

Modus: Alle Zeilen
Linker Ursprungsordner: W:\BCAST\OMA\CRs\own\OMA-IOP-TTCN-2008-xxxx-CR_ServProt-101d-e\modified TTCN-3 sources
Rechter Ursprungsordner: W:\BCAST\OMA\CRs\own\OMA-IOP-TTCN-2008-xxxx-CR_ServProt-101d-e\orig

Datei: AtsBCast_Main_Functions.ttcn

1	/**	=	1	/**
2	*		2	*
3	* @author		3	* @author
4	* ETSI CTI BCAST CON Project		4	* ETSI CTI BCAST CON Project
5	* @version		5	* @version
6	* \$Id\$		6	* \$Id\$
7	* @desc		7	* @desc
8	* This module specifies functions which create either BCAST control or		8	* This module specifies functions which create either BCAST control or
9	* Upper Tester components and start library BCAST or Upper Tester functions		9	* Upper Tester components and start library BCAST or Upper Tester functions
10	* on them.		10	* on them.
11	*/		11	*/
12	module AtsBCast_Main_Functions {		12	module AtsBCast_Main_Functions {
13	import from LibBCast_Common_TypesAndValues all;		13	import from LibBCast_Common_TypesAndValues all;
14	import from LibBCast_ServiceGuide_TypesAndValues all;		14	import from LibBCast_ServiceGuide_TypesAndValues all;
15	import from AtsBCast_ServiceGuide_Functions all;		15	import from AtsBCast_ServiceGuide_Functions all;
16	import from LibBCast_Common_Templates all;		16	import from LibBCast_Common_Templates all;
17	import from LibBCast_ServicePrimitives_TypesAndValues all;		17	import from LibBCast_ServicePrimitives_TypesAndValues all;
18	import from LibBCast_UpperTester_Functions all;		18	import from LibBCast_UpperTester_Functions all;
19	import from AtsBCast_TestSystem all;		19	import from AtsBCast_TestSystem all;
20	import from AtsBCast_ModuleParameters all;		20	import from AtsBCast_ModuleParameters all;
21	import from LibBCast_ModuleParameters all;		21	import from LibBCast_ModuleParameters all;
22	import from LibCommon_Time all;		22	import from LibCommon_Time all;
23	import from LibCommon_BasicTypesAndValues all;		23	import from LibCommon_BasicTypesAndValues all;
24	import from LibBCast_UpperTesterPrimitives_TypesAndValues all;		24	import from LibBCast_UpperTesterPrimitives_TypesAndValues all;
25	import from LibBCast_BCastNetworkControl_Functions all;		25	import from LibBCast_BCastNetworkControl_Functions all;
26	import from LibBCast_BCastNetworkData_Functions all;		26	import from LibBCast_BCastNetworkData_Functions all;
27	import from LibBCast_UpperTester_Functions all;		27	import from LibBCast_UpperTester_Functions all;
28	// change 1 (WK18): for content protection testst)		28	// change 1 (WK18): for content protection testst)
29	// change 2 (WK23): GBA authentication/authorization extensions for content protection tests	+-		
30	import from LibCommon_GBA_BSF {	=	29	import from LibCommon_GBA_BSF {
31	function BSF_ServerSimulation;		30	function BSF_ServerSimulation;
32	function f_bsf_setBtid;	+-		
33	function f_bsf_init;			
34	}	=	31	}
35	// change 1 (WK18): for content protection tests		32	// change 1 (WK18): for content protection tests
36	// change 1 (WK23): Service Request extensions for content protection tests	+-		
37	import from LibCommon_BSM {	=	33	import from LibCommon_BSM {
38	function BSM_ServerSimulation;	<>	34	function MBMS_User_Service_Registration_ServerSimulation;
39	function f_bsm_setProtectionKeyId;			
40	function f_bsm_setBtid;			
41	function f_bsm_setProtectionKeysAndValues;			
42	}	=	35	}
43	import from LibCommon_GBA_BSF_TypesAndValues {	+-		
44	group GBA;			
45	}			
46				
47	// change 1 (WK34): Service Protection: Mikey			
48	import from LibCommon_Mikey_TypesAndValues all;			
49	import from LibCommon_Mikey all;			
50				
51	// change 3 (WK34): SService Protection: secure streaming service			
52	import from LibCommon_DataStrings all;			
53		=	36	
54	group bcastCtrlMainFunctions {		37	group bcastCtrlMainFunctions {
55			38	
56	// change 01 (WK 6): global time definition		39	// change 01 (WK 6): global time definition
57	/**		40	/**
58	*		41	*
59	* @desc		42	* @desc
60	* initialize the start and end time. The start time will be		43	* initialize the start and end time. The start time will be
61	* set to a value which is test execution plus the start offset in seconds and the		44	* set to a value which is test execution plus the start offset in seconds and the
62	* endtime will be set to a value equal to the start time plus the specified duration in		45	* endtime will be set to a value equal to the start time plus the specified duration in
63	* @param		46	* @param
64	* p_startOffset Start Offset in seconds		47	* p_startOffset Start Offset in seconds
65	* @param		48	* @param
66	* p_duration Duration in seconds between the start and the end		49	* p_duration Duration in seconds between the start and the end

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

67	*/	50	*/
68	function f_main_ctrl_InitStartEndTime(integer p_startOffset, integer p_duration) runs on B	51	function f_main_ctrl_InitStartEndTime(integer p_startOffset, integer p_duration) runs on B
69	v_startTime := f_getNTPTime() + p_startOffset;	52	v_startTime := f_getNTPTime() + p_startOffset;
70	v_endTime := v_startTime + p_duration;	53	v_endTime := v_startTime + p_duration;
71	vc_CTRL.start (f_ctrl_InitStartEndTime(v_startTime,v_endTime));	54	vc_CTRL.start (f_ctrl_InitStartEndTime(v_startTime,v_endTime));
72	vc_CTRL.done;	55	vc_CTRL.done;
73	vc_DATA.start (f_data_InitStartEndTime(v_startTime,v_endTime)); // change 15 (WK18	56	vc_DATA.start (f_data_InitStartEndTime(v_startTime,v_endTime)); // change 15 (WK18
74	vc_DATA.done; // change 15 (WK18	57	vc_DATA.done; // change 15 (WK18
75	}	58	}
76		59	
77	/**	60	/**
78	*	61	*
79	* @desc	62	* @desc
80	* This function creates a test component and starts a function that	63	* This function creates a test component and starts a function that
81	* waits for the receiving of a http subscription	64	* waits for the receiving of a http subscription
82	* indication.	65	* indication.
83	* @param	66	* @param
84	* p_serviceName The service name for the subscription	67	* p_serviceName The service name for the subscription
85	* @verdict	68	* @verdict
86	* pass The substription was successfull received for the given	69	* pass The substription was successfull received for the given
87	* service.	70	* service.
88	* @verdict	71	* @verdict
89	* fail A message was received which is not expected.	72	* fail A message was received which is not expected.
90	* @verdict	73	* @verdict
91	* inconc A guard timer expires.	74	* inconc A guard timer expires.
92	*/	75	*/
93	function f_main_ctrl_awaitingSubscription(charstring p_serviceName) runs on BCastMainCompor	76	function f_main_ctrl_awaitingSubscription(charstring p_serviceName) runs on BCastMainCompor
94	vc_CTRL.start (f_ctrl_awaitingHttpSubscriptionIndication(p_serviceName));	77	vc_CTRL.start (f_ctrl_awaitingHttpSubscriptionIndication(p_serviceName));
95	vc_CTRL.done;	78	vc_CTRL.done;
96	}	79	}
97		80	
98	/**	81	/**
99	*	82	*
100	* @desc	83	* @desc
101	* This function creates a test component and starts a function that	84	* This function creates a test component and starts a function that
102	* waits for a mms indication.	85	* waits for a mms indication.
103	* @param	86	* @param
104	* content the content which should be inside of the mms.	87	* content the content which should be inside of the mms.
105	* @verdict	88	* @verdict
106	* pass The indication was successfull received for the given	89	* pass The indication was successfull received for the given
107	* service.	90	* service.
108	* @verdict	91	* @verdict
109	* fail A message was received which is not expected.	92	* fail A message was received which is not expected.
110	* @verdict	93	* @verdict
111	* inconc A guard timer expires.	94	* inconc A guard timer expires.
112	*/	95	*/
113	function f_main_ctrl_awaitingMMS(charstring content) runs on BCastMainComponent {	96	function f_main_ctrl_awaitingMMS(charstring content) runs on BCastMainComponent {
114		97	
115	vc_CTRL.start (f_ctrl_awaitingMMS(content));	98	vc_CTRL.start (f_ctrl_awaitingMMS(content));
116	vc_CTRL.done;	99	vc_CTRL.done;
117	}	100	}
118		101	
119	/**	102	/**
120	*	103	*
121	* @desc	104	* @desc
122	* This function creates a test component and starts a function that	105	* This function creates a test component and starts a function that
123	* waits for a sms indication.	106	* waits for a sms indication.
124	* @param	107	* @param
125	* content the content which should be inside of the mms.	108	* content the content which should be inside of the mms.
126	* @verdict	109	* @verdict
127	* pass The indication was successfull received for the given	110	* pass The indication was successfull received for the given
128	* service.	111	* service.
129	* @verdict	112	* @verdict
130	* fail A message was received which is not expected.	113	* fail A message was received which is not expected.
131	* @verdict	114	* @verdict
132	* inconc A guard timer expires.	115	* inconc A guard timer expires.
133	*/	116	*/
134	function f_main_ctrl_awaitingSMS(charstring content) runs on BCastMainComponent {	117	function f_main_ctrl_awaitingSMS(charstring content) runs on BCastMainComponent {
135	vc_CTRL.start (f_ctrl_awaitingSMS(content));	118	vc_CTRL.start (f_ctrl_awaitingSMS(content));
136	vc_CTRL.done;	119	vc_CTRL.done;
137	}	120	}

