the location of the ObjectType
 * @param
 *      p_Content_Type the type of the ObjectType
 * @param
 *      p_start the start flag of the ObjectType
 * @param
 *      p_PartTypes the part type list
 */
template ObjectType m_def_Object(
    template AnyUri p_Content_Location,
    charstring p_Content_Type,
    template boolean p_start_,
    template PartTypeList p_PartTypes) := {
    Content_Location := p_Content_Location,
    Content_Type := p_Content_Type,
    start_ := p_start_,
    PartTypes := p_PartTypes
}

/**
 *
 * @desc
 *      This is a default template for a SMSTemplateType

```

```

        *      p_content_Type The content type of the
» MediaObjectSetElement
        * @param
        *      p_content_Location The location of the content
        * @param
        *      p_objects The list of media objects or omit
        */
template MediaObjectSetType m_def_mediaObjectSet(
    charstring p_Content_Type,
    AnyUri p_Content_Location,
    template ObjectList p_Objects) := {
    relativePreference := omit,
    Content_Type := p_Content_Type,
    Content_Location := p_Content_Location,
    Descriptions := omit,
    Objects := p_Objects,
    File := omit
}

/**
 *
 * @desc
 *      This is a default ObjectType template
 * @param
 *      p_Content_Location the location of the ObjectType
 * @param
 *      p_Content_Type the type of the ObjectType
 * @param
 *      p_start the start flag of the ObjectType
 * @param
 *      p_PartTypes the part type list
 */
template ObjectType m_def_Object(
    template AnyUri p_Content_Location,
    charstring p_Content_Type,
    template boolean p_start_,
    template PartTypeList p_PartTypes) := {
    Content_Location := p_Content_Location,
    Content_Type := p_Content_Type,
    start_ := p_start_,
    PartTypes := p_PartTypes
}

/**
 *
 * @desc
 *      This is a default template for a SMSTemplateType

```

## Datei: LibBCast\_ServiceGuide\_Templates.ttcn (Fortsetzung)

<pre> * @param *   p_SelectChoices the list of SelectChoice */ template SMSTemplateType m_def_SmsTemplate(     SelectChoiceList p_SelectChoices) := {     relativePreference := omit,     Descriptions := omit,     SelectChoices := p_SelectChoices }  } </pre>	<pre> * @param *   p_SelectChoices the list of SelectChoice */ template SMSTemplateType m_def_SmsTemplate(     SelectChoiceList p_SelectChoices) := {     relativePreference := omit,     Descriptions := omit,     SelectChoices := p_SelectChoices }  } </pre>
--	--

## Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn

<pre> /**  *  * @author  *   ETSI OMA BCAST CON Project  * @version  *   0.0.1  * @desc  *   This file specifies SGDD, SGDU, SG Reponse, and SG fragments as well  *   as related costants as defined in OMA BCAST SG document  * @remark  *   End users should be aware that any changes made to this file may  *   negatively affect the interworking of test suite code with the  */ module LibBCast_ServiceGuide_TypesAndValues {     import from LibCommon_BasicTypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibBCast_FileStreamDistribution_TypesAndValues all;      group pduTypes {         group serviceGuideDeliveryDescriptor {             /** @desc SGDD structure as defined in SG 5.4.1.5.2             type record ServiceGuideDeliveryDescriptorType {                 AnyUri id optional,                 UInt version optional,                 NotificationReceptionType » NotificationReception optional,                 BSMLListType BSMLList optional,                 DescriptorEntryList DescriptorEntrys,                 PrivateExtType PrivateExt optional             }              /**              *              * @desc </pre>	=	<pre> /**  *  * @author  *   ETSI OMA BCAST CON Project  * @version  *   0.0.1  * @desc  *   This file specifies SGDD, SGDU, SG Reponse, and SG fragments as well  *   as related costants as defined in OMA BCAST SG document  * @remark  *   End users should be aware that any changes made to this file may  *   negatively affect the interworking of test suite code with the  */ module LibBCast_ServiceGuide_TypesAndValues {     import from LibCommon_BasicTypesAndValues all;     import from LibBCast_Common_TypesAndValues all;     import from LibBCast_FileStreamDistribution_TypesAndValues all;      group pduTypes {         group serviceGuideDeliveryDescriptor {             /** @desc SGDD structure as defined in SG 5.4.1.5.2             type record ServiceGuideDeliveryDescriptorType {                 AnyUri id optional,                 UInt version optional,                 NotificationReceptionType » NotificationReception optional,                 BSMLListType BSMLList optional,                 DescriptorEntryList DescriptorEntrys,                 PrivateExtType PrivateExt optional             }              /**              *              * @desc </pre>
--	---	--

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

» Messages.
* Reception information for general Notification
*
* In case of delivery over Broadcast channel,
* IPBroadcastDelivery specifies the address
» information
*
* for receiving Notification message. In case of
» delivery
*
* over Interaction channel, RequestURL specify
» address
*
* information for subscribing notification,
» PollURL specify
*
* address information for polling notification.
» When the
*
* Notification Message resource pointed by this
» element
*
* provides Notification Messages carrying
» Service Guide
*
* update, those SHALL relate to the currently
» bootstrapped
*
* Service Guide. If this element is present, at
» least one
*
* of the attributes "IPBroadcastDelivery",
» "RequestURL",
*
* or "PollURL" SHALL be present.
* @member
* IPBroadcastDelivery Provides IP multicast
» address and
*
* port number for reception of Notification
» Messages over
*
* the broadcast channel.
* @member
* RequestURL URL through which the terminal can
» subscribe
*
* to general Notification Messages; delivery
» over
*
* Interaction Channel.
* @member
* PollURL URL through which the terminal can
» poll general
*
* Notification Messages over Interaction
» Channel.
*/
type record NotificationReceptionType {
    IPBroadcastDeliveryType IPBroadcastDelivery
» optional,
    AnyUri RequestURL optional,

```

```

» Messages.
* Reception information for general Notification
*
* In case of delivery over Broadcast channel,
* IPBroadcastDelivery specifies the address
» information
*
* for receiving Notification message. In case of
» delivery
*
* over Interaction channel, RequestURL specify
» address
*
* information for subscribing notification,
» PollURL specify
*
* address information for polling notification.
» When the
*
* Notification Message resource pointed by this
» element
*
* provides Notification Messages carrying
» Service Guide
*
* update, those SHALL relate to the currently
» bootstrapped
*
* Service Guide. If this element is present, at
» least one
*
* of the attributes "IPBroadcastDelivery",
» "RequestURL",
*
* or "PollURL" SHALL be present.
* @member
* IPBroadcastDelivery Provides IP multicast
» address and
*
* port number for reception of Notification
» Messages over
*
* the broadcast channel.
* @member
* RequestURL URL through which the terminal can
» subscribe
*
* to general Notification Messages; delivery
» over
*
* Interaction Channel.
* @member
* PollURL URL through which the terminal can
» poll general
*
* Notification Messages over Interaction
» Channel.
*/
type record NotificationReceptionType {
    IPBroadcastDeliveryType IPBroadcastDelivery
» optional,
    AnyUri RequestURL optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        AnyUri PollURL optional
    }

    /**
     *
     * @desc
     *   Provides IP multicast address and port number
» for
     *   reception of Notification Messages over the
» broadcast
     *   channel.
     * @member
     *   Port General Notification Message delivery UDP
     *   destination port number; delivery over
» Broadcast Channel.
     * @member
     *   Address General Notification Message delivery
» IP
     *   multicast address; delivery over Broadcast
» Channel.
    */
    type record IPBroadcastDeliveryType {
        UInt port_,
        charstring address_ optional
    }

    type record BSMLListType {
        BSMSSelectorList BSMSSelectors
    }

    // could be further structured in future
    type record BSMSSelectorType {
        AnyUri id,
        AnyUri roamingRuleRequestAddress optional,
        BSMFilterCodeType BSMFilterCode optional,
        LanguageStringList Names,
        charstring RoamingRule optional
    }

    type set length (1.. infinity) of BSMSSelectorType
» BSMSSelectorList;

    type record BSMFilterCodeType {
        integer type_ (1..2),
        UInt8 serviceProviderCode optional,
        UInt8 corporateCode optional,

```

```

        AnyUri PollURL optional
    }

    /**
     *
     * @desc
     *   Provides IP multicast address and port number
» for
     *   reception of Notification Messages over the
» broadcast
     *   channel.
     * @member
     *   Port General Notification Message delivery UDP
     *   destination port number; delivery over
» Broadcast Channel.
     * @member
     *   Address General Notification Message delivery
» IP
     *   multicast address; delivery over Broadcast
» Channel.
    */
    type record IPBroadcastDeliveryType {
        UInt port_,
        charstring address_ optional
    }

    type record BSMLListType {
        BSMSSelectorList BSMSSelectors
    }

    // could be further structured in future
    type record BSMSSelectorType {
        AnyUri id,
        AnyUri roamingRuleRequestAddress optional,
        BSMFilterCodeType BSMFilterCode optional,
        LanguageStringList Names,
        charstring RoamingRule optional
    }

    type set length (1.. infinity) of BSMSSelectorType
» BSMSSelectorList;

    type record BSMFilterCodeType {
        integer type_ (1..2),
        UInt8 serviceProviderCode optional,
        UInt8 corporateCode optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        charstring serviceProviderName optional,
        charstring nonSmartCardCode optional,
        BSMFilterCodeTypeChoice choice optional
    }

    type union BSMFilterCodeTypeChoice {
        NetworkCode3GPPTType NetworkCode3GPP,
        NetworkCode3GPP2Type NetworkCode3GPP2
    }

    type record NetworkCode3GPPTType {
        Digit3Type mobileCountryCode optional,
        Digit23Type mobileNetworkCode optional,
        Digit2Type networkSubsetCode optional,
        Digit2Type networkSubsetCodeRangeStart
» optional,
        Digit2Type networkSubsetCodeRangeEnd
» optional,
        integer codeRangeStart optional,
        integer codeRangeEnd optional
    }

    type record NetworkCode3GPP2Type {
        Digit3Type mobileCountryCode optional,
        Digit23Type mobileNetworkCode optional,
        Digit4Type iRMBasedMIN optional,
        integer hRPDRealm optional,
        integer codeGroup optional
    }

    type charstring DigitType ("0" .. "9");

    type DigitType Digit2Type length (2);
    type DigitType Digit23Type length (2..3);
    type DigitType Digit3Type length (3);
    type DigitType Digit4Type length (4);

    type set length (
        1 .. infinity) of DescriptorEntryType
» DescriptorEntryList;

    type record DescriptorEntryType {
        GroupingCriteriaType GroupingCriteria
» optional,
        TransportType Transport optional,
        AlternativeAccessURLList

```

```

        charstring serviceProviderName optional,
        charstring nonSmartCardCode optional,
        BSMFilterCodeTypeChoice choice optional
    }

    type union BSMFilterCodeTypeChoice {
        NetworkCode3GPPTType NetworkCode3GPP,
        NetworkCode3GPP2Type NetworkCode3GPP2
    }

    type record NetworkCode3GPPTType {
        Digit3Type mobileCountryCode optional,
        Digit23Type mobileNetworkCode optional,
        Digit2Type networkSubsetCode optional,
        Digit2Type networkSubsetCodeRangeStart
» optional,
        Digit2Type networkSubsetCodeRangeEnd
» optional,
        integer codeRangeStart optional,
        integer codeRangeEnd optional
    }

    type record NetworkCode3GPP2Type {
        Digit3Type mobileCountryCode optional,
        Digit23Type mobileNetworkCode optional,
        Digit4Type iRMBasedMIN optional,
        integer hRPDRealm optional,
        integer codeGroup optional
    }

    type charstring DigitType ("0" .. "9");

    type DigitType Digit2Type length (2);
    type DigitType Digit23Type length (2..3);
    type DigitType Digit3Type length (3);
    type DigitType Digit4Type length (4);

    type set length (
        1 .. infinity) of DescriptorEntryType
» DescriptorEntryList;

    type record DescriptorEntryType {
        GroupingCriteriaType GroupingCriteria
» optional,
        TransportType Transport optional,
        AlternativeAccessURLList

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

<pre>» AlternativeAccessURLs optional,     ServiceGuideDeliveryUnitList » ServiceGuideDeliveryUnits     }</pre>		<pre>» AlternativeAccessURLs optional,     ServiceGuideDeliveryUnitList » ServiceGuideDeliveryUnits     }</pre>
<pre>// change 22 (WK 6): further structured TTCN-3 types type record GroupingCriteriaType {     TimeGroupingCriteriaType TimeGroupingCriteria » optional,      GenreType GenreGroupingCriteria optional,     BSMSSelectorIDRefList BSMSSelectors optional,     ServiceCriteriaType ServiceCriteria optional }  // change 22 (WK 6): further structured TTCN-3 types type record ServiceCriteriaType {     XmlElementAnyURIValue text_ }</pre>	<>	<pre>type charstring GroupingCriteriaType;     // could be further structured in future</pre>
<pre>type record TransportType {     charstring ipAddress,     UInt16 port_,     charstring srcIpAddress optional,     UInt16 transmissionSessionID,     UInt32 versionIdLength optional, // VERY » optional      boolean hasFDT optional }  type set length (     1 .. infinity) of AnyUri » AlternativeAccessURLList;  type set length (     1     .. infinity) of » ServiceGuideDeliveryUnitType ServiceGuideDeliveryUnitList;  type record ServiceGuideDeliveryUnitType {     UInt transportObjectID,     AnyUri contentLocation,     UInt validFrom optional,     UInt validTo optional,     FragmentList Fragments }</pre>	=	<pre>type record TransportType {     charstring ipAddress,     UInt16 port_,     charstring srcIpAddress optional,     UInt16 transmissionSessionID,     UInt32 versionIdLength optional, // VERY » optional      boolean hasFDT optional }  type set length (     1 .. infinity) of AnyUri » AlternativeAccessURLList;  type set length (     1     .. infinity) of » ServiceGuideDeliveryUnitType ServiceGuideDeliveryUnitList;  type record ServiceGuideDeliveryUnitType {     UInt transportObjectID,     AnyUri contentLocation,     UInt validFrom optional,     UInt validTo optional,     FragmentList Fragments }</pre>

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

type set length (1 .. infinity) of FragmentType
» FragmentList;

type record FragmentType {
    UInt transportID optional,
    // mandatory if send via broadcast!
    AnyUri id,
    UInt version,
    UInt validFrom optional,
    UInt validTo optional,
    UInt8 fragmentEncoding,
    UInt8 fragmentType optional,
    FragmentGroupingCriteriaType GroupingCriteria
» optional
}

type record FragmentGroupingCriteriaType {
    TimeGroupingCriteriaType TimeGroupingCriteria
» optional,
    BSMSSelectorIDRefList BSMSSelectors
}

type record TimeGroupingCriteriaType {
    UInt startTime,
    UInt endTime
}

type record BSMSSelectorIDRefType {
    AnyUri idRef
}

type set length (1 .. infinity) of
» BSMSSelectorIDRefType BSMSSelectorIDRefList;

type charstring PrivateExtType;
    // could be further structured in future
} // end group serviceGuideDeliveryDescriptor
group serviceGuideResponse {
    type record ServiceGuideResponse {
        UInt status,
        // Note: should use only values 0, 7, 8, 17,
» 18, 20, 21, 23, 28..255
        ServiceGuideDeliveryDescriptorList
» ServiceGuideDeliveryDescriptors optional,
        PrivateExtType privateExt optional
    }
}