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

138		121	
139	/**	122	/**
140	*	123	*
141	* @desc	124	* @desc
142	* This function creates a test component and starts a function that	125	* This function creates a test component and starts a function that
143	* broadcasts a service guide over the flute protocol	126	* broadcasts a service guide over the flute protocol
144	* @param	127	* @param
145	* p_SGDU The ServiceGuideDeliveryUnit which should be	128	* p_SGDU The ServiceGuideDeliveryUnit which should be
146	* broadcasted	129	* broadcasted
147	* @param	130	* @param
148	* p_Encoding The encoding type for compression in case that the	131	* p_Encoding The encoding type for compression in case that the
149	* service guide should be compressed otherwise this parameter	132	* service guide should be compressed otherwise this parameter
150	* should be omitted.	133	* should be omitted.
151	* @param	134	* @param
152	* p_FluteUpdateID The Flute Instance ID of the Flute Session	135	* p_FluteUpdateID The Flute Instance ID of the Flute Session
153	* @verdict	136	* @verdict
154	* pass in case the broadcast of the service guide was	137	* pass in case the broadcast of the service guide was
155	* successful	138	* successful
156	* @verdict	139	* @verdict
157	* fail in case that the broadcast was not successful	140	* fail in case that the broadcast was not successful
158	* @verdict	141	* @verdict
159	* inconc in case of the timer runs out of time	142	* inconc in case of the timer runs out of time
160	*/	143	*/
161	// CR (WK 34/2008): Flute Update Signalling	144	// CR (WK 34/2008): Flute Update Signalling
162	function f_main_ctrl_broadcastServiceGuide(ServiceGuideDeliveryUnit p_SGDU,	145	function f_main_ctrl_broadcastServiceGuide(ServiceGuideDeliveryUnit p_SGDU,
163	template charstring p_Encoding,	146	template charstring p_Encoding,
164	UInt32 p_FluteUpdateID) runs on BCastMainComponen	147	UInt32 p_FluteUpdateID) runs on BCastMainComponen
165	vc_CTRL.start (148	vc_CTRL.start (
166	f_ctrl_broadcastServiceGuide(149	f_ctrl_broadcastServiceGuide(
167	p_SGDU,	150	p_SGDU,
168	PX_SGDU_ID,	151	PX_SGDU_ID,
169	PX_SGDD_ID,	152	PX_SGDD_ID,
170	p_Encoding,	153	p_Encoding,
171	p_FluteUpdateID)	154	p_FluteUpdateID)
172);	155);
173	vc_CTRL.done;	156	vc_CTRL.done;
174	}	157	}
175		158	
176	/**	159	/**
177	*	160	*
178	* @desc	161	* @desc
179	* This functions expect a http post request in order to deliver	162	* This functions expect a http post request in order to deliver
180	* the service guide over the interaction channel.	163	* the service guide over the interaction channel.
181	* @param	164	* @param
182	* p_SGDU The Service Guide Delivery Unit to send	165	* p_SGDU The Service Guide Delivery Unit to send
183	* @verdict	166	* @verdict
184	* pass the service guide could successful delivered over the	167	* pass the service guide could successful delivered over the
185	* interaction channel	168	* interaction channel
186	* @verdict	169	* @verdict
187	* fail the service fuide could not successful deflivered.	170	* fail the service fuide could not successful deflivered.
188	* @verdict	171	* @verdict
189	* incon the guard timer runs out of time.	172	* incon the guard timer runs out of time.
190	*/	173	*/
191	function f_main_ctrl_ServiceGuideOnRequest(ServiceGuideDeliveryUnit p_SGDU) runs on BCastMa	174	function f_main_ctrl_ServiceGuideOnRequest(ServiceGuideDeliveryUnit p_SGDU) runs on BCastMa
192		175	
193	vc_UTC.start(f_ut_GetServiceGuide(PX_SGDD_ID));	176	vc_UTC.start(f_ut_GetServiceGuide(PX_SGDD_ID));
194		177	
195	vc_CTRL.start (178	vc_CTRL.start (
196	f_ctrl_ServiceGuideViaHttp(179	f_ctrl_ServiceGuideViaHttp(
197	p_SGDU,	180	p_SGDU,
198	PX_SGDU_ID,	181	PX_SGDU_ID,
199	PX_SGDD_ID));	182	PX_SGDD_ID));
200	vc_CTRL.done;	183	vc_CTRL.done;
201		184	
202	vc_UTC.done;	185	vc_UTC.done;
203	}	186	}
204		187	
205	// change 1 (WK18): for content protection tests	188	// change 1 (WK18): for content protection tests
206	/**	189	/**
207	*	190	*
208	* @desc	191	* @desc

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

209	* This function starts a BSF Server including a HTTP Digest challenge.		192	* This function starts a BSF Server including a HTTP Digest challenge.	
210	* The server is implemented as parallel component.		193	* The server is implemented as parallel component.	
211	* @verdict		194	* @verdict	
212	* pass HTTP Digest (client response value matches XRES) challenge is successful.		195	* pass HTTP Digest (client response value matches XRES) challenge is successful.	
213	* @verdict		196	* @verdict	
214	* fail client response value does not match XRES		197	* fail client response value does not match XRES	
215	* @verdict		198	* @verdict	
216	* incon the guard timer runs out of time.		199	* incon the guard timer runs out of time.	
217	*/		200	*/	
218	function f_main_ctrl_GBA_U_Bootstrapping(boolean p_isParallelComponent) runs on BCastMainCo		201	function f_main_ctrl_GBA_U_Bootstrapping(boolean p_isParallelComponent) runs on BCastMainCo	
219	vc_BSF.start(BSF_ServerSimulation(p_isParallelComponent));		202	vc_BSF.start(BSF_ServerSimulation(p_isParallelComponent));	
220	if (not p_isParallelComponent) {		203	if (not p_isParallelComponent) {	
221	vc_BSF.done;		204	vc_BSF.done;	
222	}		205	}	
223	}		206	}	
224			207		
225	// change 1 (WK23): Service Request extensions for content protection tests	<>	208	// change 1 (WK18): for content protection tests	
226	/**	=	209	/**	
227	* @desc		210	* @desc	
228	* This function starts a BSM Server including a HTTP Digest challenge.		211	* This function starts a BSM Server including a HTTP Digest challenge.	
229	* (The server could be implemented as parallel component.)		212	* (The server could be implemented as parallel component.)	
230	* @verdict		213	* @verdict	
231	* pass HTTP Digest (client response value matches XRES) challenge is successful.		214	* pass HTTP Digest (client response value matches XRES) challenge is successful.	
232	* @verdict		215	* @verdict	
233	* fail client response value does not match XRES		216	* fail client response value does not match XRES	
234	* @verdict		217	* @verdict	
235	* incon the guard timer runs out of time.		218	* incon the guard timer runs out of time.	
236	*/		219	*/	
237	function f_main_ctrl_BSM_Procedures(boolean p_activateUserServiceRegistration, boolean p_Us<>		220	function f_main_ctrl_MBMS_User_Service_Procedure(boolean p_isParallelComponent) runs on BCa	
238	if (p_activateUserServiceRegistration or p_activateServiceRequest) {		221	vc_BSM.start(MBMS_User_Service_Registration_ServerSimulation(p_isParallelComponent)	
239	vc_BSM.start(BSM_ServerSimulation(p_activateUserServiceRegistration, p_UsrA				
240	if ((not p_UsrActAsParallelComponent) or (not p_SrActAsParallelComponent))				
241	vc_BSM.done;				
242	}				
243	}				
244	}				
245					
246	// change 2 (WK23): GBA authentication/authorization extensions for content protection test				
247	/**				
248	* @desc This function sets the protection key id in the BSM component.		222	if (not p_isParallelComponent) {	
249	* @param p_protectionKeyId				
250	* @verdict				
251	*/				
252	function f_main_ctrl_SetProtectionKeyId(hexstring p_protectionKeyId) runs on BCastMainComp		223	vc_BSM.done;	
253	vc_BSM.start(f_bsm_setProtectionKeyId(p_protectionKeyId));				
254	vc_BSM.done;				
255	}				
256					
257	// change 2 (WK23): GBA authentication/authorization extensions for content protection test				
258	/**				
259	* @desc Sets the B-TID in the BSM and BSF component.				
260	*/				
261	function f_main_ctrl_SetBootstrappingTransactionId(charstring p_btid) runs on BCastMainComp				
262	vc_BSM.start(f_bsm_setBtid(p_btid));				
263	vc_BSM.done;				
264	vc_BSF.start(f_bsf_setBtid(p_btid));				
265	vc_BSF.done;				
266	}				
267					
268	//change 2 (WK23): GBA authentication/authorization extensions for content protection tests				
269	/**				
270	* @desc This method generates the KS_(ext/int)_NAF and sets this keys the in BSM and BSF				
271	* @return the KS_(ext)_NAF or KS_(int)_NAF depending on the GBA type.				
272	* @verdict				
273	*/				
274	function f_main_ctrl_InitProtectionKeysAndValues() runs on BCastMainComponent return octets				
275	var octetstring ks_ext_NAF;				
276	var octetstring ks_int_NAF;				
277	vc_BSF.start(f_bsf_init(ks_ext_NAF, ks_int_NAF));				
278	vc_BSF.done;				
279	vc_BSM.start(f_bsm_setProtectionKeysAndValues(ks_ext_NAF));				

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

280	vc_BSM.done;			
281				
282	if (PX_GBA == e_gba_u) {			
283	// if UICC supports MBMS as well			
284	return ks_int_NAF;			
285	// TODO implements 'else if' for support of MBMS			
286	}	=	224	}
287	else {	+-		
288	// GBA_ME case: returns ks_NAF (equal to ke_ext_NAF)			
289	return ks_ext_NAF;			
290	}			
291				
292	}			
293				
294	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)			
295	/**			
296	* @desc This function sends any MIKEY message via UDP.			
297	* @param p_mikeyMessage			
298	* @verdict			
299	*/			
300	function f_main_ctrl_sendMikey(MikeyMessage p_mikeyMessage) runs on BCastMainComponent {			
301	vc_Mikey.start(f_send_MikeyMessage(p_mikeyMessage));			
302	vc_Mikey.done;			
303	}			
304				
305	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)			
306	/**			
307	* @desc This function sends any MIKEY message to a multicast IP address.			
308	* @param p_mikeyMessage			
309	* @verdict			
310	*/			
311	function f_main_ctrl_broadcastMikey(MikeyMessage p_mikeyMessage) runs on BCastMainComponent {			
312	vc_Mikey.start(f_broadcast_MikeyMessage(p_mikeyMessage));			
313	vc_Mikey.done;			
314	}	=	225	}
315	}		226	}
316			227	
317	group bcastDataMainFunctions {		228	group bcastDataMainFunctions {
318	// CR (WK 34/2008): Flute Update Signalling		229	// CR (WK 34/2008): Flute Update Signalling
319	/**		230	/**
320	*		231	*
321	* @desc This function trigger the file server to deliver files to the terminal.		232	* @desc This function trigger the file server to deliver files to the terminal.
322	* @param p_TransferId the transfer id		233	* @param p_TransferId the transfer id
323	* @param p_FluteUpdateID The Flute Instance ID of the Flute Session.		234	* @param p_FluteUpdateID The Flute Instance ID of the Flute Session.
324	* @param p_Sdp the used sdp		235	* @param p_Sdp the used sdp
325	* @param p_FileList the list of files which should delivered to the terminal		236	* @param p_FileList the list of files which should delivered to the terminal
326	* @verdict		237	* @verdict
327	* pass In case that the file transfer request was successful		238	* pass In case that the file transfer request was successful
328	* @verdict		239	* @verdict
329	* fail A wrong message was received.		240	* fail A wrong message was received.
330	* @verdict		241	* @verdict
331	* inconc A guard timer expires.		242	* inconc A guard timer expires.
332	*/		243	*/
333	function f_main_data_startFileTransfer(UInt p_TransferId,		244	function f_main_data_startFileTransfer(UInt p_TransferId,
334	UInt32 p_FluteUpdateID,		245	UInt32 p_FluteUpdateID,
335	charstring p_Sdp,		246	charstring p_Sdp,
336	TransferFileList p_FileList) runs on BCastMainComponent {		247	TransferFileList p_FileList) runs on BCastMainComponent {
337	vc_DATA.start (f_data_startFileTransfer(p_TransferId, p_FluteUpdateID, p_Sdp, p_FileList));		248	vc_DATA.start (f_data_startFileTransfer(p_TransferId, p_FluteUpdateID, p_Sdp, p_FileList));
338	vc_DATA.done;		249	vc_DATA.done;
339	}		250	}
340			251	
341	/**		252	/**
342	*		253	*
343	* @desc This function triggers the file server to stop a file transfer operation		254	* @desc This function triggers the file server to stop a file transfer operation
344	* @param p_TransferId the id of the transfer to be stopped		255	* @param p_TransferId the id of the transfer to be stopped
345	* @verdict		256	* @verdict
346	* pass in case that the stop streaming request was successful		257	* pass in case that the stop streaming request was successful
347	* @verdict		258	* @verdict
348	* fail A wrong message was received.		259	* fail A wrong message was received.
349	* @verdict		260	* @verdict
350	* inconc A guard timer expires.		261	* inconc A guard timer expires.

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

<pre>351 */ 352 function f_main_data_stopFileTransfer(UInt p_TransferId) runs on BCastMainComponent { 353 vc_DATA.start (f_data_stopFileTransfer(p_TransferId)); 354 vc_DATA.done; 355 } 356 357 // change 14 (WK18): dynamic setting of destination IP 358 /** 359 * 360 * @desc This function triggers the streaming server to stream a file. 361 * @param p_StreamId The stream id 362 * @param p_ContentList The list of files which should be streamed 363 * @param p_SDP The used SDP 364 * @param p_DstIp The destination IP address 365 * @verdict 366 * pass In case that the streaming request was successful 367 * @verdict 368 * fail A wrong message was received. 369 * @verdict 370 * inconc A guard timer expires. 371 */ 372 function f_main_data_startStreamingFile(373 UInt p_StreamId, 374 charstring p_FileName, 375 charstring p_SDP, 376 charstring p_DstIP) runs on BCastMainComponent { 377 vc_DATA.start (f_data_startStreaming(p_StreamId, p_FileName, p_SDP, p_DstIP)); 378 vc_DATA.done; 379 } 380 381 /** 382 * 383 * @desc This function trigger the streaming server to stop the streaming of a file 384 * @param p_StreamId the id of the stream which should stopped 385 * @verdict 386 * pass in case that the stop streaming request was successful 387 * @verdict 388 * fail A wrong message was received. 389 * @verdict 390 * inconc A guard timer expires. 391 */ 392 function f_main_data_stopStreaming(UInt p_StreamId) runs on BCastMainComponent { 393 vc_DATA.start (f_data_stopStreaming(p_StreamId)); 394 vc_DATA.done; 395 }</pre>		<pre>262 */ 263 function f_main_data_stopFileTransfer(UInt p_TransferId) runs on BCastMainComponent { 264 vc_DATA.start (f_data_stopFileTransfer(p_TransferId)); 265 vc_DATA.done; 266 } 267 268 // change 14 (WK18): dynamic setting of destination IP 269 /** 270 * 271 * @desc This function triggers the streaming server to stream a file. 272 * @param p_StreamId The stream id 273 * @param p_ContentList The list of files which should be streamed 274 * @param p_SDP The used SDP 275 * @param p_DstIp The destination IP address 276 * @verdict 277 * pass In case that the streaming request was successful 278 * @verdict 279 * fail A wrong message was received. 280 * @verdict 281 * inconc A guard timer expires. 282 */ 283 function f_main_data_startStreamingFile(284 UInt p_StreamId, 285 charstring p_FileName, 286 charstring p_SDP, 287 charstring p_DstIP) runs on BCastMainComponent { 288 vc_DATA.start (f_data_startStreaming(p_StreamId, p_FileName, p_SDP, p_DstIP)); 289 vc_DATA.done; 290 } 291 292 /** 293 * 294 * @desc This function trigger the streaming server to stop the streaming of a file 295 * @param p_StreamId the id of the stream which should stopped 296 * @verdict 297 * pass in case that the stop streaming request was successful 298 * @verdict 299 * fail A wrong message was received. 300 * @verdict 301 * inconc A guard timer expires. 302 */ 303 function f_main_data_stopStreaming(UInt p_StreamId) runs on BCastMainComponent { 304 vc_DATA.start (f_data_stopStreaming(p_StreamId)); 305 vc_DATA.done; 306 }</pre>
<pre>396 // change 3 (WK34): Service Protection: secure streaming service 397 /** 398 * 399 * @desc This function triggers the secure streaming server to stream a file. 400 * @param p_StreamId The stream id 401 * @param p_ContentList The list of files which should be streamed 402 * @param p_SDP The used SDP 403 * @param p_DstIp The destination IP address 404 * @verdict 405 * pass In case that the streaming request was successful 406 * @verdict 407 * fail A wrong message was received. 408 * @verdict 409 * inconc A guard timer expires. 410 */ 411 412 function f_main_data_secure_startStreamingFile(UInt p_StreamId, charstring p_FileName, char 413 vc_DATA.start(f_data_startStreaming(p_StreamId, p_FileName, p_SDP, PX_SRC_IP)); 414 vc_DATA.done; 415 // encrypting service streams out the content 416 vc_DATA.start(f_data_secure_startStreaming(p_DstIP, p_ssrc)); 417 vc_DATA.done; 418 } 419 420 // change 3 (WK34): Service Protection: secure streaming service 421 /**</pre>	+-	

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

422	<pre> * 423 * @desc This function triggers the sceure streaming server to stop the streaming of a file 424 * @param p_StreamId the id of the stream which should stopped 425 * @verdict 426 * pass in case that the stop streaming request was successful 427 * @verdict 428 * fail A wrong message was received. 429 * @verdict 430 * inconc A guard timer expires. 431 */ 432 function f_main_data_secure_stopStreaming(UInt p_StreamId) runs on BCastMainComponent { 433 vc_DATA.start(f_data_secure_stopStreaming()); 434 vc_DATA.done; 435 vc_DATA.start(f_data_stopStreaming(p_StreamId)); 436 vc_DATA.done; 437 } 438 439 function f_main_data_secure_setKeys(in ListOfSTKMKeys p_keys) runs on BCastMainComponent { 440 vc_DATA.start(f_data_secure_setKeys(p_keys)); 441 vc_DATA.done; 442 } 443 444 function f_main_data_secure_activateKey(in SecureStreamingKey p_key2Activate, in integer p_ 445 vc_DATA.start(f_data_secure_activateKey(p_key2Activate, p_keyNumberInList)); 446 vc_DATA.done; 447 }</pre>			
448	}	=	307	}
449			308	
450	group upperTesterMainFunctions {		309	group upperTesterMainFunctions {
451			310	
452	/**		311	/**
453	*		312	*
454	* @desc		313	* @desc
455	* This function creates a test component and starts a function that		314	* This function creates a test component and starts a function that
456	* sends a select language request from the upper tester component.		315	* sends a select language request from the upper tester component.
457	* @param		316	* @param
458	* p_langType the language type (audio or text)		317	* p_langType the language type (audio or text)
459	* @param		318	* @param
460	* p_langId the country id for the specific language		319	* p_langId the country id for the specific language
461	* @verdict		320	* @verdict
462	* pass The request was successful		321	* pass The request was successful
463	* @verdict		322	* @verdict
464	* fail A wrong message was received		323	* fail A wrong message was received
465	* @verdict		324	* @verdict
466	* inconc A guard timer expires		325	* inconc A guard timer expires
467	*/		326	*/
468	function f_main_ut_SelectLanguage(LibBCast_UpperTesterPrimitives_TypesAndValues.LanguageTyp		327	function f_main_ut_SelectLanguage(LibBCast_UpperTesterPrimitives_TypesAndValues.LanguageTyp
469			328	
470	vc_UTC.start (f_ut_sendSelectLanguageRequest(p_langType, p_langId));		329	vc_UTC.start (f_ut_sendSelectLanguageRequest(p_langType, p_langId));
471	vc_UTC.done;		330	vc_UTC.done;
472			331	
473	}		332	}
474			333	
475	/**		334	/**
476	*		335	*
477	* @desc		336	* @desc
478	* This function creates a test component and starts a function that		337	* This function creates a test component and starts a function that
479	* sends an update service guide request from the upper tester component.		338	* sends an update service guide request from the upper tester component.
480	* @verdict		339	* @verdict
481	* pass The request was successful		340	* pass The request was successful
482	* @verdict		341	* @verdict
483	* fail A wrong message was received		342	* fail A wrong message was received
484	* @verdict		343	* @verdict
485	* inconc A guard timer expires		344	* inconc A guard timer expires
486	*/		345	*/
487	function f_main_ut_UpdateServiceGuide() runs on BCastMainComponent {		346	function f_main_ut_UpdateServiceGuide() runs on BCastMainComponent {
488	vc_UTC.start (f_ut_sendUpdateServiceGuideRequest());		347	vc_UTC.start (f_ut_sendUpdateServiceGuideRequest());
489	vc_UTC.done;		348	vc_UTC.done;
490			349	
491	}		350	}
492			351	