```

```

type set length (1 .. infinity) of FragmentType
» FragmentList;

type record FragmentType {
    UInt transportID optional,
    // mandatory if send via broadcast!
    AnyUri id,
    UInt version,
    UInt validFrom optional,
    UInt validTo optional,
    UInt8 fragmentEncoding,
    UInt8 fragmentType optional,
    FragmentGroupingCriteriaType GroupingCriteria
» optional
}

type record FragmentGroupingCriteriaType {
    TimeGroupingCriteriaType TimeGroupingCriteria
» optional,
    BSMSSelectorIDRefList BSMSSelectors
}

type record TimeGroupingCriteriaType {
    UInt startTime,
    UInt endTime
}

type record BSMSSelectorIDRefType {
    AnyUri idRef
}

type set length (1 .. infinity) of
» BSMSSelectorIDRefType BSMSSelectorIDRefList;

type charstring PrivateExtType;
    // could be further structured in future
} // end group serviceGuideDeliveryDescriptor
group serviceGuideResponse {
    type record ServiceGuideResponse {
        UInt status,
        // Note: should use only values 0, 7, 8, 17,
» 18, 20, 21, 23, 28..255
        ServiceGuideDeliveryDescriptorList
» ServiceGuideDeliveryDescriptors optional,
        PrivateExtType privateExt optional
    }
}

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        type set length (1 .. infinity) of
        ServiceGuideDeliveryDescriptorType
» ServiceGuideDeliveryDescriptorList;

        } // end group serviceGuideResponse
        group serviceGuideDeliveryUnit {
            type set of ServiceGuideFragment
» ServiceGuideDeliveryUnit;

            type record ServiceGuideFragment {
                UInt32 fragmentTransportId,
                UInt32 fragmentVersion,
                ServiceGuideFragmentType fragmentType
            }

            type union ServiceGuideFragmentType {
                XmlFragment xmlFragment,
                SdpFragment sdpFragment,
                MbmsUsbdFragment mbmsUsbdFragment,
                FileAdp adpFragment
            }

            /**
             *
             * @remark
             *   This type is binary (NOT XML) encoded! The
» codec should
             *   concatenate/seperate list elements based on
» the fixed
             *   list element length
             */
            type record length (
                0 .. infinity) of UnitHeaderInfo
» UnitHeaderInfoList;

            /**
             *
             * @remark
             *   This type is binary (NOT XML) encoded! The
» codec must
             *   compute and add/remove the offset field
             */
            type record UnitHeaderInfo {
                UInt32 fragmentTransportId,
                UInt32 fragmentVersion

```

```

        type set length (1 .. infinity) of
        ServiceGuideDeliveryDescriptorType
» ServiceGuideDeliveryDescriptorList;

        } // end group serviceGuideResponse
        group serviceGuideDeliveryUnit {
            type set of ServiceGuideFragment
» ServiceGuideDeliveryUnit;

            type record ServiceGuideFragment {
                UInt32 fragmentTransportId,
                UInt32 fragmentVersion,
                ServiceGuideFragmentType fragmentType
            }

            type union ServiceGuideFragmentType {
                XmlFragment xmlFragment,
                SdpFragment sdpFragment,
                MbmsUsbdFragment mbmsUsbdFragment,
                FileAdp adpFragment
            }

            /**
             *
             * @remark
             *   This type is binary (NOT XML) encoded! The
» codec should
             *   concatenate/seperate list elements based on
» the fixed
             *   list element length
             */
            type record length (
                0 .. infinity) of UnitHeaderInfo
» UnitHeaderInfoList;

            /**
             *
             * @remark
             *   This type is binary (NOT XML) encoded! The
» codec must
             *   compute and add/remove the offset field
             */
            type record UnitHeaderInfo {
                UInt32 fragmentTransportId,
                UInt32 fragmentVersion

```



Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

    }
    with {
        variant (fragmentTransportId) "most
» significant byte first";
        variant (fragmentVersion) "most significant
» byte first";
    }

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec must
     *   concatenate/seperate list elements based on
» the list
     *   element type structure, and compute and
» add/remove the
     *   fragmentEncoding field for each list element
     */
    type set length (0 .. infinity) of XmlFragment
» XmlFragmentList;

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec must
     *   compute and add/remove the fragmentType field.
» In case of
     *   the other fragment the union codec must not
» add/remove
     *   the fragmentType field.
     */
    type union XmlFragment {
        ServiceType Service,
        ContentType Content,
        ScheduleType Schedule,
        AccessType Access,
        PurchaseItemType PurchaseItem,
        PurchaseDataType PurchaseData,
        PurchaseChannelType PurchaseChannel,
        PreviewDataType PreviewData,
        InteractivityDataType InteractivityData,
        /* This alternative may be used to enter
» "unspecified(0)" or extensions

```

```

    }
    with {
        variant (fragmentTransportId) "most
» significant byte first";
        variant (fragmentVersion) "most significant
» byte first";
    }

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec must
     *   concatenate/seperate list elements based on
» the list
     *   element type structure, and compute and
» add/remove the
     *   fragmentEncoding field for each list element
     */
    type set length (0 .. infinity) of XmlFragment
» XmlFragmentList;

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec must
     *   compute and add/remove the fragmentType field.
» In case of
     *   the other fragment the union codec must not
» add/remove
     *   the fragmentType field.
     */
    type union XmlFragment {
        ServiceType Service,
        ContentType Content,
        ScheduleType Schedule,
        AccessType Access,
        PurchaseItemType PurchaseItem,
        PurchaseDataType PurchaseData,
        PurchaseChannelType PurchaseChannel,
        PreviewDataType PreviewData,
        InteractivityDataType InteractivityData,
        /* This alternative may be used to enter
» "unspecified(0)" or extensions

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        OtherXmlFragment otherXmlFragment
    }

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec shall
     *   simply concatenate/separate fields
     */
    type record OtherXmlFragment {
        UInt8 fragmentType,
        charstring fragment
    }
    with {
        variant (fragmentType) "most significant byte
» first";
    }

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec must
     *   concatenate/separate list elements based on
» the list
     *   element type structure
     */
    type set length (0 .. infinity) of SdpFragment
» SdpFragmentList;

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec shall
     *   simply concatenate/separate fields
     */
    type record SdpFragment {
        UInt32 validFrom,
        UInt32 validTo,
        charstring fragmentId,
        Sdp sdpFragment
    }
    with {
        variant (validFrom) "most significant byte

```

```

        OtherXmlFragment otherXmlFragment
    }

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec shall
     *   simply concatenate/separate fields
     */
    type record OtherXmlFragment {
        UInt8 fragmentType,
        charstring fragment
    }
    with {
        variant (fragmentType) "most significant byte
» first";
    }

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec must
     *   concatenate/separate list elements based on
» the list
     *   element type structure
     */
    type set length (0 .. infinity) of SdpFragment
» SdpFragmentList;

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec shall
     *   simply concatenate/separate fields
     */
    type record SdpFragment {
        UInt32 validFrom,
        UInt32 validTo,
        charstring fragmentId,
        Sdp sdpFragment
    }
    with {
        variant (validFrom) "most significant byte

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

» first";
                                variant (validTo) "most signifcant byte
» first";
                                }
                                /**
                                *
                                * @remark
                                *   This type is binary (NOT XML) encoded! The
» codec must
                                *   concatenate/seperate list elements based on
» the list
                                *   element type structure
                                */
                                type record length (
                                0 .. infinity) of MbmsUsbdFragment
» MbmsUsbdFragmentList;

                                /**
                                *
                                * @remark
                                *   This type is binary (NOT XML) encoded!
                                */
                                type record MbmsUsbdFragment {
                                    UInt32 validFrom,
                                    UInt32 validTo,
                                    charstring fragmentId,
                                    Usbd usbdFragment
                                }
                                with {
                                    variant (validFrom) "most signifcant byte
» first";
                                    variant (validTo) "most signifcant byte
» first";
                                }

                                /**
                                *
                                * @remark
                                *   This type is binary (NOT XML) encoded! The
» codec must
                                *   concatenate/seperate list elements based on
» the list
                                *   element type structure
                                */
                                type record length (0 .. infinity) of FileAdp

```

```

» first";
                                variant (validTo) "most signifcant byte
» first";
                                }
                                /**
                                *
                                * @remark
                                *   This type is binary (NOT XML) encoded! The
» codec must
                                *   concatenate/seperate list elements based on
» the list
                                *   element type structure
                                */
                                type record length (
                                0 .. infinity) of MbmsUsbdFragment
» MbmsUsbdFragmentList;

                                /**
                                *
                                * @remark
                                *   This type is binary (NOT XML) encoded!
                                */
                                type record MbmsUsbdFragment {
                                    UInt32 validFrom,
                                    UInt32 validTo,
                                    charstring fragmentId,
                                    Usbd usbdFragment
                                }
                                with {
                                    variant (validFrom) "most signifcant byte
» first";
                                    variant (validTo) "most signifcant byte
» first";
                                }

                                /**
                                *
                                * @remark
                                *   This type is binary (NOT XML) encoded! The
» codec must
                                *   concatenate/seperate list elements based on
» the list
                                *   element type structure
                                */
                                type record length (0 .. infinity) of FileAdp

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

» FileAdpList;

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec shall
     *   simply concatenate/separate fields
     */
    type record FileAdp {
        UInt32 validFrom,
        UInt32 validTo,
        charstring fragmentId,
        FileAssociatedProcedureDescription fileAdp
    }
    with {
        variant (validFrom) "most significant byte
» first";
        variant (validTo) "most significant byte
» first";
    }

    group fragmentEncoding {
        const UInt8 c_XML_BCAST_SG_fragment := 0;
        const UInt8 c_SDP_fragment := 1;
        const UInt8 c_MBMS_USD_fragment := 2;
        const UInt8 c_XML_ADP_fragment := 3;
    }

    group serviceGuideFragmentTypes {
        const UInt8 c_SG_Unspecified_Fragment := 0;
        const UInt8 c_SG_Service_Fragment := 1;
        const UInt8 c_SG_Content_Fragment := 2;
        const UInt8 c_SG_Schedule_Fragment := 3;
        const UInt8 c_SG_Access_Fragment := 4;
        const UInt8 c_SG_PurchaseItem_Fragment := 5;
        const UInt8 c_SG_PurchaseData_Fragment := 6;
        const UInt8 c_SG_PurchaseChannel_Fragment :=
» 7;

        const UInt8 c_SG_PreviewData_Fragment := 8;
        const UInt8 c_SG_InteractivityData_Fragment
» := 9;
    }

} // end group serviceGuideDeliveryUnit

```

```

» FileAdpList;

    /**
     *
     * @remark
     *   This type is binary (NOT XML) encoded! The
» codec shall
     *   simply concatenate/separate fields
     */
    type record FileAdp {
        UInt32 validFrom,
        UInt32 validTo,
        charstring fragmentId,
        FileAssociatedProcedureDescription fileAdp
    }
    with {
        variant (validFrom) "most significant byte
» first";
        variant (validTo) "most significant byte
» first";
    }

    group fragmentEncoding {
        const UInt8 c_XML_BCAST_SG_fragment := 0;
        const UInt8 c_SDP_fragment := 1;
        const UInt8 c_MBMS_USD_fragment := 2;
        const UInt8 c_XML_ADP_fragment := 3;
    }

    group serviceGuideFragmentTypes {
        const UInt8 c_SG_Unspecified_Fragment := 0;
        const UInt8 c_SG_Service_Fragment := 1;
        const UInt8 c_SG_Content_Fragment := 2;
        const UInt8 c_SG_Schedule_Fragment := 3;
        const UInt8 c_SG_Access_Fragment := 4;
        const UInt8 c_SG_PurchaseItem_Fragment := 5;
        const UInt8 c_SG_PurchaseData_Fragment := 6;
        const UInt8 c_SG_PurchaseChannel_Fragment :=
» 7;

        const UInt8 c_SG_PreviewData_Fragment := 8;
        const UInt8 c_SG_InteractivityData_Fragment
» := 9;
    }

} // end group serviceGuideDeliveryUnit

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ftcn  
(Fortsetzung)

```

    group interactivityMediaDocument {
        type record InteractivityMediaDocumentType {
            AnyUri groupId,
            UInt16 groupPosition,
            AnyUri id,
            UInt version,
            UInt vaildFrom optional,
            UInt valudTo optional,
            MediaObjectGroupList MediaObjectGroups,
            PrivateExtType PrivateExt optional
        }

        type set length(1..infinity) of MediaObjectGroupType
    » MediaObjectGroupList ;

        type record MediaObjectGroupType {
            AnyUri id,
            boolean startMediaFlag,
            ActionDescriptorType ActionDescriptor

            BackOffTimingType BackOffTiming optional,
            MediaObjectSetList MediaObjectSets optional,
            SMSTemplateType SMSTemplate optional,
            EmailTemplateType EmailTemplate optional,
            VoiceCallType VoiceCall optional,
            WebLinkType WebLink optional,
            LanguageString AlternativeText optional
        }

        type set of MediaObjectSetType MediaObjectSetList;

        type record MediaObjectSetType {
            UInt32 relativePreference optional,
            charstring Content_Type,
            AnyUri Content_Location,
            LanguageStringList Descriptions optional,
            ObjectList Objects optional,
            FileType File optional
        } with {variant "replace '_' with '-'" }

        type record WebLinkType {
            UInt relativePreference optional,
            AnyUri webUrl,
            LanguageStringList Descriptions optional
        }

```

```

    group interactivityMediaDocument {
        type record InteractivityMediaDocumentType {
            AnyUri groupId,
            UInt16 groupPosition,
            AnyUri id,
            UInt version,
            UInt vaildFrom optional,
            UInt valudTo optional,
            MediaObjectGroupList MediaObjectGroups,
            PrivateExtType PrivateExt optional
        }

        type set length(1..infinity) of MediaObjectGroupType
    » MediaObjectGroupList ;

        type record MediaObjectGroupType {
            AnyUri id,
            boolean startMediaFlag,
            ActionDescriptorType ActionDescriptor

            BackOffTimingType BackOffTiming optional,
            MediaObjectSetList MediaObjectSets optional,
            SMSTemplateType SMSTemplate optional,
            EmailTemplateType EmailTemplate optional,
            VoiceCallType VoiceCall optional,
            WebLinkType WebLink optional,
            LanguageString AlternativeText optional
        }

        type set of MediaObjectSetType MediaObjectSetList;

        type record MediaObjectSetType {
            UInt32 relativePreference optional,
            charstring Content_Type,
            AnyUri Content_Location,
            LanguageStringList Descriptions optional,
            ObjectList Objects optional,
            FileType File optional
        } with {variant "replace '_' with '-'" }

        type record WebLinkType {
            UInt relativePreference optional,
            AnyUri webUrl,
            LanguageStringList Descriptions optional
        }

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        type record VoiceCallType {
            UInt relativePreference optional,
            LanguageStringList Descriptions optional,
            set of AnyUri PhoneNumber optional
        }

        type record EmailTemplateType {
            UInt relativePreference optional,
            charstring toHeader,
            charstring ccHeader optional,
            charstring bccHeader optional,
            charstring subjectHeader optional,
            LanguageStringList Descriptions optional,
            charstring MessageBody optional
        }

        type record SMSTemplateType {
            UInt relativePreference optional,
            DescriptionList Descriptions optional,
            SelectChoiceList SelectChoices optional
        }

        type set length (1 .. infinity) of DescriptionType
» DescriptionList;

        type record DescriptionType {
            XmlLang lang optional,
            XmlElementStringValue text,
            XmlElementStringValue text_
        }

        type set of SelectChoiceType SelectChoiceList;

        type record SelectChoiceType {
            AnyUri smsURI,
            LanguageStringList ChoiceTexts optional
        }

        type set of ObjectType ObjectList;

        type record ObjectType {
            AnyUri Content_Location optional,
            charstring Content_Type,
            boolean start_ optional,