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

493		352	
494	/**	353	/**
495	*	354	*
496	* @desc	355	* @desc
497	* This function creates a test component and starts a function that	356	* This function creates a test component and starts a function that
498	* sends a power on request from the upper tester component	357	* sends a power on request from the upper tester component
499	* @verdict	358	* @verdict
500	* pass The request was successful	359	* pass The request was successful
501	* @verdict	360	* @verdict
502	* fail A wrong message was received	361	* fail A wrong message was received
503	* @verdict	362	* @verdict
504	* inconc A guard timer expires	363	* inconc A guard timer expires
505	*/	364	*/
506	function f_main_ut_PowerOn() runs on BCastMainComponent {	365	function f_main_ut_PowerOn() runs on BCastMainComponent {
507	vc_UTC.start (f_ut_sendPowerOnRequest());	366	vc_UTC.start (f_ut_sendPowerOnRequest());
508	vc_UTC.done;	367	vc_UTC.done;
509	}	368	}
510		369	
511	/**	370	/**
512	*	371	*
513	* @desc	372	* @desc
514	* This function creates a test component and starts a function that	373	* This function creates a test component and starts a function that
515	* sends a power off request from the upper tester component	374	* sends a power off request from the upper tester component
516	* @verdict	375	* @verdict
517	* pass The request was successful	376	* pass The request was successful
518	* @verdict	377	* @verdict
519	* fail A wrong message was received	378	* fail A wrong message was received
520	* @verdict	379	* @verdict
521	* inconc A guard timer expires	380	* inconc A guard timer expires
522	*/	381	*/
523	function f_main_ut_PowerOff() runs on BCastMainComponent {	382	function f_main_ut_PowerOff() runs on BCastMainComponent {
524	vc_UTC.start (f_ut_sendPowerOffRequest());	383	vc_UTC.start (f_ut_sendPowerOffRequest());
525	vc_UTC.done;	384	vc_UTC.done;
526	}	385	}
527		386	
528	/**	387	/**
529	*	388	*
530	* @desc	389	* @desc
531	* This function creates a test component and starts a function that	390	* This function creates a test component and starts a function that
532	* sends a run BCAST application request from the upper tester component	391	* sends a run BCAST application request from the upper tester component
533	* @verdict	392	* @verdict
534	* pass The request was successful	393	* pass The request was successful
535	* @verdict	394	* @verdict
536	* fail A wrong message was received	395	* fail A wrong message was received
537	* @verdict	396	* @verdict
538	* inconc A guard timer expires	397	* inconc A guard timer expires
539	*/	398	*/
540	function f_main_ut_RunBCastApplication() runs on BCastMainComponent {	399	function f_main_ut_RunBCastApplication() runs on BCastMainComponent {
541	vc_UTC.start (f_ut_RunBCastApplication());	400	vc_UTC.start (f_ut_RunBCastApplication());
542	vc_UTC.done;	401	vc_UTC.done;
543	}	402	}
544		403	
545		404	
546	/**	405	/**
547	*	406	*
548	* @desc	407	* @desc
549	* This function creates a test component and starts a function that	408	* This function creates a test component and starts a function that
550	* sends a clear service guide cache request from the upper	409	* sends a clear service guide cache request from the upper
551	* tester component.	410	* tester component.
552	* @verdict	411	* @verdict
553	* pass in case the clear service guide cache request was	412	* pass in case the clear service guide cache request was
554	* successful.	413	* successful.
555	* @verdict	414	* @verdict
556	* inconc case that the request was not successful or the timer	415	* inconc case that the request was not successful or the timer
557	* runs out of time.	416	* runs out of time.
558	*/	417	*/
559	function f_main_ut_ClearServiceGuide() runs on BCastMainComponent {	418	function f_main_ut_ClearServiceGuide() runs on BCastMainComponent {
560	vc_UTC.start (f_ut_ClearServiceGuideCache());	419	vc_UTC.start (f_ut_ClearServiceGuideCache());
561	vc_UTC.done;	420	vc_UTC.done;
562		421	
563	}	422	}

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

564		423	
565	/**	424	/**
566	*	425	*
567	* @desc	426	* @desc
568	* This function creates a test component and starts a function that	427	* This function creates a test component and starts a function that
569	* sends a service check request from the upper tester component and expects	428	* sends a service check request from the upper tester component and expects
570	* a success response back.	429	* a success response back.
571	* @param	430	* @param
572	* p_content The name of the content displayed to the user, e.g., program 1	431	* p_content The name of the content displayed to the user, e.g., program 1
573	* @verdict	432	* @verdict
574	* pass in case service check request was successful.	433	* pass in case service check request was successful.
575	* @verdict	434	* @verdict
576	* inconc case that the request was not successful or the timer	435	* inconc case that the request was not successful or the timer
577	* runs out of time.	436	* runs out of time.
578	*/	437	*/
579	function f_main_ut_CheckService(charstring p_content) runs on BCastMainComponent {	438	function f_main_ut_CheckService(charstring p_content) runs on BCastMainComponent {
580	vc_UTC.start(f_ut_CheckService(p_content));	439	vc_UTC.start(f_ut_CheckService(p_content));
581	vc_UTC.done;	440	vc_UTC.done;
582		441	
583	}	442	}
584		443	
585	/**	444	/**
586	*	445	*
587	* @desc	446	* @desc
588	* This function creates a test component and starts a function that	447	* This function creates a test component and starts a function that
589	* sends a Use Service Request from the upper tester component and expects a	448	* sends a Use Service Request from the upper tester component and expects a
590	* success response back.	449	* success response back.
591	* @param	450	* @param
592	* p_service The name of the service to be used (e.g., viewed) by	451	* p_service The name of the service to be used (e.g., viewed) by
593	* the user, e.g., TV Channel 1	452	* the user, e.g., TV Channel 1
594	* @verdict	453	* @verdict
595	* pass in case service check request was successful.	454	* pass in case service check request was successful.
596	* @verdict	455	* @verdict
597	* inconc case that the request was not successful or the timer	456	* inconc case that the request was not successful or the timer
598	* runs out of time.	457	* runs out of time.
599	*/	458	*/
600	function f_main_ut_UseService(charstring p_service) runs on BCastMainComponent {	459	function f_main_ut_UseService(charstring p_service) runs on BCastMainComponent {
601	vc_UTC.start(f_ut_UseService(p_service));	460	vc_UTC.start(f_ut_UseService(p_service));
602	vc_UTC.done;	461	vc_UTC.done;
603		462	
604	}	463	}
605		464	
606	/**	465	/**
607	*	466	*
608	* @desc	467	* @desc
609	* This function creates a test component and starts a function that	468	* This function creates a test component and starts a function that
610	* sends a Get Service Guide Request from the upper tester component and expects a	469	* sends a Get Service Guide Request from the upper tester component and expects a
611	* success response back.	470	* success response back.
612	* @param	471	* @param
613	* p_ServiceGuideIdentifier The name of the service guide to retrieved	472	* p_ServiceGuideIdentifier The name of the service guide to retrieved
614	* @verdict	473	* @verdict
615	* pass in case service check request was successful.	474	* pass in case service check request was successful.
616	* @verdict	475	* @verdict
617	* inconc case that the request was not successful or the timer	476	* inconc case that the request was not successful or the timer
618	* runs out of time.	477	* runs out of time.
619	*/	478	*/
620	function f_main_ut_GetServiceGuide(charstring p_ServiceGuideIdentifier) runs on BCastMainCo	479	function f_main_ut_GetServiceGuide(charstring p_ServiceGuideIdentifier) runs on BCastMainCo
621	vc_UTC.start(f_ut_GetServiceGuide(p_ServiceGuideIdentifier));	480	vc_UTC.start(f_ut_GetServiceGuide(p_ServiceGuideIdentifier));
622	vc_UTC.done;	481	vc_UTC.done;
623		482	
624	}	483	}
625		484	
626	/**	485	/**
627	*	486	*
628	* @desc	487	* @desc
629	* This function creates a test component and starts a function thatsends a select serv	488	* This function creates a test component and starts a function thatsends a select serv
630	* upper tester.	489	* upper tester.
631	* @param	490	* @param
632	* p_service The name of the service to be selected by the user, e.g., TV Channel 1	491	* p_service The name of the service to be selected by the user, e.g., TV Channel 1
633	* @verdict	492	* @verdict
634	* pass The request was successful	493	* pass The request was successful

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

635	* @verdict	494	* @verdict
636	* fail A wrong message was received	495	* fail A wrong message was received
637	* @verdict	496	* @verdict
638	* inconc A guard timer expires	497	* inconc A guard timer expires
639	*/	498	*/
640	function f_main_ut_SelectService(charstring p_service) runs on BCastMainComponent {	499	function f_main_ut_SelectService(charstring p_service) runs on BCastMainComponent {
641	vc_UTC.start(f_ut_SelectService(p_service));	500	vc_UTC.start(f_ut_SelectService(p_service));
642	vc_UTC.done;	501	vc_UTC.done;
643		502	
644	}	503	}
645		504	
646	/**	505	/**
647	*	506	*
648	* @desc	507	* @desc
649	* This function creates a test component and starts a function that	508	* This function creates a test component and starts a function that
650	* sends a check content request from the upper tester component.	509	* sends a check content request from the upper tester component.
651	* @param	510	* @param
652	* p_content The name of the content displayed to the user	511	* p_content The name of the content displayed to the user
653	* @param	512	* @param
654	* p_start Start of broadcasting of content displayed to the	513	* p_start Start of broadcasting of content displayed to the
655	* user, e.g., Program 1	514	* user, e.g., Program 1
656	* @param	515	* @param
657	* p_end End of broadcasting of content displayed to the user	516	* p_end End of broadcasting of content displayed to the user
658	* @verdict	517	* @verdict
659	* pass The request was successful	518	* pass The request was successful
660	* @verdict	519	* @verdict
661	* fail A wrong message was received	520	* fail A wrong message was received
662	* @verdict	521	* @verdict
663	* inconc A guard timer expires	522	* inconc A guard timer expires
664	*/	523	*/
665	function f_main_ut_CheckContent(charstring p_content, template DateTime p_start, template I	524	function f_main_ut_CheckContent(charstring p_content, template DateTime p_start, template I
666	vc_UTC.start(f_ut_CheckContent(p_content, p_start, p_end));	525	vc_UTC.start(f_ut_CheckContent(p_content, p_start, p_end));
667	vc_UTC.done;	526	vc_UTC.done;
668	}	527	}
669		528	
670	/**	529	/**
671	*	530	*
672	* @desc	531	* @desc
673	* This function creates a test component and starts a function that	532	* This function creates a test component and starts a function that
674	* sends a check interactivity request from the upper tester component.	533	* sends a check interactivity request from the upper tester component.
675	* @param	534	* @param
676	* p_interactivity Name of the interactivity displayed to the user, e.g., vote	535	* p_interactivity Name of the interactivity displayed to the user, e.g., vote
677	* @verdict	536	* @verdict
678	* pass The request was successful	537	* pass The request was successful
679	* @verdict	538	* @verdict
680	* fail A wrong message was received	539	* fail A wrong message was received
681	* @verdict	540	* @verdict
682	* inconc A guard timer expires	541	* inconc A guard timer expires
683	*/	542	*/
684	function f_main_ut_CheckInteractivity(charstring p_interactivity) runs on BCastMainComponer	543	function f_main_ut_CheckInteractivity(charstring p_interactivity) runs on BCastMainComponer
685	vc_UTC.start(f_ut_CheckInteractivity(p_interactivity));	544	vc_UTC.start(f_ut_CheckInteractivity(p_interactivity));
686	vc_UTC.done;	545	vc_UTC.done;
687	}	546	}
688		547	
689	/**	548	/**
690	*	549	*
691	* @desc	550	* @desc
692	* This function creates a test component and starts a function that	551	* This function creates a test component and starts a function that
693	* sends a select interactivity request from the upper tester component.	552	* sends a select interactivity request from the upper tester component.
694	* @param	553	* @param
695	* p_interactivity Name of the interactivity to be selected to the user, e.g., vote	554	* p_interactivity Name of the interactivity to be selected to the user, e.g., vote
696	* @verdict	555	* @verdict
697	* pass The request was successful	556	* pass The request was successful
698	* @verdict	557	* @verdict
699	* fail A wrong message was received	558	* fail A wrong message was received
700	* @verdict	559	* @verdict
701	* inconc A guard timer expires	560	* inconc A guard timer expires
702	*/	561	*/
703	function f_main_ut_SelectInteractivity(charstring p_interactivity) runs on BCastMainComponer	562	function f_main_ut_SelectInteractivity(charstring p_interactivity) runs on BCastMainComponer
704	vc_UTC.start(f_ut_SelectInteractivity(p_interactivity));	563	vc_UTC.start(f_ut_SelectInteractivity(p_interactivity));
705	vc_UTC.done;	564	vc_UTC.done;

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

706	}	565	}
707		566	
708	/**	567	/**
709	*	568	*
710	* @desc	569	* @desc
711	* This function creates a test component and starts a function that	570	* This function creates a test component and starts a function that
712	* sends a check interactivity choices request from the upper tester component.	571	* sends a check interactivity choices request from the upper tester component.
713	* @param	572	* @param
714	* p_choices Name of the interactivity choices displayed to the	573	* p_choices Name of the interactivity choices displayed to the
715	* user	574	* user
716	* @verdict	575	* @verdict
717	* pass The request was successful	576	* pass The request was successful
718	* @verdict	577	* @verdict
719	* fail A wrong message was received	578	* fail A wrong message was received
720	* @verdict	579	* @verdict
721	* inconc A guard timer expires	580	* inconc A guard timer expires
722	*/	581	*/
723	function f_main_ut_CheckInteractivityChoices(LanguageStringList p_ChoiceTexts) runs on BCas	582	function f_main_ut_CheckInteractivityChoices(LanguageStringList p_ChoiceTexts) runs on BCas
724	vc_UTC.start(f_ut_CheckInteractivityChoice(p_ChoiceTexts));	583	vc_UTC.start(f_ut_CheckInteractivityChoice(p_ChoiceTexts));
725	vc_UTC.done;	584	vc_UTC.done;
726	}	585	}
727		586	
728	/**	587	/**
729	*	588	*
730	* @desc	589	* @desc
731	* This function creates a test component and starts a function that	590	* This function creates a test component and starts a function that
732	* sends a select interactivity coice request to	591	* sends a select interactivity coice request to
733	* the upper tester.	592	* the upper tester.
734	* @param	593	* @param
735	* p_choice Name of the interactivity choice to be selected by the	594	* p_choice Name of the interactivity choice to be selected by the
736	* user, e.g., choice a	595	* user, e.g., choice a
737	* @verdict	596	* @verdict
738	* pass The request was successful	597	* pass The request was successful
739	* @verdict	598	* @verdict
740	* fail A wrong message was received	599	* fail A wrong message was received
741	* @verdict	600	* @verdict
742	* inconc A guard timer expires	601	* inconc A guard timer expires
743	*/	602	*/
744	function f_main_ut_SelectInteractivityChoice(charstring p_choice) runs on BCastMainComponen	603	function f_main_ut_SelectInteractivityChoice(charstring p_choice) runs on BCastMainComponen
745	vc_UTC.start(f_ut_SelectInteractivityChoice(p_choice));	604	vc_UTC.start(f_ut_SelectInteractivityChoice(p_choice));
746	vc_UTC.done;	605	vc_UTC.done;
747	}	606	}
748		607	
749	/**	608	/**
750	*	609	*
751	* @desc	610	* @desc
752	* This function creates a test component and starts a function that	611	* This function creates a test component and starts a function that
753	* sends a check audio language request from the upper tester component.	612	* sends a check audio language request from the upper tester component.
754	* @param	613	* @param
755	* p_content The identifer of the content displayed to the user, e.g., Program 1	614	* p_content The identifer of the content displayed to the user, e.g., Program 1
756	* @param	615	* @param
757	* p_type Type of language setting to be changed by the user	616	* p_type Type of language setting to be changed by the user
758	* @param	617	* @param
759	* p_lang Language that the user shall hear	618	* p_lang Language that the user shall hear
760	* @verdict	619	* @verdict
761	* pass The request was successful	620	* pass The request was successful
762	* @verdict	621	* @verdict
763	* fail A wrong message was received	622	* fail A wrong message was received
764	* @verdict	623	* @verdict
765	* inconc A guard timer expires	624	* inconc A guard timer expires
766	*/	625	*/
767	function f_main_ut_CheckLanguage(charstring p_content, LibBCast_UpperTesterPrimitives_Types	626	function f_main_ut_CheckLanguage(charstring p_content, LibBCast_UpperTesterPrimitives_Types
768	vc_UTC.start(f_ut_CheckLanguage(p_content, p_type, p_lang));	627	vc_UTC.start(f_ut_CheckLanguage(p_content, p_type, p_lang));
769	vc_UTC.done;	628	vc_UTC.done;
770	}	629	}
771		630	
772	/**	631	/**
773	*	632	*
774	* @desc	633	* @desc
775	* This function creates a test component and starts a function that	634	* This function creates a test component and starts a function that
776	* sends a check browser request from the upper tester component.	635	* sends a check browser request from the upper tester component.