```

```

        type record VoiceCallType {
            UInt relativePreference optional,
            LanguageStringList Descriptions optional,
            set of AnyUri PhoneNumber optional
        }

        type record EmailTemplateType {
            UInt relativePreference optional,
            charstring toHeader,
            charstring ccHeader optional,
            charstring bccHeader optional,
            charstring subjectHeader optional,
            LanguageStringList Descriptions optional,
            charstring MessageBody optional
        }

        type record SMSTemplateType {
            UInt relativePreference optional,
            DescriptionList Descriptions optional,
            SelectChoiceList SelectChoices optional
        }

        type set length (1 .. infinity) of DescriptionType

        type record DescriptionType {
            XmlLang lang optional,
            XmlElementStringValue text,
            XmlElementStringValue text_
        }

        type set of SelectChoiceType SelectChoiceList;

        type record SelectChoiceType {
            AnyUri smsURI,
            LanguageStringList ChoiceTexts optional
        }

        type set of ObjectType ObjectList;

        type record ObjectType {
            AnyUri Content_Location optional,
            charstring Content_Type,
            boolean start_ optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

<pre>                 PartTypeList PartTypes optional             } with {variant "replace '_' with '-'" }              type set of charstring PartTypeList;              type record ActionDescriptorType {                 UInt inputAllowedTime optional,                 AnyUri onTimeOutPointer optional,                 boolean updateFlag optional,                 AnyUri onActionPointer optional             }         } </pre>		<pre>                 PartTypeList PartTypes optional             } with {variant "replace '_' with '-'" }              type set of charstring PartTypeList;              type record ActionDescriptorType {                 UInt inputAllowedTime optional,                 AnyUri onTimeOutPointer optional,                 boolean updateFlag optional,                 AnyUri onActionPointer optional             }         } </pre>
<pre> // change 23 (WK 6): type definitions for DVB-H bootstrapping group dvbhBootstrapDescriptor {     type record DvbhEsgDescriptor {         ESGProviderDiscoveryType     }     ESGProviderDiscovery,         ESGAccessType ESGAccess     }      // * @desc ESGProviderDiscovery structure as defined     » in ETSI TS 102 471 section 9.1.1         type record ESGProviderDiscoveryType {             ServiceProviderList ServiceProviders         } with {variant "namespacedef nsprefix = '' nsuri =     » 'urn:dvb:ipdc:esgbs:2005'"};          type record ServiceProviderType {             charstring format,             ProviderURIType ProviderURI,             ProviderNameType ProviderName,             ProviderIDType ProviderID         }          type set length (1 .. infinity) of     » ServiceProviderType ServiceProviderList;          type record ProviderURIType {             XmlElementStringValue text_         }          type record ProviderNameType {             XmlElementStringValue text_         } </pre>	+-	

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

<pre>         type record ProviderIDType {             XmlElementStringValue text_         }          type record ESGAccessType {             UInt8 ESGEntryVersion,             boolean MultipleStreamTransport,             boolean IPVersion6,             UInt16 ProviderID,             charstring SrcIpAddress,             charstring DstIpAddress,             UInt16 DstIpPort,             UInt16 TSI         }      } // end of group dvbhdBootstrapDescriptor </pre>		
<pre>     } // end group pduTypes      group xmlFragments {          group commonTypes {              type set length (1 .. infinity) of AnyXmlElement » AnyXmlElementList;              type record AnyXmlElement {                 AnyXmlAttributeList attributes optional,                 charstring name_,                 XmlElementStringValue text_ optional,                 AnyXmlElementList childs optional             }              type set length (1 .. infinity) of AnyXmlAttribute » AnyXmlAttributeList;              type record AnyXmlAttribute {                 charstring name_,                 charstring text_ optional             }              type DateTime XmlElementDateTimeValue;              type UInt8 XmlElementUnsignedByteValue;              type UInt16 XmlElementUnsignedShortValue; </pre>	=	<pre>     } // end group pduTypes      group xmlFragments {          group commonTypes {              type set length (1 .. infinity) of AnyXmlElement » AnyXmlElementList;              type record AnyXmlElement {                 AnyXmlAttributeList attributes optional,                 charstring name_,                 XmlElementStringValue text_ optional,                 AnyXmlElementList childs optional             }              type set length (1 .. infinity) of AnyXmlAttribute » AnyXmlAttributeList;              type record AnyXmlAttribute {                 charstring name_,                 charstring text_ optional             }              type DateTime XmlElementDateTimeValue;              type UInt8 XmlElementUnsignedByteValue;              type UInt16 XmlElementUnsignedShortValue; </pre>



Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        type UInt XmlElementUnsignedIntValue;

        type charstring XmlElementStringValue;

        type float XmlElementDecimalValue;

        type AnyUri XmlElementAnyURIValue;

        type charstring XmlLang with {variant "namespace
» prefix = 'xml'"};

        type charstring XmlSpace with {variant "namespace
» prefix = 'xml'"};

        type XmlSpace PreserveXmlSpace ("preserve");

        type set length (1 .. infinity) of LanguageString
» LanguageStringList;

        type record      LanguageString {
            XmlLang lang optional,
            XmlElementStringValue text_
        }

        type record      EncodedLanguageString {
            XmlLang lang optional,
            charstring encoding optional,
            XmlElementStringValue text_
        }

        type set length (1 .. infinity) of
» EncodedLanguageString EncodedLanguageStringList;

        // the AudioOrTextLanguageType type is an extension
» of the LanguageString type
        type record AudioOrTextLanguageType {
            XmlLang lang optional,
            AnyUri id optional,
            charstring languageSDPTag,
            XmlElementStringValue text_
        }

        type set length (1 .. infinity) of
» AudioOrTextLanguageType AudioOrTextLanguageList;

```

```

        type UInt XmlElementUnsignedIntValue;

        type charstring XmlElementStringValue;

        type float XmlElementDecimalValue;

        type AnyUri XmlElementAnyURIValue;

        type charstring XmlLang with {variant "namespace
» prefix = 'xml'"};

        type charstring XmlSpace with {variant "namespace
» prefix = 'xml'"};

        type XmlSpace PreserveXmlSpace ("preserve");

        type set length (1 .. infinity) of LanguageString
» LanguageStringList;

        type record      LanguageString {
            XmlLang lang optional,
            XmlElementStringValue text_
        }

        type record      EncodedLanguageString {
            XmlLang lang optional,
            charstring encoding optional,
            XmlElementStringValue text_
        }

        type set length (1 .. infinity) of
» EncodedLanguageString EncodedLanguageStringList;

        // the AudioOrTextLanguageType type is an extension
» of the LanguageString type
        type record AudioOrTextLanguageType {
            XmlLang lang optional,
            AnyUri id optional,
            charstring languageSDPTag,
            XmlElementStringValue text_
        }

        type set length (1 .. infinity) of
» AudioOrTextLanguageType AudioOrTextLanguageList;

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ftcn  
(Fortsetzung)

```

        type record ExtensionType {
            AnyUri url,
            LanguageStringList Descriptions optional
        }

        type set length (1 .. infinity) of ExtensionType
» ExtensionList;

        type record ParentalRatingType {
            charstring ratingSystem optional,
            XmlElementStringValue text_
        }

        type set length (1 .. infinity) of ParentalRatingType
» ParentalRatingList;
    }

    group serviceFragment {
        type record ServiceType {
            // The attributes
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AnyUri globalServiceID optional,
            UInt16 weight optional,
            //boolean serviceContentProtection optional,
            charstring baseCID optional,
            boolean emergency optional,
            // The Elements
            ProtectionKeyIDList ProtectionKeyIDs

            ServiceTypeRangeList ServiceTypes optional,
            // Start of program guide information
            //NameList Names,
            LanguageStringList Names,
            LanguageStringList Descriptions optional,
            AudioOrTextLanguageList AudioLanguages

            AudioOrTextLanguageList TextLanguages

            ParentalRatingList ParentalRatings optional,
            TargetUserProfileList TargetUserProfiles

            GenreList Genres optional,
            ExtensionList Extensions optional,

```

```

        type record ExtensionType {
            AnyUri url,
            LanguageStringList Descriptions optional
        }

        type set length (1 .. infinity) of ExtensionType
» ExtensionList;

        type record ParentalRatingType {
            charstring ratingSystem optional,
            XmlElementStringValue text_
        }

        type set length (1 .. infinity) of ParentalRatingType
» ParentalRatingList;
    }

    group serviceFragment {
        type record ServiceType {
            // The attributes
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AnyUri globalServiceID optional,
            UInt16 weight optional,
            //boolean serviceContentProtection optional,
            charstring baseCID optional,
            boolean emergency optional,
            // The Elements
            ProtectionKeyIDList ProtectionKeyIDs

            ServiceTypeRangeList ServiceTypes optional,
            // Start of program guide information
            //NameList Names,
            LanguageStringList Names,
            LanguageStringList Descriptions optional,
            AudioOrTextLanguageList AudioLanguages

            AudioOrTextLanguageList TextLanguages

            ParentalRatingList ParentalRatings optional,
            TargetUserProfileList TargetUserProfiles

            GenreList Genres optional,
            ExtensionList Extensions optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        // End of Service guide
        PreviewDataReferenceList
» PreviewDataReferences optional,
        BroadcastAreaType BroadcastArea optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    type record ProtectionKeyIDType {
        UInt8 type_,
        // simple content base64Binary
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of
» ProtectionKeyIDType ProtectionKeyIDList;

    type record ProtectionKeyIDMinMaxType {
        UInt8 type_,
        UInt min optional,
        UInt max optional,
        // simple content base64Binary
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of
» ProtectionKeyIDMinMaxType ProtectionKeyIDMinMaxList;

    // Key Domain ID concatenated with SEK/PEK ID, where
» both values are as used in the Smartcard Profile [BCAST10-ServContProt].
    const XmlElementStringValue
» c_protectionKeyId_keyDomainId := "0";

    // simpleType unsignedByte
    type record ServiceTypeRangeType {
        XmlElementUnsignedByteValue text_
    }

    type set length (1 .. infinity) of
» ServiceTypeRangeType ServiceTypeRangeList;

    const XmlElementUnsignedByteValue
» c_unspecified_ServiceType := 0;
    const XmlElementUnsignedByteValue
» c_basicTv_ServiceType := 1;
    const XmlElementUnsignedByteValue

```

```

        // End of Service guide
        PreviewDataReferenceList
» PreviewDataReferences optional,
        BroadcastAreaType BroadcastArea optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    type record ProtectionKeyIDType {
        UInt8 type_,
        // simple content base64Binary
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of
» ProtectionKeyIDType ProtectionKeyIDList;

    type record ProtectionKeyIDMinMaxType {
        UInt8 type_,
        UInt min optional,
        UInt max optional,
        // simple content base64Binary
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of
» ProtectionKeyIDMinMaxType ProtectionKeyIDMinMaxList;

    // Key Domain ID concatenated with SEK/PEK ID, where
» both values are as used in the Smartcard Profile [BCAST10-ServContProt].
    const XmlElementStringValue
» c_protectionKeyId_keyDomainId := "0";

    // simpleType unsignedByte
    type record ServiceTypeRangeType {
        XmlElementUnsignedByteValue text_
    }

    type set length (1 .. infinity) of
» ServiceTypeRangeType ServiceTypeRangeList;

    const XmlElementUnsignedByteValue
» c_unspecified_ServiceType := 0;
    const XmlElementUnsignedByteValue
» c_basicTv_ServiceType := 1;
    const XmlElementUnsignedByteValue

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

» c_basicRadio_ServiceType := 2;
    const XmlElementUnsignedByteValue
» c_riServices_ServiceType := 3;
    const XmlElementUnsignedByteValue
» c_cachecast_ServiceType := 4;
    const XmlElementUnsignedByteValue
» c_fileDownloadServices_ServiceType := 5;
    const XmlElementUnsignedByteValue
» c_softwareManagementServices_ServiceType := 6;
    const XmlElementUnsignedByteValue
» c_notification_ServiceType := 7;
    const XmlElementUnsignedByteValue
» c_serviceGuide_ServiceType := 8;
    const XmlElementUnsignedByteValue
» c_terminalProvisioning_ServiceType := 9;

    type record TargetUserProfileType {
        charstring attributeName,
        charstring attributeValue
    }

    type set length (1 .. infinity) of
» TargetUserProfileType TargetUserProfileList;

    type record GenreType {
        XmlLang lang optional,
        AnyUri href optional,
        GenreTypeType type_ optional,
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of GenreType
» GenreList;

    type enumerated GenreTypeType { main, secondary,
» other };

    type record PreviewDataReferenceType {
        AnyUri idRef,
        UInt8 usage
    }

    type set length (1 .. infinity) of
» PreviewDataReferenceType PreviewDataReferenceList;

    const UInt8 c_unspecified_usage := 0;

```

```

» c_basicRadio_ServiceType := 2;
    const XmlElementUnsignedByteValue
» c_riServices_ServiceType := 3;
    const XmlElementUnsignedByteValue
» c_cachecast_ServiceType := 4;
    const XmlElementUnsignedByteValue
» c_fileDownloadServices_ServiceType := 5;
    const XmlElementUnsignedByteValue
» c_softwareManagementServices_ServiceType := 6;
    const XmlElementUnsignedByteValue
» c_notification_ServiceType := 7;
    const XmlElementUnsignedByteValue
» c_serviceGuide_ServiceType := 8;
    const XmlElementUnsignedByteValue
» c_terminalProvisioning_ServiceType := 9;

    type record TargetUserProfileType {
        charstring attributeName,
        charstring attributeValue
    }

    type set length (1 .. infinity) of
» TargetUserProfileType TargetUserProfileList;

    type record GenreType {
        XmlLang lang optional,
        AnyUri href optional,
        GenreTypeType type_ optional,
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of GenreType
» GenreList;

    type enumerated GenreTypeType { main, secondary,
» other };

    type record PreviewDataReferenceType {
        AnyUri idRef,
        UInt8 usage
    }

    type set length (1 .. infinity) of
» PreviewDataReferenceType PreviewDataReferenceList;

    const UInt8 c_unspecified_usage := 0;