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

777	* @param	636	* @param
778	* p_xhtml Reference or description of the content of the xhtml	637	* p_xhtml Reference or description of the content of the xhtml
779	* file to be displayed to the user	638	* file to be displayed to the user
780	* @verdict	639	* @verdict
781	* pass The request was successful	640	* pass The request was successful
782	* @verdict	641	* @verdict
783	* fail A wrong message was received	642	* fail A wrong message was received
784	* @verdict	643	* @verdict
785	* inconc A guard timer expires	644	* inconc A guard timer expires
786	*/	645	*/
787	function f_main_ut_CheckBrowser(charstring p_xhtml) runs on BCastMainComponent {	646	function f_main_ut_CheckBrowser(charstring p_xhtml) runs on BCastMainComponent {
788	vc_UTC.start(f_ut_CheckBrowser(p_xhtml));	647	vc_UTC.start(f_ut_CheckBrowser(p_xhtml));
789	vc_UTC.done;	648	vc_UTC.done;
790	}	649	}
791		650	
792	/**	651	/**
793	*	652	*
794	* @desc	653	* @desc
795	* This function creates a test component and starts a function that	654	* This function creates a test component and starts a function that
796	* sends a check preview request from the upper tester component.	655	* sends a check preview request from the upper tester component.
797	* @param	656	* @param
798	* p_data Reference to icon or text to be displayed to the user, e.g., TV Channel logo	657	* p_data Reference to icon or text to be displayed to the user, e.g., TV Channel logo
799	* @verdict	658	* @verdict
800	* pass The request was successful	659	* pass The request was successful
801	* @verdict	660	* @verdict
802	* fail A wrong message was received	661	* fail A wrong message was received
803	* @verdict	662	* @verdict
804	* inconc A guard timer expires	663	* inconc A guard timer expires
805	*/	664	*/
806	function f_main_ut_CheckPreview(charstring p_data) runs on BCastMainComponent {	665	function f_main_ut_CheckPreview(charstring p_data) runs on BCastMainComponent {
807	vc_UTC.start(f_ut_CheckPreview(p_data));	666	vc_UTC.start(f_ut_CheckPreview(p_data));
808	vc_UTC.done;	667	vc_UTC.done;
809	}	668	}
810		669	
811	/**	670	/**
812	*	671	*
813	* @desc	672	* @desc
814	* This function creates a test component and starts a function that	673	* This function creates a test component and starts a function that
815	* sends a select file request from the upper tester component.	674	* sends a select file request from the upper tester component.
816	* @param	675	* @param
817	* p_file File name to be selected by the user, e.g., foo.c	676	* p_file File name to be selected by the user, e.g., foo.c
818	* @verdict	677	* @verdict
819	* pass The request was successful	678	* pass The request was successful
820	* @verdict	679	* @verdict
821	* fail A wrong message was received	680	* fail A wrong message was received
822	* @verdict	681	* @verdict
823	* inconc A guard timer expires	682	* inconc A guard timer expires
824	*/	683	*/
825	function f_main_ut_SelectFile(charstring p_file) runs on BCastMainComponent {	684	function f_main_ut_SelectFile(charstring p_file) runs on BCastMainComponent {
826	vc_UTC.start(f_ut_SelectFile(p_file));	685	vc_UTC.start(f_ut_SelectFile(p_file));
827	vc_UTC.done;	686	vc_UTC.done;
828	}	687	}
829		688	
830	/**	689	/**
831	*	690	*
832	* @desc	691	* @desc
833	* This function creates a test component and starts a function that	692	* This function creates a test component and starts a function that
834	* sends a check file request from the upper tester component.	693	* sends a check file request from the upper tester component.
835	* @param	694	* @param
836	* p_file Reference to file name or description of its content	695	* p_file Reference to file name or description of its content
837	* to be displayed to the user, e.g., foo.c	696	* to be displayed to the user, e.g., foo.c
838	* @verdict	697	* @verdict
839	* pass The request was successful	698	* pass The request was successful
840	* @verdict	699	* @verdict
841	* fail A wrong message was received	700	* fail A wrong message was received
842	* @verdict	701	* @verdict
843	* inconc A guard timer expires	702	* inconc A guard timer expires
844	*/	703	*/
845	function f_main_ut_CheckFile(charstring p_file) runs on BCastMainComponent {	704	function f_main_ut_CheckFile(charstring p_file) runs on BCastMainComponent {
846	vc_UTC.start(f_ut_CheckFile(p_file));	705	vc_UTC.start(f_ut_CheckFile(p_file));
847	vc_UTC.done;	706	vc_UTC.done;

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

848	}	707	}
849		708	
850	/**	709	/**
851	*	710	*
852	* @desc	711	* @desc
853	* This function creates a test component and starts a function that	712	* This function creates a test component and starts a function that
854	* sends a check file received request from the upper tester component.	713	* sends a check file received request from the upper tester component.
855	* @param	714	* @param
856	* p_file File name or description of its content that is to be received	715	* p_file File name or description of its content that is to be received
857	* by the user, e.g., foo.c	716	* by the user, e.g., foo.c
858	* @verdict	717	* @verdict
859	* pass The request was successful	718	* pass The request was successful
860	* @verdict	719	* @verdict
861	* fail A wrong message was received	720	* fail A wrong message was received
862	* @verdict	721	* @verdict
863	* inconc A guard timer expires	722	* inconc A guard timer expires
864	*/	723	*/
865	function f_main_ut_CheckFileReceived(charstring p_file) runs on BCastMainComponent {	724	function f_main_ut_CheckFileReceived(charstring p_file) runs on BCastMainComponent {
866	vc_UTC.start(f_ut_CheckFileReceived(p_file));	725	vc_UTC.start(f_ut_CheckFileReceived(p_file));
867	vc_UTC.done;	726	vc_UTC.done;
868	}	727	}
869		728	
870	/**	729	/**
871	*	730	*
872	* @desc	731	* @desc
873	* This function creates a test component and starts a function that	732	* This function creates a test component and starts a function that
874	* sends a check video request from the upper tester component.	733	* sends a check video request from the upper tester component.
875	* @param	734	* @param
876	* p_video Reference to video name or description of its content	735	* p_video Reference to video name or description of its content
877	* to be displayed to the user, e.g., starwars	736	* to be displayed to the user, e.g., starwars
878	* @verdict	737	* @verdict
879	* pass The request was successful	738	* pass The request was successful
880	* @verdict	739	* @verdict
881	* fail A wrong message was received	740	* fail A wrong message was received
882	* @verdict	741	* @verdict
883	* inconc A guard timer expires	742	* inconc A guard timer expires
884	*/	743	*/
885	function f_main_ut_CheckVideo(charstring p_video) runs on BCastMainComponent {	744	function f_main_ut_CheckVideo(charstring p_video) runs on BCastMainComponent {
886	vc_UTC.start(f_ut_CheckVideo(p_video));	745	vc_UTC.start(f_ut_CheckVideo(p_video));
887	vc_UTC.done;	746	vc_UTC.done;
888	}	747	}
889		748	
890	/**	749	/**
891	*	750	*
892	* @desc	751	* @desc
893	* This function creates a test component and starts a function that	752	* This function creates a test component and starts a function that
894	* sends a check purchase info request from the upper tester component.	753	* sends a check purchase info request from the upper tester component.
895	* @param	754	* @param
896	* p_info The purchase info content to be displayed to the user	755	* p_info The purchase info content to be displayed to the user
897	* @verdict	756	* @verdict
898	* pass The request was successful	757	* pass The request was successful
899	* @verdict	758	* @verdict
900	* fail A wrong message was received	759	* fail A wrong message was received
901	* @verdict	760	* @verdict
902	* inconc A guard timer expires	761	* inconc A guard timer expires
903	*/	762	*/
904	function f_main_ut_CheckPurchaseInfo(charstring p_info) runs on BCastMainComponent {	763	function f_main_ut_CheckPurchaseInfo(charstring p_info) runs on BCastMainComponent {
905	vc_UTC.start(f_ut_CheckPurchaseInfo(p_info));	764	vc_UTC.start(f_ut_CheckPurchaseInfo(p_info));
906	vc_UTC.done;	765	vc_UTC.done;
907	}	766	}
908		767	
909	/**	768	/**
910	*	769	*
911	* @desc	770	* @desc
912	* This function creates a test component and starts a function that	771	* This function creates a test component and starts a function that
913	* sends a purchase service request from the upper tester component.	772	* sends a purchase service request from the upper tester component.
914	* @param	773	* @param
915	* p_service The name of the service to be purchased by the user, e.g., TV Channel 1	774	* p_service The name of the service to be purchased by the user, e.g., TV Channel 1
916	* @verdict	775	* @verdict
917	* pass The request was successful	776	* pass The request was successful
918	* @verdict	777	* @verdict

Datei: AtsBCast_Main_Functions.ttcn (Fortsetzung)

919	* fail A wrong message was received	778	* fail A wrong message was received
920	* @verdict	779	* @verdict
921	* inconc A guard timer expires	780	* inconc A guard timer expires
922	*/	781	*/
923	function f_main_ut_PurchaseService(charstring p_service) runs on BCastMainComponent {	782	function f_main_ut_PurchaseService(charstring p_service) runs on BCastMainComponent {
924	vc_UTC.start(f_ut_CheckPurchaseInfo(p_service));	783	vc_UTC.start(f_ut_CheckPurchaseInfo(p_service));
925	vc_UTC.done;	784	vc_UTC.done;
926	}	785	}
927		786	
928		787	
929		788	
930	}	789	}
931	}	790	}

Datei: AtsBCast_ModuleParameters.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies ATS specific module parameters and their default values 9 * which can be still modified in their value as late as just prior to 10 * test case execution. The parameters in this file OMA BCAST specific. 11 * Module parameters realize so called PIXIT. 12 */ 13 module AtsBCast_ModuleParameters { 14 import from LibBCast_UpperTesterPrimitives_TypesAndValues all; 15 import from LibCommon_TextStrings { 16 const c_CRLF 17 } 18 // change 1 (WK18): for content protection tests 19 // hint: for GBS authentication parameter 20 import from LibCommon_DataStrings { 21 group bitStringSubTypes 22 } 23 24 import from LibCommon_DataStrings_Extended { 25 group hexStringSubTypes 26 } 27 28 // change 2 (WK23): GBA authentication/authorization extensions for content protection tests 29 import from LibCommon_GBA_BSF_TypesAndValues { 30 group GBA 31 } 32 33 group testCaseSelectionSwitches { 34 /** 35 * 36 * @desc Specifies to execute all the BCAST test cases 37 */ 38 modulepar boolean PX_ALL_TCS := true; 39 40 /** 41 * 42 * @desc Specifies to execute all the Service Provisioning test cases 43 */ 44 modulepar boolean PX_ALL_SP_TCS := false; 45 46 /** 47 * 48 * @desc Specifies to execute all the Service Guide test cases 49 */ 50 modulepar boolean PX_ALL_SG_TCS := false; 51 52 /** 53 * 54 * @desc Specifies to execute all the File and Stream Distribution test cases 55 */ 56 modulepar boolean PX_ALL_FD_TCS := false; 57 58 /** 59 * 60 * @desc Specifies all Service Interactcion test cases 61 */ 62 modulepar boolean PX_ALL_SI_TCS := false; 63 64 /** 65 * 66 * @desc Specifies to execute all the Content Protection test cases 67 */ 68 modulepar boolean PX_ALL_CP_TCS := false; 69 70 // change 3 (WK34): Service Protection: secure streaming service 71 /** 72 *</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies ATS specific module parameters and their default values 9 * which can be still modified in their value as late as just prior to 10 * test case execution. The parameters in this file OMA BCAST specific. 11 * Module parameters realize so called PIXIT. 12 */ 13 module AtsBCast_ModuleParameters { 14 import from LibBCast_UpperTesterPrimitives_TypesAndValues all; 15 import from LibCommon_TextStrings { 16 const c_CRLF 17 } 18 // change 1 (WK18): for content protection tests 19 // hint: for GBS authentication parameter 20 import from LibCommon_DataStrings { 21 group bitStringSubTypes 22 } 23 24 import from LibCommon_DataStrings_Extended { 25 group hexStringSubTypes 26 } 27 28 group testCaseSelectionSwitches { 29 /** 30 * 31 * @desc Specifies to execute all the BCAST test cases 32 */ 33 modulepar boolean PX_ALL_TCS := true; 34 35 /** 36 * 37 * @desc Specifies to execute all the Service Provisioning test cases 38 */ 39 modulepar boolean PX_ALL_SP_TCS := false; 40 41 /** 42 * 43 * @desc Specifies to execute all the Service Guide test cases 44 */ 45 modulepar boolean PX_ALL_SG_TCS := false; 46 47 /** 48 * 49 * @desc Specifies to execute all the File and Stream Distribution test cases 50 */ 51 modulepar boolean PX_ALL_FD_TCS := false; 52 53 /** 54 * 55 * @desc Specifies all Service Interactcion test cases 56 */ 57 modulepar boolean PX_ALL_SI_TCS := false; 58 59 /** 60 * 61 * @desc Specifies to execute all the Content Protection test cases 62 */ 63 modulepar boolean PX_ALL_CP_TCS := false;</pre>
	+-	
	=	
	+-	

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

72	<pre> * @desc Specifies to execute all the Service Protection test cases */ modulepar boolean PX_ALL_SProt_TCS := false;</pre>				
75	<pre>} // end group testCaseSelectionSwitches</pre>	=	64	<pre>} // end group testCaseSelectionSwitches</pre>	
76			65		
77	<pre>group terminalUserApi {</pre>		66	<pre>group terminalUserApi {</pre>	
78			67		
79	<pre> /**</pre>		68	<pre> /**</pre>	
80	<pre> *</pre>		69	<pre> *</pre>	
81	<pre> * @desc Indicates if BCAST application under test needs (manual) operator</pre>		70	<pre> * @desc Indicates if BCAST application under test needs (manual) operator</pre>	
82	<pre> * interaction in order get service guide content displayed. If set to false</pre>		71	<pre> * interaction in order get service guide content displayed. If set to false</pre>	
83	<pre> * the application is assumed to display the service guide automatically.</pre>		72	<pre> * the application is assumed to display the service guide automatically.</pre>	
84	<pre> */</pre>		73	<pre> */</pre>	
85	<pre>modulepar boolean PX_GET_SG_DISPLAY_MANUALLY := false;</pre>		74	<pre>modulepar boolean PX_GET_SG_DISPLAY_MANUALLY := false;</pre>	
86			75		
87	<pre>/**</pre>		76	<pre>/**</pre>	
88	<pre>*</pre>		77	<pre>*</pre>	
89	<pre>* @desc Indicates if BCAST application under test needs (manual) operator</pre>		78	<pre>* @desc Indicates if BCAST application under test needs (manual) operator</pre>	
90	<pre>* interaction in order get a service guide update displayed. If set to false</pre>		79	<pre>* interaction in order get a service guide update displayed. If set to false</pre>	
91	<pre>* the application is assumed to display the service guide update automatically.</pre>		80	<pre>* the application is assumed to display the service guide update automatically.</pre>	
92	<pre>*/</pre>		81	<pre>*/</pre>	
93	<pre>modulepar boolean PX_GET_SG_UPDATE_MANUALLY := false;</pre>		82	<pre>modulepar boolean PX_GET_SG_UPDATE_MANUALLY := false;</pre>	
94			83		
95	<pre>} // end group terminalUserApi</pre>		84	<pre>} // end group terminalUserApi</pre>	
96			85		
97	<pre>group ServiceGuideIds {</pre>		86	<pre>group ServiceGuideIds {</pre>	
98			87		
99	<pre> group ServiceFragment {</pre>		88	<pre> group ServiceFragment {</pre>	
100	<pre> /**</pre>		89	<pre> /**</pre>	
101	<pre> *</pre>		90	<pre> *</pre>	
102	<pre> * @desc</pre>		91	<pre> * @desc</pre>	
103	<pre> * Specifies the globally unique URI for the Service fragment</pre>		92	<pre> * Specifies the globally unique URI for the Service fragment</pre>	
104	<pre> */</pre>		93	<pre> */</pre>	
105	<pre>modulepar charstring PX_SGDU_SERVICE_ID :=</pre>		94	<pre>modulepar charstring PX_SGDU_SERVICE_ID :=</pre>	
106	<pre> "bcast://bcast.testingtech.com/service/service_id";</pre>		95	<pre> "bcast://bcast.testingtech.com/service/service_id";</pre>	
107			96		
108	<pre>}</pre>		97	<pre>}</pre>	
109			98		
110	<pre>group ContentFragment {</pre>		99	<pre>group ContentFragment {</pre>	
111	<pre> /**</pre>		100	<pre> /**</pre>	
112	<pre> *</pre>		101	<pre> *</pre>	
113	<pre> * @desc</pre>		102	<pre> * @desc</pre>	
114	<pre> * Specifies the globally unique URI for the Content fragment.</pre>		103	<pre> * Specifies the globally unique URI for the Content fragment.</pre>	
115	<pre> */</pre>		104	<pre> */</pre>	
116	<pre>modulepar charstring PX_SGDU_CONTENT_ID_PROG_1 :=</pre>		105	<pre>modulepar charstring PX_SGDU_CONTENT_ID_PROG_1 :=</pre>	
117	<pre> "bcast://bcast.testingtech.com/content/content_id_1";</pre>		106	<pre> "bcast://bcast.testingtech.com/content/content_id_1";</pre>	
118			107		
119	<pre>/**</pre>		108	<pre>/**</pre>	
120	<pre>*</pre>		109	<pre>*</pre>	
121	<pre>* @desc</pre>		110	<pre>* @desc</pre>	
122	<pre>* Specifies the globally unique URI for the Content fragment.</pre>		111	<pre>* Specifies the globally unique URI for the Content fragment.</pre>	
123	<pre>*/</pre>		112	<pre>*/</pre>	
124	<pre>modulepar charstring PX_SGDU_CONTENT_ID_PROG_2 :=</pre>		113	<pre>modulepar charstring PX_SGDU_CONTENT_ID_PROG_2 :=</pre>	
125	<pre> "bcast://bcast.testingtech.com/content/content_id_2";</pre>		114	<pre> "bcast://bcast.testingtech.com/content/content_id_2";</pre>	
126			115		
127	<pre>}</pre>		116	<pre>}</pre>	
128			117		
129	<pre>group ScheduleFragment {</pre>		118	<pre>group ScheduleFragment {</pre>	
130	<pre> /**</pre>		119	<pre> /**</pre>	
131	<pre> *</pre>		120	<pre> *</pre>	
132	<pre> * @desc</pre>		121	<pre> * @desc</pre>	
133	<pre> * Specifies the globally unique URI for the Schedule fragment.</pre>		122	<pre> * Specifies the globally unique URI for the Schedule fragment.</pre>	
134	<pre> */</pre>		123	<pre> */</pre>	
135	<pre>modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_1 :=</pre>		124	<pre>modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_1 :=</pre>	
136	<pre> "bcast://bcast.testingtech.com/schedule/schedule_id_1";</pre>		125	<pre> "bcast://bcast.testingtech.com/schedule/schedule_id_1";</pre>	
137			126		
138	<pre>/**</pre>		127	<pre>/**</pre>	
139	<pre>*</pre>		128	<pre>*</pre>	
140	<pre>* @desc</pre>		129	<pre>* @desc</pre>	
141	<pre>* Specifies the globally unique URI for the Schedule fragment.</pre>		130	<pre>* Specifies the globally unique URI for the Schedule fragment.</pre>	
142	<pre>*/</pre>		131	<pre>*/</pre>	