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

const UInt8 c_serviceByServiceSwitching_usage := 1;
const UInt8 c_serviceGuideBrowsing_usage := 2;
const UInt8 c_servicePreview_usage := 3;
const UInt8 c_barker_usage := 4;
const UInt8 c_alternativeToBlackout_usage := 5;

type record BroadcastAreaType {
    boolean polarity optional,
    TargetAreaList TargetAreas optional,
    hor_accList hor_accs optional
}

type set length (1 .. infinity) of TargetAreaType
» TargetAreaList;

type record TargetAreaType {
    shapeType shape optional,
    ccType cc optional,
    LanguageStringList name_areas optional,
    ZipCodeType ZipCode optional,
    CellTargetAreaType CellTargetArea optional
}

type record shapeType {
    AnyXmlElementList elements optional
}

type record ccType {
    XmlElementUnsignedShortValue text_
}

type record ZipCodeType {
    XmlElementStringValue text_
}

type record CellTargetAreaType {
    UInt8 type_,
    CellAreaType CellArea
}

type record CellAreaType {
    UInt16 value_,
    PP2CellIDList PP2CellIDs optional
}

type record PP2CellIDType {

```

```

const UInt8 c_serviceByServiceSwitching_usage := 1;
const UInt8 c_serviceGuideBrowsing_usage := 2;
const UInt8 c_servicePreview_usage := 3;
const UInt8 c_barker_usage := 4;
const UInt8 c_alternativeToBlackout_usage := 5;

type record BroadcastAreaType {
    boolean polarity optional,
    TargetAreaList TargetAreas optional,
    hor_accList hor_accs optional
}

type set length (1 .. infinity) of TargetAreaType
» TargetAreaList;

type record TargetAreaType {
    shapeType shape optional,
    ccType cc optional,
    LanguageStringList name_areas optional,
    ZipCodeType ZipCode optional,
    CellTargetAreaType CellTargetArea optional
}

type record shapeType {
    AnyXmlElementList elements optional
}

type record ccType {
    XmlElementUnsignedShortValue text_
}

type record ZipCodeType {
    XmlElementStringValue text_
}

type record CellTargetAreaType {
    UInt8 type_,
    CellAreaType CellArea
}

type record CellAreaType {
    UInt16 value_,
    PP2CellIDList PP2CellIDs optional
}

type record PP2CellIDType {

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ftcn  
(Fortsetzung)

```

        XmlElementUnsignedIntValue text_
    }

    type set length (1 .. infinity) of PP2CellIDType
» PP2CellIDList;

    type record hor_accType {
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of hor_accType
» hor_accList;

    type set length (1 .. infinity) of TermsOfUseType
» TermsOfUseList;

    type record TermsOfUseType {
        UInt8 type_,
        AnyUri id,
        boolean userConsentRequired,
        CountryList Countrys,
        LanguageType Language,
        TermsOfUseTypeChoice choice
    }

    type set length (1 .. infinity) of CountryType
» CountryList;

    type record CountryType {
        XmlElementStringValue text_
    }

    type record LanguageType {
        XmlElementStringValue text_
    }

    type union TermsOfUseTypeChoice {
        PreviewDataIDRefList PreviewDataIDRefs,
        TermsOfUseTextType TermsOfUseText
    }

    type set length (1 .. infinity) of
» PreviewDataIDRefType PreviewDataIDRefList;

    type record PreviewDataIDRefType {
        AnyUri text_

```

```

        XmlElementUnsignedIntValue text_
    }

    type set length (1 .. infinity) of PP2CellIDType
» PP2CellIDList;

    type record hor_accType {
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of hor_accType
» hor_accList;

    type set length (1 .. infinity) of TermsOfUseType
» TermsOfUseList;

    type record TermsOfUseType {
        UInt8 type_,
        AnyUri id,
        boolean userConsentRequired,
        CountryList Countrys,
        LanguageType Language,
        TermsOfUseTypeChoice choice
    }

    type set length (1 .. infinity) of CountryType
» CountryList;

    type record CountryType {
        XmlElementStringValue text_
    }

    type record LanguageType {
        XmlElementStringValue text_
    }

    type union TermsOfUseTypeChoice {
        PreviewDataIDRefList PreviewDataIDRefs,
        TermsOfUseTextType TermsOfUseText
    }

    type set length (1 .. infinity) of
» PreviewDataIDRefType PreviewDataIDRefList;

    type record PreviewDataIDRefType {
        AnyUri text_

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

    }

    type record TermsOfUseTextType {
        XmlElementStringValue text_
    }
}

group scheduleFragment {
    type record ScheduleType {
        AnyUri id,
        UInt version,
        boolean defaultSchedule optional,
        boolean onDemand optional,
        UInt validFrom optional,
        UInt validTo optional,
        ScheduleServiceReferenceType
» ServiceReference,
        InteractivityDataReferenceList
» InteractivityDataReferences optional,
        ContentReferenceList ContentReferences
» optional,
        PreviewDataReferenceList
» PreviewDataReferences optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    type record ScheduleServiceReferenceType {
        AnyUri idRef,
        AnyUri audioLanguageIDRef optional,
        AnyUri textLanguageIDRef optional
    }

    type record InteractivityDataReferenceType {
        AnyUri idRef,
        AutoStartList AutoStarts optional,
        DistributionWindowList DistributionWindows
» optional
    }

    type set length (1 .. infinity) of
» InteractivityDataReferenceType InteractivityDataReferenceList;

    type record AutoStartType {
        XmlElementUnsignedIntValue text_
    }

```

```

    }

    type record TermsOfUseTextType {
        XmlElementStringValue text_
    }
}

group scheduleFragment {
    type record ScheduleType {
        AnyUri id,
        UInt version,
        boolean defaultSchedule optional,
        boolean onDemand optional,
        UInt validFrom optional,
        UInt validTo optional,
        ScheduleServiceReferenceType
» ServiceReference,
        InteractivityDataReferenceList
» InteractivityDataReferences optional,
        ContentReferenceList ContentReferences
» optional,
        PreviewDataReferenceList
» PreviewDataReferences optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    type record ScheduleServiceReferenceType {
        AnyUri idRef,
        AnyUri audioLanguageIDRef optional,
        AnyUri textLanguageIDRef optional
    }

    type record InteractivityDataReferenceType {
        AnyUri idRef,
        AutoStartList AutoStarts optional,
        DistributionWindowList DistributionWindows
» optional
    }

    type set length (1 .. infinity) of
» InteractivityDataReferenceType InteractivityDataReferenceList;

    type record AutoStartType {
        XmlElementUnsignedIntValue text_
    }

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

<pre> type set length (1 .. infinity) of AutoStartType » AutoStartList;  type record DistributionWindowType {     UInt startTime optional,     UInt endTime optional,     UInt duration optional,     UInt id optional }  type set length (1 .. infinity) of » DistributionWindowType DistributionWindowList;  type record ContentReferenceType {     AnyUri idRef, </pre>		<pre> type set length (1 .. infinity) of AutoStartType » AutoStartList;  type record DistributionWindowType {     UInt startTime optional,     UInt endTime optional,     UInt duration optional,     UInt id optional }  type set length (1 .. infinity) of » DistributionWindowType DistributionWindowList;  type record ContentReferenceType {     AnyUri idRef, </pre>
<pre>     AnyUri contentLocation optional, » // change 24 (WK 6): change necessary, based on OMA BCAS specification </pre>	<>	<pre>     AnyUri contentLocation, </pre>
<pre> AnyUri audioLanguageIDRef optional, AnyUri textLanguageIDRef optional, boolean repeatPlayback optional, AutoStartList AutoStarts optional, DistributionWindowList DistributionWindows » optional, PresentationWindowList PresentationWindows » optional }  type set length (1 .. infinity) of » ContentReferenceType ContentReferenceList;  /** @desc Used for expressing start and end time for » scheduled rendering of cachecast content type record PresentationWindowType {     UInt startTime optional,     UInt endTime optional,     UInt duration optional,     UInt id optional }  type set length (1 .. infinity) of » PresentationWindowType PresentationWindowList; }  group contentFragment {     type record ContentType { </pre>	=	<pre> AnyUri audioLanguageIDRef optional, AnyUri textLanguageIDRef optional, boolean repeatPlayback optional, AutoStartList AutoStarts optional, DistributionWindowList DistributionWindows » optional, PresentationWindowList PresentationWindows » optional }  type set length (1 .. infinity) of » ContentReferenceType ContentReferenceList;  /** @desc Used for expressing start and end time for » scheduled rendering of cachecast content type record PresentationWindowType {     UInt startTime optional,     UInt endTime optional,     UInt duration optional,     UInt id optional }  type set length (1 .. infinity) of » PresentationWindowType PresentationWindowList; }  group contentFragment {     type record ContentType { </pre>



Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        AnyUri id,
        UInt version,
        UInt validFrom optional,
        UInt validTo optional,
        AnyUri globalContentID optional,
        boolean emergency optional,
        //boolean serviceContentProtection optional,
        charstring baseCID optional,
        ContentServiceReferenceList ServiceReferences
    » optional,
        ProtectionKeyIDMinMaxList ProtectionKeyIDs
    » optional,
        // Start of Service guide
        // Service name list
        LanguageStringList Names,
        LanguageStringList Descriptions optional,
        // Applicable to scheduled rendering of
    » non-cachecast content
        StartTimeType StartTime optional,
        // Applicable to scheduled rendering of
    » non-cachecast content
        EndTimeType EndTime optional,
        AudioOrTextLanguageList AudioLanguages
    » optional,
        AudioOrTextLanguageList TextLanguages
    » optional,
        ParentalRatingList ParentalRatings optional,
        TargetUserProfileList TargetUserProfiles
    » optional,
        GenreList Genres optional,
        ExtensionList Extensions optional,
        // End of Service guide
        PreviewDataReferenceList
    » PreviewDataReferences optional,
        BroadcastAreaType BroadcastArea optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    type record ContentServiceReferenceType {
        AnyUri idRef,
        UInt16 weight optional
    }

    type set length (1 .. infinity) of
    » ContentServiceReferenceType ContentServiceReferenceList;

```

```

        AnyUri id,
        UInt version,
        UInt validFrom optional,
        UInt validTo optional,
        AnyUri globalContentID optional,
        boolean emergency optional,
        //boolean serviceContentProtection optional,
        charstring baseCID optional,
        ContentServiceReferenceList ServiceReferences
    » optional,
        ProtectionKeyIDMinMaxList ProtectionKeyIDs
    » optional,
        // Start of Service guide
        // Service name list
        LanguageStringList Names,
        LanguageStringList Descriptions optional,
        // Applicable to scheduled rendering of
    » non-cachecast content
        StartTimeType StartTime optional,
        // Applicable to scheduled rendering of
    » non-cachecast content
        EndTimeType EndTime optional,
        AudioOrTextLanguageList AudioLanguages
    » optional,
        AudioOrTextLanguageList TextLanguages
    » optional,
        ParentalRatingList ParentalRatings optional,
        TargetUserProfileList TargetUserProfiles
    » optional,
        GenreList Genres optional,
        ExtensionList Extensions optional,
        // End of Service guide
        PreviewDataReferenceList
    » PreviewDataReferences optional,
        BroadcastAreaType BroadcastArea optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    type record ContentServiceReferenceType {
        AnyUri idRef,
        UInt16 weight optional
    }

    type set length (1 .. infinity) of
    » ContentServiceReferenceType ContentServiceReferenceList;

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        type record StartTimeType {
            XmlElementDateTimeValue text_
        }

        type record EndTimeType {
            XmlElementDateTimeValue text_
        }
    }

    group accessFragment {
        type record AccessType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AccessTypeType AccessType,
            KeyManagementSystemList KeyManagementSystems
» optional,

            EncryptionTypeList EncryptionTypes optional,
            AccessTypeReferenceChoice choice optional,
            TerminalCapabilityRequirementType
» TerminalCapabilityRequirement optional,
            BandwidthRequirementType BandwidthRequirement
» optional,

            ServiceClassList ServiceClasses,
            PreviewDataReferenceList
» PreviewDataReferences optional,
            NotificationReceptionType
» NotificationReception optional,
            PrivateExtType PrivateExt optional
        }

        type record AccessTypeType {
            AccessTypeChoice choice
        }

        type union AccessTypeChoice {
            BroadcastServiceDeliveryType
» BroadcastServiceDelivery,
            UnicastServiceDeliveryList
» UnicastServiceDeliverys
        }

        type record BroadcastServiceDeliveryType {
            BDSTypeType BDSType optional,

```

```

        type record StartTimeType {
            XmlElementDateTimeValue text_
        }

        type record EndTimeType {
            XmlElementDateTimeValue text_
        }
    }

    group accessFragment {
        type record AccessType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AccessTypeType AccessType,
            KeyManagementSystemList KeyManagementSystems
» optional,

            EncryptionTypeList EncryptionTypes optional,
            AccessTypeReferenceChoice choice optional,
            TerminalCapabilityRequirementType
» TerminalCapabilityRequirement optional,
            BandwidthRequirementType BandwidthRequirement
» optional,

            ServiceClassList ServiceClasses,
            PreviewDataReferenceList
» PreviewDataReferences optional,
            NotificationReceptionType
» NotificationReception optional,
            PrivateExtType PrivateExt optional
        }

        type record AccessTypeType {
            AccessTypeChoice choice
        }

        type union AccessTypeChoice {
            BroadcastServiceDeliveryType
» BroadcastServiceDelivery,
            UnicastServiceDeliveryList
» UnicastServiceDeliverys
        }

        type record BroadcastServiceDeliveryType {
            BDSTypeType BDSType optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        SessionDescriptionType SessionDescription,
        /* field shall be present with ALC but shall
» not be present with FLUTE

        FileDescriptionType FileDescription optional
    }

    type record BDSTypeType {
        BDSTypeRangeType Type optional,
        VersionList Versions optional
    }

    type record VersionType {
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of VersionType

» VersionList;

    type record BDSTypeRangeType {
        XmlElementUnsignedByteValue text_
    }

    const UInt8 c_ipdcOverDvbH_bcType := 0;
    const UInt8 c_3gppMbms_bcType := 1;
    const UInt8 c_3gpp2Bcmcs_bcType := 2;

    /**
     *
     * @remark
     *   The presence of sdp and sdpRef fields is
» mutually
     *   exclusive!
     */
    type record SessionDescriptionType {
        /* inline SDP
        SessionDescriptionTypeChoice choice,
        SessionDescriptionReferenceType USBDRef

» optional,
        SessionDescriptionReferenceType ADPRef

» optional
    }

    /**
     *
     * @remark
     *   If both uri and idRef are present they must be

```

```

        SessionDescriptionType SessionDescription,
        /* field shall be present with ALC but shall
» not be present with FLUTE

        FileDescriptionType FileDescription optional
    }

    type record BDSTypeType {
        BDSTypeRangeType Type optional,
        VersionList Versions optional
    }

    type record VersionType {
        XmlElementStringValue text_
    }

    type set length (1 .. infinity) of VersionType

» VersionList;

    type record BDSTypeRangeType {
        XmlElementUnsignedByteValue text_
    }

    const UInt8 c_ipdcOverDvbH_bcType := 0;
    const UInt8 c_3gppMbms_bcType := 1;
    const UInt8 c_3gpp2Bcmcs_bcType := 2;

    /**
     *
     * @remark
     *   The presence of sdp and sdpRef fields is
» mutually
     *   exclusive!
     */
    type record SessionDescriptionType {
        /* inline SDP
        SessionDescriptionTypeChoice choice,
        SessionDescriptionReferenceType USBDRef

» optional,
        SessionDescriptionReferenceType ADPRef

» optional
    }

    /**
     *
     * @remark
     *   If both uri and idRef are present they must be

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

» identical

    */
    type record SessionDescriptionReferenceType {
        AnyUri uri optional,
        AnyUri idRef optional
    }

    type union SessionDescriptionTypeChoice {
        SDPType SDP,
        SessionDescriptionReferenceType SDPRef
    }

    type record SDPType {
        PreserveXmlSpace space,
        charstring encoding optional,
        XmlElementStringValue text_
    }

    type union AccessTypeReferenceChoice {
        IDRefTypeList ServiceReferences,
        ScheduleReferenceList ScheduleReferences
    }

    type record ScheduleReferenceType {
        AnyUri idRef,
        DistributionWindowIDList
    }
» DistributionWindowIDs optional
    }

    type set length (1 .. infinity) of
» ScheduleReferenceType ScheduleReferenceList;

    type record DistributionWindowIDType {
        XmlElementUnsignedIntValue text_
    }

    type set length (1 .. infinity) of
» DistributionWindowIDType DistributionWindowIDList;

    type record FileDescriptionType {
        charstring Content_Type optional,
        charstring Content_Encoding optional,
        UInt8 FEC_OTI_FEC_Encoding_ID optional,
        UInt32 FEC_OTI_FEC_Instance_ID optional,
        UInt32 FEC_OTI_Maximum_Source_Block_Length
    }
» optional,

```

```

» identical

    */
    type record SessionDescriptionReferenceType {
        AnyUri uri optional,
        AnyUri idRef optional
    }

    type union SessionDescriptionTypeChoice {
        SDPType SDP,
        SessionDescriptionReferenceType SDPRef
    }

    type record SDPType {
        PreserveXmlSpace space,
        charstring encoding optional,
        XmlElementStringValue text_
    }

    type union AccessTypeReferenceChoice {
        IDRefTypeList ServiceReferences,
        ScheduleReferenceList ScheduleReferences
    }

    type record ScheduleReferenceType {
        AnyUri idRef,
        DistributionWindowIDList
    }
» DistributionWindowIDs optional
    }

    type set length (1 .. infinity) of
» ScheduleReferenceType ScheduleReferenceList;

    type record DistributionWindowIDType {
        XmlElementUnsignedIntValue text_
    }

    type set length (1 .. infinity) of
» DistributionWindowIDType DistributionWindowIDList;

    type record FileDescriptionType {
        charstring Content_Type optional,
        charstring Content_Encoding optional,
        UInt8 FEC_OTI_FEC_Encoding_ID optional,
        UInt32 FEC_OTI_FEC_Instance_ID optional,
        UInt32 FEC_OTI_Maximum_Source_Block_Length
    }
» optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        UInt32 FEC_OTI_Encoding_Symbol_Length
» optional,
        UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols
» optional,
        charstring FEC_OTI_Scheme_Specific_Info
» optional,
        FileList Files
    } with {variant "replace '_' with '-'" }

    /**
     *
     * @desc
     *   More info on parameters available in RFC 3926
» 3.4.2
     */
    type record FileType {
        AnyUri Content_Location,
        UInt TOI,
        UInt32 Content_Length optional,
        UInt32 Transfer_Length optional,
        charstring Content_Type optional,
        charstring Content_Encoding optional,
        charstring Content_MD5 optional,
        UInt8 FEC_OTI_FEC_Encoding_ID optional,
        UInt32 FEC_OTI_FEC_Instance_ID optional,
        UInt32 FEC_OTI_Maximum_Source_Block_Length

» optional,
        UInt32 FEC_OTI_Encoding_Symbol_Length
» optional,
        UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols
» optional,
        charstring FEC_OTI_Scheme_Specific_Info
» optional
    } with {variant "replace '_' with '-'" }

    type set length (1 .. infinity) of FileType FileList;

    type record UnicastServiceDeliveryType {
        UInt8 type_,
        AccessServerURLList AccessServerURLs,
        SessionDescriptionType SessionDescription
» optional,
        ServiceAccessNotificationURLList
» ServiceAccessNotificationURLs optional
    }

```

```

        UInt32 FEC_OTI_Encoding_Symbol_Length
» optional,
        UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols
» optional,
        charstring FEC_OTI_Scheme_Specific_Info
» optional,
        FileList Files
    } with {variant "replace '_' with '-'" }

    /**
     *
     * @desc
     *   More info on parameters available in RFC 3926
» 3.4.2
     */
    type record FileType {
        AnyUri Content_Location,
        UInt TOI,
        UInt32 Content_Length optional,
        UInt32 Transfer_Length optional,
        charstring Content_Type optional,
        charstring Content_Encoding optional,
        charstring Content_MD5 optional,
        UInt8 FEC_OTI_FEC_Encoding_ID optional,
        UInt32 FEC_OTI_FEC_Instance_ID optional,
        UInt32 FEC_OTI_Maximum_Source_Block_Length

» optional,
        UInt32 FEC_OTI_Encoding_Symbol_Length
» optional,
        UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols
» optional,
        charstring FEC_OTI_Scheme_Specific_Info
» optional
    } with {variant "replace '_' with '-'" }

    type set length (1 .. infinity) of FileType FileList;

    type record UnicastServiceDeliveryType {
        UInt8 type_,
        AccessServerURLList AccessServerURLs,
        SessionDescriptionType SessionDescription
» optional,
        ServiceAccessNotificationURLList
» ServiceAccessNotificationURLs optional
    }

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        type set length (1 .. infinity) of
» UnicastServiceDeliveryType UnicastServiceDeliveryList;

        type record AccessServerURLType {
            XmlElementAnyURIValue text_
        }

        type set length (1 .. infinity) of
» AccessServerURLType AccessServerURLList;

        type record ServiceAccessNotificationURLType {
            XmlElementAnyURIValue text_
        }

        type set length (1 .. infinity) of
» ServiceAccessNotificationURLType ServiceAccessNotificationURLList;

        const UInt8 c_http_ucType := 0;
        const UInt8 c_wap10_ucType := 1;
        const UInt8 c_wap2x_ucType := 2;
        const UInt8 c_rtspRtp_ucType := 3;
        const UInt8 c_rtspRtp3gppPss_ucType := 4;
        const UInt8 c_rtspRtp3gpp2Mss_ucType := 5;
        const UInt8 c_fluteUnicast_ucType := 6;

        type record KeyManagementSystemType {
            UInt8 kmsType,
            UInt8 protectionType,
            PermissionsIssuerURIType
» PermissionsIssuerURI,
            ProtectionKeyIDList ProtectionKeyIDs
» optional,
            TerminalBindingKeyIDType TerminalBindingKeyID
» optional
        }

        type set length (1 .. infinity) of
» KeyManagementSystemType KeyManagementSystemList;

        type record PermissionsIssuerURIType {
            //boolean type_,
            XmlElementAnyURIValue text_
        }

        type record TerminalBindingKeyIDType {
            AnyUri tbkPermissionsIssuerURI optional,

```

```

        type set length (1 .. infinity) of
» UnicastServiceDeliveryType UnicastServiceDeliveryList;

        type record AccessServerURLType {
            XmlElementAnyURIValue text_
        }

        type set length (1 .. infinity) of
» AccessServerURLType AccessServerURLList;

        type record ServiceAccessNotificationURLType {
            XmlElementAnyURIValue text_
        }

        type set length (1 .. infinity) of
» ServiceAccessNotificationURLType ServiceAccessNotificationURLList;

        const UInt8 c_http_ucType := 0;
        const UInt8 c_wap10_ucType := 1;
        const UInt8 c_wap2x_ucType := 2;
        const UInt8 c_rtspRtp_ucType := 3;
        const UInt8 c_rtspRtp3gppPss_ucType := 4;
        const UInt8 c_rtspRtp3gpp2Mss_ucType := 5;
        const UInt8 c_fluteUnicast_ucType := 6;

        type record KeyManagementSystemType {
            UInt8 kmsType,
            UInt8 protectionType,
            PermissionsIssuerURIType
» PermissionsIssuerURI,
            ProtectionKeyIDList ProtectionKeyIDs
» optional,
            TerminalBindingKeyIDType TerminalBindingKeyID
» optional
        }

        type set length (1 .. infinity) of
» KeyManagementSystemType KeyManagementSystemList;

        type record PermissionsIssuerURIType {
            //boolean type_,
            XmlElementAnyURIValue text_
        }

        type record TerminalBindingKeyIDType {
            AnyUri tbkPermissionsIssuerURI optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

<pre> XmlElementUnsignedIntValue text_ }  type record EncryptionTypeType {     XmlElementUnsignedByteValue text_ (0..3) }  type set length (1 .. infinity) of EncryptionTypeType » EncryptionTypeList;  const UInt8 c_ipsec_encryptType := 0; const UInt8 c_strp_encryptType := 1; const UInt8 c_ismaCryp_encryptType := 2; const UInt8 c_dcf_encryptType := 3;  type record IDRefType {     AnyUri idRef }  type set length (1 .. infinity) of IDRefType » IDRefTypeList;  type record TerminalCapabilityRequirementType {     VideoType Video optional,     AudioType Audio optional,     DownloadFileType DownloadFile optional }  type record VideoType {     VideoComplexityType Complexity }  type record VideoComplexityType {     BitrateType Bitrate optional,     ResolutionType Resolution optional,     MinimumBufferSizeType MinimumBufferSize » optional }  type record BitrateType { </pre>		<pre> XmlElementUnsignedIntValue text_ }  type record EncryptionTypeType {     XmlElementUnsignedByteValue text_ (0..3) }  type set length (1 .. infinity) of EncryptionTypeType » EncryptionTypeList;  const UInt8 c_ipsec_encryptType := 0; const UInt8 c_strp_encryptType := 1; const UInt8 c_ismaCryp_encryptType := 2; const UInt8 c_dcf_encryptType := 3;  type record IDRefType {     AnyUri idRef }  type set length (1 .. infinity) of IDRefType » IDRefTypeList;  type record TerminalCapabilityRequirementType {     VideoType Video optional,     AudioType Audio optional,     DownloadFileType DownloadFile optional }  type record VideoType {     VideoComplexityType Complexity }  type record VideoComplexityType {     BitrateType Bitrate optional,     ResolutionType Resolution optional,     MinimumBufferSizeType MinimumBufferSize » optional }  type record BitrateType { </pre>
<pre>         UInt16 average optional, // » change 24 (WK 6): change necessary, based on OMA BCAST specification         UInt16 maximum optional // » change 24 (WK 6): change necessary, based on OMA BCAST specification     } </pre>	<>	<pre>         UInt8 average optional,         UInt8 maximum optional     } </pre>
}	=	}

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

<pre> type record ResolutionType {     UInt16 horizontal,     » change 24 (WK 6): change necessary, based on OMA BCAST specification     UInt16 vertical,     » change 24 (WK 6): change necessary, based on OMA BCAST specification     float temporal }  type record MinimumBufferSizeType {     XmlElementUnsignedIntValue text_ }  type record AudioType {     AudioComplexityType Complexity }  type record AudioComplexityType {     BitrateType Bitrate optional,     MinimumBufferSizeType MinimumBufferSize » optional }  type record DownloadFileType {     MIMETYPEList MIMETypes }  type record MIMETYPEType {     charstring codec optional,     XmlElementStringValue text_ }  type set length (1 .. infinity) of MIMETYPEType » MIMETYPEList;  type record BandwidthRequirementType {     XmlElementUnsignedIntValue text_ }  /* * Note: TS SG 20071220 proposes a complexType for » ServiceClass which is not XML-compliant * Until this is resolved in BCAST group, » ServiceClass datatype is not changed in schema */ type record ServiceClassType {     XmlElementStringValue text_ </pre>	<=>	<pre> type record ResolutionType {     UInt8 horizontal,     UInt8 vertical,     float temporal }  type record MinimumBufferSizeType {     XmlElementUnsignedIntValue text_ }  type record AudioType {     AudioComplexityType Complexity }  type record AudioComplexityType {     BitrateType Bitrate optional,     MinimumBufferSizeType MinimumBufferSize » optional }  type record DownloadFileType {     MIMETYPEList MIMETypes }  type record MIMETYPEType {     charstring codec optional,     XmlElementStringValue text_ }  type set length (1 .. infinity) of MIMETYPEType » MIMETYPEList;  type record BandwidthRequirementType {     XmlElementUnsignedIntValue text_ }  /* * Note: TS SG 20071220 proposes a complexType for » ServiceClass which is not XML-compliant * Until this is resolved in BCAST group, » ServiceClass datatype is not changed in schema */ type record ServiceClassType {     XmlElementStringValue text_ </pre>
--	-----	--



Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

    }

    type set length (1 .. infinity) of ServiceClassType
» ServiceClassList;

    }

    group purchaseItemFragment {
        type record PurchaseItemType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AnyUri globalPurchaseItemID,
            UInt binaryPurchaseItemID optional,
            UInt16 weight optional,
            boolean closed optional,
            PurchaseItemChoice choice,
            ProtectionKeyIDList ProtectionKeyIDs

» optional,

            LanguageStringList Names,
            LanguageStringList Descriptions optional,
            StartTimeType StartTime optional,
            EndTimeType EndTime optional,
            ParentalRatingList ParentalRatings optional,
            ExtensionList Extensions optional,
            IDRefTypeList DependencyReferences optional,
            IDRefTypeList ExclusionReferences optional,
            PrivateExtType PrivateExt optional

        }

        type union PurchaseItemChoice {
            IDRefTypeList ServiceReferences,
            ScheduleReferenceList ScheduleReferences,
            IDRefTypeList ContentReferences,
            IDRefTypeList PurchaseItemReferences

        }

    }

    group purchaseDataFragment {
        type record PurchaseDataType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            LanguageStringList Descriptions optional,

```

```

    }

    type set length (1 .. infinity) of ServiceClassType
» ServiceClassList;

    }

    group purchaseItemFragment {
        type record PurchaseItemType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AnyUri globalPurchaseItemID,
            UInt binaryPurchaseItemID optional,
            UInt16 weight optional,
            boolean closed optional,
            PurchaseItemChoice choice,
            ProtectionKeyIDList ProtectionKeyIDs

» optional,

            LanguageStringList Names,
            LanguageStringList Descriptions optional,
            StartTimeType StartTime optional,
            EndTimeType EndTime optional,
            ParentalRatingList ParentalRatings optional,
            ExtensionList Extensions optional,
            IDRefTypeList DependencyReferences optional,
            IDRefTypeList ExclusionReferences optional,
            PrivateExtType PrivateExt optional

        }

        type union PurchaseItemChoice {
            IDRefTypeList ServiceReferences,
            ScheduleReferenceList ScheduleReferences,
            IDRefTypeList ContentReferences,
            IDRefTypeList PurchaseItemReferences

        }

    }

    group purchaseDataFragment {
        type record PurchaseDataType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            LanguageStringList Descriptions optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        PriceInfoType PriceInfo optional,
        PromotionInfoList PromotionInfos optional,
        ExtensionList Extensions optional,
        IDRefType PurchaseItemReference,
        IDRefTypeList PurchaseChannelReferences,
        PreviewDataReferenceList
    » PreviewDataReferences optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    group PurchaseDataUsage {
        const UInt8 c_PurchaseDataUsage_unspecified
    » := 0;
        const UInt8
    » c_PurchaseDataUsage_ServiceByServiceSwitching := 1;
        const UInt8
    » c_PurchaseDataUsage_ServiceGuideBrowsing := 2;
        const UInt8
    » c_PurchaseDataUsage_ServiceGuidePreview := 3;
        const UInt8 c_PurchaseDataUsage_Barker := 4;
        const UInt8
    » c_PurchaseDataUsage_AlternativToBlackout := 5;
    }

    group subscriptionTypes {
        const UInt c_openTimeSubscription := 0;
        const UInt c_openEndedSubscription := 1;
    }

    type record PriceInfoType {
        UInt subscriptionType,
        PriceList MonetaryPrices optional,
        TotalNumberTokenType TotalNumberToken
    » optional,
        SubscriptionPeriodType SubscriptionPeriod
    » optional
    }

    type record SubscriptionPeriodType {
        XmlElementStringValue text_
    }

    type record PromotionInfoType {
        AnyUri id,
        UInt validFrom optional,

```

```

        PriceInfoType PriceInfo optional,
        PromotionInfoList PromotionInfos optional,
        ExtensionList Extensions optional,
        IDRefType PurchaseItemReference,
        IDRefTypeList PurchaseChannelReferences,
        PreviewDataReferenceList
    » PreviewDataReferences optional,
        TermsOfUseList TermsOfUses optional,
        PrivateExtType PrivateExt optional
    }

    group PurchaseDataUsage {
        const UInt8 c_PurchaseDataUsage_unspecified
    » := 0;
        const UInt8
    » c_PurchaseDataUsage_ServiceByServiceSwitching := 1;
        const UInt8
    » c_PurchaseDataUsage_ServiceGuideBrowsing := 2;
        const UInt8
    » c_PurchaseDataUsage_ServiceGuidePreview := 3;
        const UInt8 c_PurchaseDataUsage_Barker := 4;
        const UInt8
    » c_PurchaseDataUsage_AlternativToBlackout := 5;
    }

    group subscriptionTypes {
        const UInt c_openTimeSubscription := 0;
        const UInt c_openEndedSubscription := 1;
    }

    type record PriceInfoType {
        UInt subscriptionType,
        PriceList MonetaryPrices optional,
        TotalNumberTokenType TotalNumberToken
    » optional,
        SubscriptionPeriodType SubscriptionPeriod
    » optional
    }

    type record SubscriptionPeriodType {
        XmlElementStringValue text_
    }

    type record PromotionInfoType {
        AnyUri id,
        UInt validFrom optional,

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        UInt validTo optional,
        LanguageStringList Titles,
        TargetUserProfileList TargetUserProfiles
» optional,

        LanguageStringList Descriptions optional,
        PromotionExtensionURLList PromotionExtensions
» optional
    }

    type set length (1 .. infinity) of PromotionInfoType
» PromotionInfoList;

    type record PromotionExtensionURLType {
        AnyUri url,
        LanguageStringList Descriptions optional
    }

    type set length (1 .. infinity) of
» PromotionExtensionURLType PromotionExtensionURLList;

    group tokenTypes {
        const UInt8 c_tokenType_unspecified := 0;
        const UInt8 c_tokenType_tokensForDrmProfile
» := 1;

        const UInt8
» c_tokenType_timeTokensForServicePurseSmartcard := 2;
        const UInt8
» c_tokenType_timeTokensForUserPurseSmartcard := 3;
        const UInt8
» c_tokenType_playTokensForServicePurseSmartcard := 4;
        const UInt8
» c_tokenType_playTokensForUserPurseSmartcard := 5;
    }

    group consumptionUnits {
        // used for token type 2 or 3
        const UInt8 c_consumptionUnit_timeInSeconds
» := 0;

        // used for token type 2 or 3
        const UInt8 c_consumptionUnit_timeInMinutes
» := 1;

        // used for token type 2 or 3
        const UInt8 c_consumptionUnit_timeInHours :=
» 2;

        // used for token type 4 or 5
        const UInt8 c_consumptionUnit_numberOfPlays

```

```

        UInt validTo optional,
        LanguageStringList Titles,
        TargetUserProfileList TargetUserProfiles
» optional,

        LanguageStringList Descriptions optional,
        PromotionExtensionURLList PromotionExtensions
» optional
    }

    type set length (1 .. infinity) of PromotionInfoType
» PromotionInfoList;

    type record PromotionExtensionURLType {
        AnyUri url,
        LanguageStringList Descriptions optional
    }

    type set length (1 .. infinity) of
» PromotionExtensionURLType PromotionExtensionURLList;

    group tokenTypes {
        const UInt8 c_tokenType_unspecified := 0;
        const UInt8 c_tokenType_tokensForDrmProfile
» := 1;

        const UInt8
» c_tokenType_timeTokensForServicePurseSmartcard := 2;
        const UInt8
» c_tokenType_timeTokensForUserPurseSmartcard := 3;
        const UInt8
» c_tokenType_playTokensForServicePurseSmartcard := 4;
        const UInt8
» c_tokenType_playTokensForUserPurseSmartcard := 5;
    }

    group consumptionUnits {
        // used for token type 2 or 3
        const UInt8 c_consumptionUnit_timeInSeconds
» := 0;

        // used for token type 2 or 3
        const UInt8 c_consumptionUnit_timeInMinutes
» := 1;

        // used for token type 2 or 3
        const UInt8 c_consumptionUnit_timeInHours :=
» 2;

        // used for token type 4 or 5
        const UInt8 c_consumptionUnit_numberOfPlays

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

» := 0;

    }

    type record TotalNumberTokenType {
        UInt8 tokenType,
        UInt16 consumptionAmount optional,
        UInt8 consumptionUnit,
        UInt16 maxReplay optional,
        XmlElementUnsignedShortValue text_
    }

    type record PriceType {
        charstring currency,
        XmlElementDecimalValue text_
    }

    type set length (1 .. infinity) of PriceType

» PriceList;
    }

    group purchaseChannelFragment {
        type record PurchaseChannelType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AnyUri rightsIssuerURI optional,
            LanguageStringList Names,
            LanguageStringList Descriptions optional,
            ContactInfoType ContactInfo optional,
            PortalURLType PortalURL optional,
            PurchaseURLList PurchaseURLs optional ,
            ExtensionList Extensions optional,
            PrivateExtType PrivateExt optional
        }

        type record ContactInfoType {
            XmlElementStringValue text_
        }

        type record PortalURLType {
            UInt8 supportedService optional,
            XmlElementAnyURIValue text_
        }

        type record PurchaseURLType {

```

```

» := 0;

    }

    type record TotalNumberTokenType {
        UInt8 tokenType,
        UInt16 consumptionAmount optional,
        UInt8 consumptionUnit,
        UInt16 maxReplay optional,
        XmlElementUnsignedShortValue text_
    }

    type record PriceType {
        charstring currency,
        XmlElementDecimalValue text_
    }

    type set length (1 .. infinity) of PriceType

» PriceList;
    }

    group purchaseChannelFragment {
        type record PurchaseChannelType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            AnyUri rightsIssuerURI optional,
            LanguageStringList Names,
            LanguageStringList Descriptions optional,
            ContactInfoType ContactInfo optional,
            PortalURLType PortalURL optional,
            PurchaseURLList PurchaseURLs optional ,
            ExtensionList Extensions optional,
            PrivateExtType PrivateExt optional
        }

        type record ContactInfoType {
            XmlElementStringValue text_
        }

        type record PortalURLType {
            UInt8 supportedService optional,
            XmlElementAnyURIValue text_
        }

        type record PurchaseURLType {

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

        XmlElementAnyURIValue text_
    }

    type set length (1 .. infinity) of PurchaseURLType
» PurchaseURLList;
    }

    group previewDataFragment {
        type record PreviewDataType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            SMILType SMIL optional,
            VideoPType Video optional,
            AudioPType Audio optional,
            PictureType Picture optional,
            EncodedLanguageStringList Texts optional,
            AccessReferenceType AccessReference optional,
            PrivateExtType PrivateExt optional
        }

        type record SMILType {
            UInt8 type_ optional,
            charstring encoding optional,
            // could be further structured
            XmlElementStringValue text_
        }

        type record VideoPType {
            VideoURIType VideoURI,
            MIMETYPEType MIMETYPE optional,
            LanguageStringList AlternativeTexts optional,
            PictureType AlternativePicture optional
        }

        type record VideoURIType {
            XmlElementAnyURIValue text_
        }

        type record AudioPType {
            AudioURIType AudioURI,
            MIMETYPEType MIMETYPE optional,
            LanguageStringList AlternativeTexts optional,
            PictureType AlternativePicture optional
        }
    }

```

```

        XmlElementAnyURIValue text_
    }

    type set length (1 .. infinity) of PurchaseURLType
» PurchaseURLList;
    }

    group previewDataFragment {
        type record PreviewDataType {
            AnyUri id,
            UInt version,
            UInt validFrom optional,
            UInt validTo optional,
            SMILType SMIL optional,
            VideoPType Video optional,
            AudioPType Audio optional,
            PictureType Picture optional,
            EncodedLanguageStringList Texts optional,
            AccessReferenceType AccessReference optional,
            PrivateExtType PrivateExt optional
        }

        type record SMILType {
            UInt8 type_ optional,
            charstring encoding optional,
            // could be further structured
            XmlElementStringValue text_
        }

        type record VideoPType {
            VideoURIType VideoURI,
            MIMETYPEType MIMETYPE optional,
            LanguageStringList AlternativeTexts optional,
            PictureType AlternativePicture optional
        }

        type record VideoURIType {
            XmlElementAnyURIValue text_
        }

        type record AudioPType {
            AudioURIType AudioURI,
            MIMETYPEType MIMETYPE optional,
            LanguageStringList AlternativeTexts optional,
            PictureType AlternativePicture optional
        }
    }

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```

    type record AudioURIType {
        XmlElementAnyURIValue text_
    }

    type record PictureType {
        PictureChoice choice,
        MIMETYPEType MIMETYPE optional,
        LanguageStringList AlternativeTexts optional
    }

    type union PictureChoice {
        PictureURIType PictureURI,
        PictureDataType PictureData
    }

    type record PictureURIType {
        XmlElementAnyURIValue text_
    }

    type record PictureDataType {
        XmlElementStringValue text_
    }

    type charstring Text;

    type record AccessReferenceType {
        AnyUri idRef,
        UInt8 usage
    }

}

group interactivityDataFragment {
    type record InteractivityDataType {
        AnyUri id,
        UInt version,
        UInt validFrom optional,
        UInt validTo optional,
        boolean preListenIndicator,
        AnyUri interactivityMediaDocumentPointer,
        LanguageStringList InteractivityTypes

» optional,

        IDRefType ServiceReference,
        InteractivityDataChoice choice optional,
        InteractivityDeliveryType

```

```

    type record AudioURIType {
        XmlElementAnyURIValue text_
    }

    type record PictureType {
        PictureChoice choice,
        MIMETYPEType MIMETYPE optional,
        LanguageStringList AlternativeTexts optional
    }

    type union PictureChoice {
        PictureURIType PictureURI,
        PictureDataType PictureData
    }

    type record PictureURIType {
        XmlElementAnyURIValue text_
    }

    type record PictureDataType {
        XmlElementStringValue text_
    }

    type charstring Text;

    type record AccessReferenceType {
        AnyUri idRef,
        UInt8 usage
    }

}

group interactivityDataFragment {
    type record InteractivityDataType {
        AnyUri id,
        UInt version,
        UInt validFrom optional,
        UInt validTo optional,
        boolean preListenIndicator,
        AnyUri interactivityMediaDocumentPointer,
        LanguageStringList InteractivityTypes

» optional,

        IDRefType ServiceReference,
        InteractivityDataChoice choice optional,
        InteractivityDeliveryType

```

Datei: LibBCast\_ServiceGuide\_TypesAndValues.ttcn  
(Fortsetzung)

```
» InteractivityDelivery optional,
    ExtensionType Extension optional,
    BackOffTimingType BackOffTiming optional,
    TermsOfUseList TermsOfUses optional,
    PrivateExtType PrivateExt optional
}

type union InteractivityDataChoice {
    IDRefTypeList ContentReferences,
    ScheduleReferenceList ScheduleReferences,
    InteractivityWindowList InteractivityWindows
}

type record BackOffTimingType {
    float offsetTime,
    float randomTime
}

type record InteractivityDeliveryType {
    AnyUri interactivityMediaURL optional,
    boolean pushDelivery optional
}

type record InteractivityWindowType {
    UInt startTime,
    UInt endTime
}

type set length (1 .. infinity) of
» InteractivityWindowType InteractivityWindowList;
}
} // end group xmlFragments
} // end module LibBCast_ServiceGuide_TypesAndValues
```

```
» InteractivityDelivery optional,
    ExtensionType Extension optional,
    BackOffTimingType BackOffTiming optional,
    TermsOfUseList TermsOfUses optional,
    PrivateExtType PrivateExt optional
}

type union InteractivityDataChoice {
    IDRefTypeList ContentReferences,
    ScheduleReferenceList ScheduleReferences,
    InteractivityWindowList InteractivityWindows
}

type record BackOffTimingType {
    float offsetTime,
    float randomTime
}

type record InteractivityDeliveryType {
    AnyUri interactivityMediaURL optional,
    boolean pushDelivery optional
}

type record InteractivityWindowType {
    UInt startTime,
    UInt endTime
}

type set length (1 .. infinity) of
» InteractivityWindowType InteractivityWindowList;
}
} // end group xmlFragments
} // end module LibBCast_ServiceGuide_TypesAndValues
```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn

```
/**
 *
 * @author
 *   ETSI CTI OMA BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies message types and values that are used to
 *   sconfigure the transport of BCAST messages. There are no requirements
 *   regarding the encoding of these primitives.
```

```
= /**
 *
 * @author
 *   ETSI CTI OMA BCAST CON Project
 * @version
 *   $Id$
 * @desc
 *   This module specifies message types and values that are used to
 *   sconfigure the transport of BCAST messages. There are no requirements
 *   regarding the encoding of these primitives.
```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn

(Fortsetzung)

<pre> * @remark *   End users should be aware that any changes made to this file may *   negatively affect the interworking of test suite code with the BCast *   SUT Adapter during test execution. */ module LibBCast_ServicePrimitives_TypesAndValues {   import from LibBCast_FileStreamDistribution_TypesAndValues all;   import from LibBCast_ServiceGuide_TypesAndValues all;   import from LibCommon_BasicTypesAndValues all;   import from LibBCast_FileStreamDistribution_TypesAndValues all;   import from LibBCast_Common_TypesAndValues all;   import from LibCommon_TextStrings all; </pre>	<>	<pre> * @remark *   End users should be aware that any changes made to this file may *   negatively affect the interworking of test suite code with the BCast *   SUT Adapter during test execution. */ module LibBCast_ServicePrimitives_TypesAndValues {   import from LibBCast_FileStreamDistribution_TypesAndValues all;   import from LibBCast_ServiceGuide_TypesAndValues all;   import from LibCommon_BasicTypesAndValues all;   import from LibBCast_FileStreamDistribution_TypesAndValues all;   import from LibBCast_Common_TypesAndValues all; </pre>
<pre> /**  *  * @desc  *   This group specifies information to be forwarded from TTCN-3 » to  *   the FLUTE application  */ group fluteConfigPrimitives {    /**    *    * @desc    *   Send to the SA to request the start of a flute session    * @member    *   fluteSessionId The flute session identifier    * @member    *   FDT_Instance The used FDT    * @member    *   fluteSessionPayload Includes a SGDU or SGDD    */   type record StartFluteSessionRequest {     UInt FluteSessionId, </pre>	=	<pre> /**  *  * @desc  *   This group specifies information to be forwarded from TTCN-3 » to  *   the FLUTE application  */ group fluteConfigPrimitives {    /**    *    * @desc    *   Send to the SA to request the start of a flute session    * @member    *   fluteSessionId The flute session identifier    * @member    *   FDT_Instance The used FDT    * @member    *   fluteSessionPayload Includes a SGDU or SGDD    */   type record StartFluteSessionRequest {     UInt FluteSessionId, </pre>
<pre>       charstring FluteSessionIp, // » change 22 (WK 6): further structured TTCN-3 types       IPAddressType FluteSessionIpType, // » change 22 (WK 6): further structured TTCN-3 types       UInt16 FluteSessionPort, // » change 22 (WK 6): further structured TTCN-3 types </pre>	+ -	
<pre>       FDT_InstanceType FDT_Instance,       FluteSessionPayloadType FluteSessionPayload     }      type union FluteSessionPayloadType { </pre>	=	<pre>       FDT_InstanceType FDT_Instance,       FluteSessionPayloadType FluteSessionPayload     }      type union FluteSessionPayloadType { </pre>



Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

<pre> » LibBCast_ServicePrimitives_TypesAndValues.ServiceGuideDeliveryUnitList » ServiceGuideDeliveryUnits, </pre>		<pre> » LibBCast_ServicePrimitives_TypesAndValues.ServiceGuideDeliveryUnitList » ServiceGuideDeliveryUnits, </pre>
<pre> ServiceGuideDeliveryDescriptorList » ServiceGuideDeliveryDescriptors,     DvbhEsgDescriptor ESGDescriptor // change » 13 (WK 6): send DVB-H bootstrap files via FLUTE </pre>	<>	<pre> ServiceGuideDeliveryDescriptorList » ServiceGuideDeliveryDescriptors </pre>
<pre> }  type set of InteractivityMediaDocumentType » InteractivityMediaDocumentList;  type set of ServiceGuideDeliveryUnit » ServiceGuideDeliveryUnitList;  //type set length(1 .. infinity) of » ServiceGuideDeliveryDescriptor ServiceGuideDeliveryDescriptorList;  type record FDT_InstanceType {     FileList Files,     MBMS_Session_Identity_Expiry_List MBMS_Session_Identity_Expirys » optional,     charstring Expires,     boolean Complete optional,     charstring Content_Type optional,     charstring Content_Encoding optional,     UInt8 FEC_OTI_FEC_Encoding_ID optional,     UInt32 FEC_OTI_FEC_Instance_ID optional,     UInt32 FEC_OTI_Maximum_Source_Block_Length optional,     UInt32 FEC_OTI_Encoding_Symbol_Length optional,     UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,     charstring FEC_OTI_Scheme_Specific_Info optional,     boolean FullFDT optional,     UInt32 Version_ID_Length optional </pre>	=	<pre> }  type set of InteractivityMediaDocumentType » InteractivityMediaDocumentList;  type set of ServiceGuideDeliveryUnit » ServiceGuideDeliveryUnitList;  //type set length(1 .. infinity) of » ServiceGuideDeliveryDescriptor ServiceGuideDeliveryDescriptorList;  type record FDT_InstanceType {     FileList Files,     MBMS_Session_Identity_Expiry_List MBMS_Session_Identity_Expirys » optional,     charstring Expires,     boolean Complete optional,     charstring Content_Type optional,     charstring Content_Encoding optional,     UInt8 FEC_OTI_FEC_Encoding_ID optional,     UInt32 FEC_OTI_FEC_Instance_ID optional,     UInt32 FEC_OTI_Maximum_Source_Block_Length optional,     UInt32 FEC_OTI_Encoding_Symbol_Length optional,     UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,     charstring FEC_OTI_Scheme_Specific_Info optional,     boolean FullFDT optional,     UInt32 Version_ID_Length optional </pre>
<pre> } with { // change 25 (WK 6): variant extension for definitio » ns of XML namespaces     variant "replace '_' with '-' &amp;&amp; namespacedef » nsprefix = '' nsuri = 'urn:oma:xml:bcast:fd:fdt:1.0' nsprefix = 'xsi' nsuri » = 'http://www.w3.org/2001/XMLSchema-instance'     } </pre>	<>	<pre> } with {variant "replace '_' with '-' } </pre>
<pre> type set length (1 .. infinity) of FileType FileList;  type record FileType { </pre>	=	<pre> type set length (1 .. infinity) of FileType FileList;  type record FileType { </pre>

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

        MBMS_Session_Identity_List MBMS_Session_Identity optional,
        AnyUri Content_Location,
        UInt TOI,
        UInt32 Content_Length optional,
        UInt32 Transfer_Length optional,
        charstring Content_Type optional,
        charstring Content_Encoding optional,
        charstring Content_MD5 optional,
        UInt8 FEC_OTI_FEC_Encoding_ID optional,
        UInt32 FEC_OTI_FEC_Instance_ID optional,
        UInt32 FEC_OTI_Maximum_Source_Block_Length optional,
        UInt32 FEC_OTI_Encoding_Symbol_Length optional,
        UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,
        charstring FEC_OTI_Scheme_Specific_Info optional,
        UInt32 Version_ID_Length optional
    } with {variant "replace '_' with '-'"}

    type set length (1 .. infinity) of
» MBMS_Session_Identity_Expiry_Type MBMS_Session_Identity_Expiry_List;

    type set length (1 .. infinity) of MBMS_Session_Identity_Type
» MBMS_Session_Identity_List;

    type record MBMS_Session_Identity_Expiry_Type {
        UInt8 text_,
        UInt32 value_
    }

    type record MBMS_Session_Identity_Type {
        UInt8 text_
    }

    /**
     *
     * @desc
     *     Send by the SA to indicate that a start flute session
» request
     *
     *     could be send.
     * @member
     *     returnCode 0 - success 1+ - error code
     * @member
     *     reason should be set if return code > 0
     */
    type record StartFluteSessionResponse {
        UInt returnCode,
        charstring reason optional

```

```

        MBMS_Session_Identity_List MBMS_Session_Identity optional,
        AnyUri Content_Location,
        UInt TOI,
        UInt32 Content_Length optional,
        UInt32 Transfer_Length optional,
        charstring Content_Type optional,
        charstring Content_Encoding optional,
        charstring Content_MD5 optional,
        UInt8 FEC_OTI_FEC_Encoding_ID optional,
        UInt32 FEC_OTI_FEC_Instance_ID optional,
        UInt32 FEC_OTI_Maximum_Source_Block_Length optional,
        UInt32 FEC_OTI_Encoding_Symbol_Length optional,
        UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,
        charstring FEC_OTI_Scheme_Specific_Info optional,
        UInt32 Version_ID_Length optional
    } with {variant "replace '_' with '-'"}

    type set length (1 .. infinity) of
» MBMS_Session_Identity_Expiry_Type MBMS_Session_Identity_Expiry_List;

    type set length (1 .. infinity) of MBMS_Session_Identity_Type
» MBMS_Session_Identity_List;

    type record MBMS_Session_Identity_Expiry_Type {
        UInt8 text_,
        UInt32 value_
    }

    type record MBMS_Session_Identity_Type {
        UInt8 text_
    }

    /**
     *
     * @desc
     *     Send by the SA to indicate that a start flute session
» request
     *
     *     could be send.
     * @member
     *     returnCode 0 - success 1+ - error code
     * @member
     *     reason should be set if return code > 0
     */
    type record StartFluteSessionResponse {
        UInt returnCode,
        charstring reason optional

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

<pre>     }   } with {     encode "FLUTECodec"   } // End fluteConfigPrimitives    /**    *    * @desc    *   These primitives extract information from or provide it to the    *   upper interface of a HTTP stack    */   group httpConfigPrimitives {      /**      *      * @desc      *   The http request which is send by the end user to » subscribe      *   to a service.      * @member      *   connectionId Contains a unique identifier for the » (TCP)      *   connection on which the POST was received      * @member      *   serviceName Contains the name of the service which the » end      *   user want to subscribe.      */     type record HttpSubscriptionRequestIndication {       UInt connectionId,       charstring serviceName     }      /**      * Note: Modifications to types still likely to cover HTTP » use in      * service provisioning      *      * @desc      *   Used for terminal requests on interactive channel      * @member      *   connectionId Contains a unique identifier for the » (TCP)      *   connection on which the POST was received      * @member      *   serviceGuideRequestType Indicates the requested SG </pre>	<pre> &lt;&gt; = </pre>	<pre>     }   } with {     encode "FLUTECodec"   } // End fluteConfigPrimitives    /**    *    * @desc    *   These primitives extract information from or provide it to the    *   upper interface of a HTTP stack    */   group httpConfigPrimitives {      /**      *      * @desc      *   The http request which is send by the end user to » subscribe      *   to a service.      * @member      *   connectionId Contains a unique identifier for the » (TCP)      *   connection on which the POST was received      * @member      *   serviceName Contains the name of the service which the » end      *   user want to subscribe.      */     type record HttpSubscriptionRequestIndication {       UInt connectionId,       charstring serviceName     }      /**      * Note: Modifications to types still likely to cover HTTP » use in      * service provisioning      *      * @desc      *   Used for terminal requests on interactive channel      * @member      *   connectionId Contains a unique identifier for the » (TCP)      *   connection on which the POST was received      * @member      *   serviceGuideRequestType Indicates the requested SG </pre>
---	-------------------------	---

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

» kind
    * @member
    *   keyValuePairs Indicates more detailed SG requests
    * @member
    *   streamingReceptionReport Contains a terminal streaming
    *   reception report (if requested)
    * @remark
    *   if serviceGuideRequestType and keyValuePairs are
» bothomitted
    *   then default view to service guide shall be sent (see
» OMA SG
    *   doc 5.4.3.2)
    */
    type record HttpPostIndication {
        UInt connectionId,
        ServiceGuideRequestType serviceGuideRequestType
» optional,
        KeyValuePairList keyValuePairs optional,
        StreamingReceptionReportType StreamingReceptionReport
» optional
    }

    /**
    *
    * @desc
    *   Used for terminal signalling of file repair or to
» request
    *   fragments referenced in a SGDU
    * @member
    *   connectionId Contains a unique identifier for the
» (TCP)
    *   connection on which the GET was received
    * @member
    *   messageUri The message URI
    */
    type record HttpGetIndication {
        UInt connectionId,
        AnyUri messageUri
    }

    type enumerated ServiceGuideRequestType {
        e_sgdd,
        e_sgdu,
        e_sgdd_sgdu
    }

```

```

» kind
    * @member
    *   keyValuePairs Indicates more detailed SG requests
    * @member
    *   streamingReceptionReport Contains a terminal streaming
    *   reception report (if requested)
    * @remark
    *   if serviceGuideRequestType and keyValuePairs are
» bothomitted
    *   then default view to service guide shall be sent (see
» OMA SG
    *   doc 5.4.3.2)
    */
    type record HttpPostIndication {
        UInt connectionId,
        ServiceGuideRequestType serviceGuideRequestType
» optional,
        KeyValuePairList keyValuePairs optional,
        StreamingReceptionReportType StreamingReceptionReport
» optional
    }

    /**
    *
    * @desc
    *   Used for terminal signalling of file repair or to
» request
    *   fragments referenced in a SGDU
    * @member
    *   connectionId Contains a unique identifier for the
» (TCP)
    *   connection on which the GET was received
    * @member
    *   messageUri The message URI
    */
    type record HttpGetIndication {
        UInt connectionId,
        AnyUri messageUri
    }

    type enumerated ServiceGuideRequestType {
        e_sgdd,
        e_sgdu,
        e_sgdd_sgdu
    }

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

    type record length (0 .. infinity) of KeyValuePair
» KeyValuePairList;

    type record KeyValuePair {
        charstring key,
        charstring valuePart optional
    }

    /**
     *
     * @desc
     *   Used for terminal response on interactive channel
     * @member
     *   connectionId Contains a unique identifier for the
» (TCP)
     *   connection on which the http request was received
     * @member
     *   responseCode http code to be used, e.g., 200 (OK)
     * @member
     *   responsePayload This field shall only be set in case
» of
     *   file repair responses; syntax is follows the one used
» in the
     *   corresponding HTTP GET URI
     */
    type record HttpResponseRequest {
        UInt connectionId,
        UInt responseCode,
        ResponsePayload responsePayload optional
    }

    type union ResponsePayload {
        ServiceGuidePayload ServiceGuidePayload,
        charstring text
    }

    type record ServiceGuidePayload {
        ServiceGuideResponse serviceGuideResponse,
        ServiceGuideDeliveryUnit serviceGuideDeliveryUnit
» optional
    }

    /**
     *
     * @desc
     *   Send by the SA to indicate that a http response could

```

```

    type record length (0 .. infinity) of KeyValuePair
» KeyValuePairList;

    type record KeyValuePair {
        charstring key,
        charstring valuePart optional
    }

    /**
     *
     * @desc
     *   Used for terminal response on interactive channel
     * @member
     *   connectionId Contains a unique identifier for the
» (TCP)
     *   connection on which the http request was received
     * @member
     *   responseCode http code to be used, e.g., 200 (OK)
     * @member
     *   responsePayload This field shall only be set in case
» of
     *   file repair responses; syntax is follows the one used
» in the
     *   corresponding HTTP GET URI
     */
    type record HttpResponseRequest {
        UInt connectionId,
        UInt responseCode,
        ResponsePayload responsePayload optional
    }

    type union ResponsePayload {
        ServiceGuidePayload ServiceGuidePayload,
        charstring text
    }

    type record ServiceGuidePayload {
        ServiceGuideResponse serviceGuideResponse,
        ServiceGuideDeliveryUnit serviceGuideDeliveryUnit
» optional
    }

    /**
     *
     * @desc
     *   Send by the SA to indicate that a http response could

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

» be send
    * or not
    * @member
    * returnCode 0 - success 1+ - error code
    * @member
    * reason should be set if return code > 0
    */
    type record HttpResponseResponse {
        UInt returnCode,
        charstring reason optional
    }
} // end httpConfigPrimitives
group smsConfigPrimitives {
    /**
    *
    * @desc
    * Send by the SA to indicate that a sms was received.
    * @member
    * messageBody The message body of the sms
    */
    type record SmsGetIndication {
        charstring messageBody // should be further
    }
} // end smsConfigPrimitives
group mmsConfigPrimitives {
    /**
    *
    * @desc
    * Send by the SA to indicate that a mms was received.
    * @member
    * messageBody The message body of the mms
    */
    type record MmsGetIndication {
        charstring messageBody // should be further
    }
} // end mmsConfigPrimitives
/**
*
* @desc This group specifies information to be forwarded from TTCN-3
» to the streaming server
*/
group streamingServerPrimitives {
    /**
    *

```

```

» be send
    * or not
    * @member
    * returnCode 0 - success 1+ - error code
    * @member
    * reason should be set if return code > 0
    */
    type record HttpResponseResponse {
        UInt returnCode,
        charstring reason optional
    }
} // end httpConfigPrimitives
group smsConfigPrimitives {
    /**
    *
    * @desc
    * Send by the SA to indicate that a sms was received.
    * @member
    * messageBody The message body of the sms
    */
    type record SmsGetIndication {
        charstring messageBody // should be further
    }
} // end smsConfigPrimitives
group mmsConfigPrimitives {
    /**
    *
    * @desc
    * Send by the SA to indicate that a mms was received.
    * @member
    * messageBody The message body of the mms
    */
    type record MmsGetIndication {
        charstring messageBody // should be further
    }
} // end mmsConfigPrimitives
/**
*
* @desc This group specifies information to be forwarded from TTCN-3
» to the streaming server
*/
group streamingServerPrimitives {
    /**
    *

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

* @desc
*   Sends start streaming request to a streaming server
* @member
*   streamId The stream identifier
* @member
*   fileName The file name reference to a file which
» should be
*   streamed.
* @member
*   sdp Includes IP information, codecs, rates, unicast vs
*   broadcast, etc
* @member
*   contentProtection For future use
* @member
*   serviceProtection For future use
*/
type record StartStreamingRequest {
    UInt streamId,
    charstring fileName,
    Sdp sdp,
    ContentProtection contentProtection optional, // for
» future use
    ServiceProtection serviceProtection optional // for
» future use
};

type charstring ContentProtection;
type charstring ServiceProtection;

/**
*
* @desc
*   Send by the SA to indicate that a start streaming
» request
*   could be send.
* @member
*   returnCode 0 - success 1+ - error code
* @member
*   reason should be set if return code > 0
*/
type record StartStreamingResponse {
    UInt returnCode,
    charstring reason optional
}

/**

```

```

* @desc
*   Sends start streaming request to a streaming server
* @member
*   streamId The stream identifier
* @member
*   fileName The file name reference to a file which
» should be
*   streamed.
* @member
*   sdp Includes IP information, codecs, rates, unicast vs
*   broadcast, etc
* @member
*   contentProtection For future use
* @member
*   serviceProtection For future use
*/
type record StartStreamingRequest {
    UInt streamId,
    charstring fileName,
    Sdp sdp,
    ContentProtection contentProtection optional, // for
» future use
    ServiceProtection serviceProtection optional // for
» future use
};

type charstring ContentProtection;
type charstring ServiceProtection;

/**
*
* @desc
*   Send by the SA to indicate that a start streaming
» request
*   could be send.
* @member
*   returnCode 0 - success 1+ - error code
* @member
*   reason should be set if return code > 0
*/
type record StartStreamingResponse {
    UInt returnCode,
    charstring reason optional
}

/**

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

<pre> * * @desc *   Send to a streaming server to stop streaming. * @member *   streamId The stream identifier */ type record StopStreamingRequest {     UInt streamId };  /** * * @desc *   Send by the SA to indicate that a stop streaming » request *   could be send. * @member *   returnCode 0 - success 1+ - error code * @member *   reason should be set if return code &gt; 0 */ type record StopStreamingResponse {     UInt returnCode,     charstring reason optional } </pre>		<pre> * * @desc *   Send to a streaming server to stop streaming. * @member *   streamId The stream identifier */ type record StopStreamingRequest {     UInt streamId };  /** * * @desc *   Send by the SA to indicate that a stop streaming » request *   could be send. * @member *   returnCode 0 - success 1+ - error code * @member *   reason should be set if return code &gt; 0 */ type record StopStreamingResponse {     UInt returnCode,     charstring reason optional } </pre>
<pre> } with { // change 26 (WK 6): support of new codec         encode "StreamingServerCodec" </pre>	+ -	
<pre> } // end streamServerPrimitives  group fileServerPrimitives {     /**     *     * @desc     *   The message wich triggers the file server to transfer » files.     * @member     *   transferId The id of the transfer     * @member     *   fileList The list of file which should be transfered » by the     *   file server     * @member     *   sdp Includes IP information, codecs, rates, unicast vs     *   broadcast, etc     * @member     *   contentProtection for futher use </pre>	=	<pre> } // end streamServerPrimitives  group fileServerPrimitives {     /**     *     * @desc     *   The message wich triggers the file server to transfer » files.     * @member     *   transferId The id of the transfer     * @member     *   fileList The list of file which should be transfered » by the     *   file server     * @member     *   sdp Includes IP information, codecs, rates, unicast vs     *   broadcast, etc     * @member     *   contentProtection for futher use </pre>



Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

        * @member
        *   serviceProtection for futher use
        */
type record StartFileTransferRequest {
    UInt transferId,
    TransferFileList fileList,
    Sdp sdp,
    ContentProtection contentProtection optional,
    ServiceProtection serviceProtection optional // for
» future use
};

/**
 *
 * @desc
 *   A list of Transfer files used by the file server
 */
type set of TransferFile TransferFileList;

/**
 *
 * @desc
 *   This type represents a Transfer file. A transfer file
» could
 *   be a file reference, a zip file or an interactivity
» media
 *   document.
 * @member
 *   fileRef
 * @member
 *   zipFile
 * @member
 *   interactivityMediaDocument
 */
type union TransferFile {
    FileReference fileRef,
    ZipFile zipFile,
    InteractivityMediaDocumentType
» InteractivityMediaDocument
}

/**
 *
 * @desc
 *   This type represents a file reference.
 */

```

```

        * @member
        *   serviceProtection for futher use
        */
type record StartFileTransferRequest {
    UInt transferId,
    TransferFileList fileList,
    Sdp sdp,
    ContentProtection contentProtection optional,
    ServiceProtection serviceProtection optional // for
» future use
};

/**
 *
 * @desc
 *   A list of Transfer files used by the file server
 */
type set of TransferFile TransferFileList;

/**
 *
 * @desc
 *   This type represents a Transfer file. A transfer file
» could
 *   be a file reference, a zip file or an interactivity
» media
 *   document.
 * @member
 *   fileRef
 * @member
 *   zipFile
 * @member
 *   interactivityMediaDocument
 */
type union TransferFile {
    FileReference fileRef,
    ZipFile zipFile,
    InteractivityMediaDocumentType
» InteractivityMediaDocument
}

/**
 *
 * @desc
 *   This type represents a file reference.
 */

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

type charstring FileReference;

/**
 *
 * @desc
 *   This type is a list of file references.
 */
type set of FileReference FileReferenceList;

/**
 *
 * @desc
 *   This type represents a zip file. It includes all
» information
 *   for building the zip file at the test adaption.
 * @member
 *   name The name of the zip file
 * @member
 *   fileList The list of file references which should be
» included
 *   inside the zip file
 */
type record ZipFile {
    charstring name,
    FileReferenceList fileList
}

/**
 *
 * @desc
 *   This response will be send by the file server to
» indicate the
 *   status of the file transfer start.
 * @member
 *   returnCode 0 - success, 1+ - error code
 * @member
 *   reason should be set if return code > 0
 */
type record StartFileTransferResponse {
    UInt returnCode,
    charstring reason optional
}

/**
 *
 * @desc

```

```

type charstring FileReference;

/**
 *
 * @desc
 *   This type is a list of file references.
 */
type set of FileReference FileReferenceList;

/**
 *
 * @desc
 *   This type represents a zip file. It includes all
» information
 *   for building the zip file at the test adaption.
 * @member
 *   name The name of the zip file
 * @member
 *   fileList The list of file references which should be
» included
 *   inside the zip file
 */
type record ZipFile {
    charstring name,
    FileReferenceList fileList
}

/**
 *
 * @desc
 *   This response will be send by the file server to
» indicate the
 *   status of the file transfer start.
 * @member
 *   returnCode 0 - success, 1+ - error code
 * @member
 *   reason should be set if return code > 0
 */
type record StartFileTransferResponse {
    UInt returnCode,
    charstring reason optional
}

/**
 *
 * @desc

```

Datei: LibBCast\_ServicePrimitives\_TypesAndValues.ttcn  
(Fortsetzung)

```

        *      Send to a file server to stop streaming.
        * @member
        *      transferId The stream identifier
        */
    type record StopFileTransferRequest {
        UInt transferId
    };

    /**
     *
     * @desc
     *      Send by the SA to indicate that a stop fiel transfer
» request

     *      could be send.
     * @member
     *      returnCode 0 - success 1+ - error code
     * @member
     *      reason should be set if return code > 0
     */
    type record StopFileTransferResponse {
        UInt returnCode,
        charstring reason optional
    }

    } // end fileServerPrimitives
} // end module AtsBCast_ConfigPrimitives_TypesAndValues

```

```

        *      Send to a file server to stop streaming.
        * @member
        *      transferId The stream identifier
        */
    type record StopFileTransferRequest {
        UInt transferId
    };

    /**
     *
     * @desc
     *      Send by the SA to indicate that a stop fiel transfer
» request

     *      could be send.
     * @member
     *      returnCode 0 - success 1+ - error code
     * @member
     *      reason should be set if return code > 0
     */
    type record StopFileTransferResponse {
        UInt returnCode,
        charstring reason optional
    }

    } // end fileServerPrimitives
} // end module AtsBCast_ConfigPrimitives_TypesAndValues

```

Datei: LibCommon\_TextStrings.ttcn

```

/*
 * @author   ETSI STF 276
 * @version  $Id$
 * @desc     A collection of text string type and value definitions which
 *           may be useful in the implementation of any TTCN-3 test
 *           suite. "Text string" refers to TTCN-3 charstring and universal
 *           charstring types.
 * @remark   End users should be aware that any changes made to the in
 *           definitions this module may be overwritten in future releases.
 *           End users are encouraged to contact the distributors of this
 *           module regarding their modifications or additions so that
» future
 *           updates will include your changes.
 */
module LibCommon_TextStrings {

    /*

```

```

=
/*
 * @author   ETSI STF 276
 * @version  $Id$
 * @desc     A collection of text string type and value definitions which
 *           may be useful in the implementation of any TTCN-3 test
 *           suite. "Text string" refers to TTCN-3 charstring and universal
 *           charstring types.
 * @remark   End users should be aware that any changes made to the in
 *           definitions this module may be overwritten in future releases.
 *           End users are encouraged to contact the distributors of this
 *           module regarding their modifications or additions so that
» future
 *           updates will include your changes.
 */
module LibCommon_TextStrings {

    /*

```

## Datei: LibCommon\_TextStrings.ttcn (Fortsetzung)

```

* @desc These constants can be used to add special characters into
*       TTCN-3 text strings by using the concatenation operator.
*       Example use:
*       var charstring v_text := "Hi!" & c_CRLF & "Hello!";
*/
group usefulConstants {

  const charstring c_NUL := oct2str('00'O);
  const charstring c_SOH := oct2str('01'O);
  const charstring c_STX := oct2str('02'O);
  const charstring c_ETX := oct2str('03'O);
  const charstring c_EOT := oct2str('04'O);
  const charstring c_ENQ := oct2str('05'O);
  const charstring c_ACK := oct2str('06'O);
  const charstring c_BEL := oct2str('07'O);
  const charstring c_BS  := oct2str('08'O);
  const charstring c_TAB := oct2str('09'O);
  const charstring c_LF  := oct2str('0A'O);
  const charstring c_VT  := oct2str('0B'O);
  const charstring c_FF  := oct2str('0C'O);
  const charstring c_CR  := oct2str('0D'O);
  const charstring c_SO  := oct2str('0E'O);
  const charstring c_SI  := oct2str('0F'O);
  const charstring c_DLE := oct2str('10'O);
  const charstring c_DC1 := oct2str('11'O);
  const charstring c_DC2 := oct2str('12'O);
  const charstring c_DC3 := oct2str('13'O);
  const charstring c_DC4 := oct2str('14'O);
  const charstring c_NAK := oct2str('15'O);
  const charstring c_SYN := oct2str('16'O);
  const charstring c_ETB := oct2str('17'O);
  const charstring c_CAN := oct2str('18'O);
  const charstring c_EM  := oct2str('19'O);
  const charstring c_SUB := oct2str('1A'O);
  const charstring c_ESC := oct2str('1B'O);
  const charstring c_FS  := oct2str('1C'O);
  const charstring c_GS  := oct2str('1D'O);
  const charstring c_RS  := oct2str('1E'O);
  const charstring c_US  := oct2str('1F'O);
  const charstring c_DEL := oct2str('7F'O);

  const charstring c_CRLF := oct2str('0D'O) & oct2str('0A'O);

} // end group usefulConstants

/*
* @remark Number in name indicates string length in number of

```

```

* @desc These constants can be used to add special characters into
*       TTCN-3 text strings by using the concatenation operator.
*       Example use:
*       var charstring v_text := "Hi!" & c_CRLF & "Hello!";
*/
group usefulConstants {

  const charstring c_NUL := oct2str('00'O);
  const charstring c_SOH := oct2str('01'O);
  const charstring c_STX := oct2str('02'O);
  const charstring c_ETX := oct2str('03'O);
  const charstring c_EOT := oct2str('04'O);
  const charstring c_ENQ := oct2str('05'O);
  const charstring c_ACK := oct2str('06'O);
  const charstring c_BEL := oct2str('07'O);
  const charstring c_BS  := oct2str('08'O);
  const charstring c_TAB := oct2str('09'O);
  const charstring c_LF  := oct2str('0A'O);
  const charstring c_VT  := oct2str('0B'O);
  const charstring c_FF  := oct2str('0C'O);
  const charstring c_CR  := oct2str('0D'O);
  const charstring c_SO  := oct2str('0E'O);
  const charstring c_SI  := oct2str('0F'O);
  const charstring c_DLE := oct2str('10'O);
  const charstring c_DC1 := oct2str('11'O);
  const charstring c_DC2 := oct2str('12'O);
  const charstring c_DC3 := oct2str('13'O);
  const charstring c_DC4 := oct2str('14'O);
  const charstring c_NAK := oct2str('15'O);
  const charstring c_SYN := oct2str('16'O);
  const charstring c_ETB := oct2str('17'O);
  const charstring c_CAN := oct2str('18'O);
  const charstring c_EM  := oct2str('19'O);
  const charstring c_SUB := oct2str('1A'O);
  const charstring c_ESC := oct2str('1B'O);
  const charstring c_FS  := oct2str('1C'O);
  const charstring c_GS  := oct2str('1D'O);
  const charstring c_RS  := oct2str('1E'O);
  const charstring c_US  := oct2str('1F'O);
  const charstring c_DEL := oct2str('7F'O);

  const charstring c_CRLF := oct2str('0D'O) & oct2str('0A'O);

} // end group usefulConstants

/*
* @remark Number in name indicates string length in number of

```

Datei: LibCommon\_TextStrings.ttcn (Fortsetzung)

```

*           _characters_
*/
group textStringSubTypes {

    type charstring String1    length(1) with { encode "length(1)";
    type charstring String2    length(2) with { encode "length(2)";
    type charstring String3    length(3) with { encode "length(3)";
    type charstring String4    length(4) with { encode "length(4)";
    type charstring String5    length(5) with { encode "length(5)";
    type charstring String6    length(6) with { encode "length(6)";
    type charstring String7    length(7) with { encode "length(7)";
    type charstring String8    length(8) with { encode "length(8)";
    type charstring String9    length(9) with { encode "length(9)";
    type charstring String10   length(10) with { encode "length(10)";
    type charstring String11   length(11) with { encode "length(11)";
    type charstring String12   length(12) with { encode "length(12)";
    type charstring String13   length(13) with { encode "length(13)";
    type charstring String14   length(14) with { encode "length(14)";
    type charstring String15   length(15) with { encode "length(15)";
    type charstring String16   length(16) with { encode "length(16)";

    type charstring String1To63 length(1..63) with {encode "length(1..63)";
    type charstring String1To64 length(1..64) with {encode "length(1..64)";
    type charstring String1To127 length(1..127) with {encode
» "length(1..127)";
    type charstring String1To128 length(1..128) with {encode
» "length(1..128)";
    type charstring String1to255 length(1..255) with {encode
» "length(1..255)";

} // end stringSubTypes

group usefulTextStringTypes {

    type universal charstring UnicodeText;

    type universal charstring UnicodeText1to255 length(1..255)
with {encode "length(1..255)";

    type charstring AlphaNum ("0".."9","a".."z","A".."Z");
    type AlphaNum    AlphaNum2 length(2) with { encode "length(2)";
    type AlphaNum    AlphaNum1To32 length(1..32) with {encode
» "length(1..32)";

```

```

// change 27 (WK 6): new definition of IP address versions
type enumerated IPAddressType {
    IPv4,

```

```

*           _characters_
*/
group textStringSubTypes {

    type charstring String1    length(1) with { encode "length(1)";
    type charstring String2    length(2) with { encode "length(2)";
    type charstring String3    length(3) with { encode "length(3)";
    type charstring String4    length(4) with { encode "length(4)";
    type charstring String5    length(5) with { encode "length(5)";
    type charstring String6    length(6) with { encode "length(6)";
    type charstring String7    length(7) with { encode "length(7)";
    type charstring String8    length(8) with { encode "length(8)";
    type charstring String9    length(9) with { encode "length(9)";
    type charstring String10   length(10) with { encode "length(10)";
    type charstring String11   length(11) with { encode "length(11)";
    type charstring String12   length(12) with { encode "length(12)";
    type charstring String13   length(13) with { encode "length(13)";
    type charstring String14   length(14) with { encode "length(14)";
    type charstring String15   length(15) with { encode "length(15)";
    type charstring String16   length(16) with { encode "length(16)";

    type charstring String1To63 length(1..63) with {encode "length(1..63)";
    type charstring String1To64 length(1..64) with {encode "length(1..64)";
    type charstring String1To127 length(1..127) with {encode
» "length(1..127)";
    type charstring String1To128 length(1..128) with {encode
» "length(1..128)";
    type charstring String1to255 length(1..255) with {encode
» "length(1..255)";

} // end stringSubTypes

group usefulTextStringTypes {

    type universal charstring UnicodeText;

    type universal charstring UnicodeText1to255 length(1..255)
with {encode "length(1..255)";

    type charstring AlphaNum ("0".."9","a".."z","A".."Z");
    type AlphaNum    AlphaNum2 length(2) with { encode "length(2)";
    type AlphaNum    AlphaNum1To32 length(1..32) with {encode
» "length(1..32)";

```

+-

Datei: LibCommon\_TextStrings.ttcn (Fortsetzung)

<div>IPv6</div> <div>}</div>		
} // end group usefulTextStringTypes	=	} // end group usefulTextStringTypes
} // end module LibCommon_TextStrings		} // end module LibCommon_TextStrings