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

143	modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_2 :=	132	modulepar charstring PX_SGDU_SCHEDULE_ID_PROG_2 :=
144	"bcast://bcast.testingtech.com/schedule/schedule_id_2";	133	"bcast://bcast.testingtech.com/schedule/schedule_id_2";
145	}	134	}
146		135	
147	group AccessFragment {	136	group AccessFragment {
148	/**	137	/**
149	*	138	*
150	* @desc	139	* @desc
151	* Specifies the globally unique URI for the Access fragment	140	* Specifies the globally unique URI for the Access fragment
152	*/	141	*/
153	modulepar charstring PX_SGDU_ACCESS_ID_PROG_1 :=	142	modulepar charstring PX_SGDU_ACCESS_ID_PROG_1 :=
154	"bcast://bcast.testingtech.com/access/access_id_1";	143	"bcast://bcast.testingtech.com/access/access_id_1";
155		144	
156	/**	145	/**
157	*	146	*
158	* @desc	147	* @desc
159	* Specifies the globally unique URI for the Access fragment	148	* Specifies the globally unique URI for the Access fragment
160	*/	149	*/
161	modulepar charstring PX_SGDU_ACCESS_ID_PROG_2 :=	150	modulepar charstring PX_SGDU_ACCESS_ID_PROG_2 :=
162	"bcast://bcast.testingtech.com/access/access_id_2";	151	"bcast://bcast.testingtech.com/access/access_id_2";
163	}	152	}
164		153	
165	group PreviewDataFragment {	154	group PreviewDataFragment {
166	/**	155	/**
167	*	156	*
168	* @desc	157	* @desc
169	* Specifies the globally unique URI for the PreviewData fragment	158	* Specifies the globally unique URI for the PreviewData fragment
170	*/	159	*/
171	modulepar charstring PX_SGDU_PREVIEW_DATA_ID :=	160	modulepar charstring PX_SGDU_PREVIEW_DATA_ID :=
172	"bcast://bcast.testingtech.com/previewData/previewData_id_1";	161	"bcast://bcast.testingtech.com/previewData/previewData_id_1";
173	}	162	}
174		163	
175	group PurchaseItemFragment {	164	group PurchaseItemFragment {
176	/**	165	/**
177	*	166	*
178	* @desc	167	* @desc
179	* Specifies the globally unique URI for the PurchaseItem fragment	168	* Specifies the globally unique URI for the PurchaseItem fragment
180	*/	169	*/
181	modulepar charstring PX_SGDU_PURCHASE_ITEM_ID :=	170	modulepar charstring PX_SGDU_PURCHASE_ITEM_ID :=
182	"bcast://bcast.testingtech.com/purchaseItem/purchaseItem_id_1";	171	"bcast://bcast.testingtech.com/purchaseItem/purchaseItem_id_1";
183	}	172	}
184		173	
185	group PurchaseDataFragment {	174	group PurchaseDataFragment {
186	/**	175	/**
187	*	176	*
188	* @desc	177	* @desc
189	* Specifies the globally unique URI for the PurchaseData fragment	178	* Specifies the globally unique URI for the PurchaseData fragment
190	*/	179	*/
191	modulepar charstring PX_SGDU_PURCHASE_DATA_ID :=	180	modulepar charstring PX_SGDU_PURCHASE_DATA_ID :=
192	"bcast://bcast.testingtech.com/purchaseData/purchaseData_id_1";	181	"bcast://bcast.testingtech.com/purchaseData/purchaseData_id_1";
193	}	182	}
194		183	
195	group PurchaseChannelFragment {	184	group PurchaseChannelFragment {
196	/**	185	/**
197	*	186	*
198	* @desc	187	* @desc
199	* Specifies the globally unique URI for the PurchaseChannel fragment	188	* Specifies the globally unique URI for the PurchaseChannel fragment
200	*/	189	*/
201	modulepar charstring PX_SGDU_PURCHASE_CHANNEL_ID :=	190	modulepar charstring PX_SGDU_PURCHASE_CHANNEL_ID :=
202	"bcast://bcast.testingtech.com/purchaseChannel/purchaseChannel_id_1";	191	"bcast://bcast.testingtech.com/purchaseChannel/purchaseChannel_id_1";
203	}	192	}
204		193	
205	group InteractivityDataFragment {	194	group InteractivityDataFragment {
206	/**	195	/**
207	*	196	*
208	* @desc	197	* @desc
209	* Specifies the globally unique URI for the Interactivity fragment	198	* Specifies the globally unique URI for the Interactivity fragment
210	*/	199	*/
211	modulepar charstring PX_SGDU_INTERACTIVITY_DATA_ID :=	200	modulepar charstring PX_SGDU_INTERACTIVITY_DATA_ID :=
212	"bcast://bcast.testingtech.com/interactivityData/interactivityData_id_1";	201	"bcast://bcast.testingtech.com/interactivityData/interactivityData_id_1";
213	}	202	}

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

214	}	203	}
215		204	
216	group PurchasingData {	205	group PurchasingData {
217	/**	206	/**
218	*	207	*
219	* @desc	208	* @desc
220	* Specifies the PurchaseURL in the PurchaseChannel fragment	209	* Specifies the PurchaseURL in the PurchaseChannel fragment
221	*/	210	*/
222	modulepar charstring PX_PURCHASE_URL :=	211	modulepar charstring PX_PURCHASE_URL :=
223	"bcast://bcast.testingtech.com/purchaseUrl";	212	"bcast://bcast.testingtech.com/purchaseUrl";
224		213	
225	/**	214	/**
226	*	215	*
227	* @desc	216	* @desc
228	* Specifies a value for the PriceInfo element of the PurchaseItem.	217	* Specifies a value for the PriceInfo element of the PurchaseItem.
229	*/	218	*/
230	modulepar float PX_MONETARY_PRICE := 15.00;	219	modulepar float PX_MONETARY_PRICE := 15.00;
231		220	
232	/**	221	/**
233	*	222	*
234	* @desc	223	* @desc
235	* Specifies the currency associated with the PX_MONETARY_PRICE, e.g., eur	224	* Specifies the currency associated with the PX_MONETARY_PRICE, e.g., eur
236	*/	225	*/
237	modulepar charstring PX_CURRENCY := "EUR";	226	modulepar charstring PX_CURRENCY := "EUR";
238		227	
239	}	228	}
240		229	
241	group InteractivityMediaDocument {	230	group InteractivityMediaDocument {
242	/**	231	/**
243	*	232	*
244	* @desc	233	* @desc
245	* Specifies the 'id' of the globally unique URI of the InteractivityMediaDocument	234	* Specifies the 'id' of the globally unique URI of the InteractivityMediaDocument
246	*/	235	*/
247	modulepar charstring PX_MEDIA_DOCUMENT_ID :=	236	modulepar charstring PX_MEDIA_DOCUMENT_ID :=
248	"InteractivityMediaDocumentId";	237	"InteractivityMediaDocumentId";
249		238	
250	/**	239	/**
251	*	240	*
252	* @desc	241	* @desc
253	* TSpecifies the globally unique URI of the Interactivity Media Document Group	242	* TSpecifies the globally unique URI of the Interactivity Media Document Group
254	*/	243	*/
255	modulepar charstring PX_MEDIA_DOCUMENT_GROUP_ID :=	244	modulepar charstring PX_MEDIA_DOCUMENT_GROUP_ID :=
256	"InteractivityMediaDocumentGroupId";	245	"InteractivityMediaDocumentGroupId";
257		246	
258	// change 4 (WK18): according to OMA-TS-BCAST_Services 5.3.6.2	247	// change 4 (WK18): according to OMA-TS-BCAST_Services 5.3.6.2
259	/**	248	/**
260	*	249	*
261	* @desc	250	* @desc
262	* Specifies the globally unique URI of the Interactivity Media Object Group	251	* Specifies the globally unique URI of the Interactivity Media Object Group
263	*/	252	*/
264	modulepar charstring PX_MEDIA_OBJECT_GROUP_ID :=	253	modulepar charstring PX_MEDIA_OBJECT_GROUP_ID :=
265	"InteractivityMediaObjectGroupId";	254	"InteractivityMediaObjectGroupId";
266		255	
267	/**	256	/**
268	*	257	*
269	* @desc	258	* @desc
270	* Specifies the preListenIndicator of the InteractivityData fragment	259	* Specifies the preListenIndicator of the InteractivityData fragment
271	*/	260	*/
272	modulepar boolean PX_PRELISTEN_INDICATOR := false;	261	modulepar boolean PX_PRELISTEN_INDICATOR := false;
273		262	
274	/**	263	/**
275	*	264	*
276	* @desc	265	* @desc
277	* Specifies the ContentLocation of the MediaObjectSet	266	* Specifies the ContentLocation of the MediaObjectSet
278	*/	267	*/
279	modulepar charstring PX_MEDIA_OBJECT_SET_URI :=	268	modulepar charstring PX_MEDIA_OBJECT_SET_URI :=
280	"http://www.testingtech.com/mediaObjectSet.gz";	269	"http://www.testingtech.com/mediaObjectSet.gz";
281	}	270	}
282		271	
283	group SMS {	272	group SMS {
284	/**	273	/**

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

285	*	274	*
286	* @desc	275	* @desc
287	* Specifies the URI of the SMS	276	* Specifies the URI of the SMS
288	*/	277	*/
289	modulepar charstring PX_SMS_URI := "http://www.openmobilealliance.org/";	278	modulepar charstring PX_SMS_URI := "http://www.openmobilealliance.org/";
290	}	279	}
291		280	
292	group MMS {	281	group MMS {
293	/**	282	/**
294	*	283	*
295	* @desc	284	* @desc
296	* Specifies the file name of a MMS template	285	* Specifies the file name of a MMS template
297	*/	286	*/
298	modulepar charstring PX_MMS_TEMPLATE := "vote.mtd";	287	modulepar charstring PX_MMS_TEMPLATE := "vote.mtd";
299		288	
300	/**	289	/**
301	*	290	*
302	* @desc	291	* @desc
303	* Specifies the file name of a picture	292	* Specifies the file name of a picture
304	*/	293	*/
305	modulepar charstring PX_MMS_VOTE_A_PICTURE := "ChoiceA.png";	294	modulepar charstring PX_MMS_VOTE_A_PICTURE := "ChoiceA.png";
306		295	
307	/**	296	/**
308	*	297	*
309	* @desc	298	* @desc
310	* Specifies the file name of a picture	299	* Specifies the file name of a picture
311	*/	300	*/
312	modulepar charstring PX_MMS_VOTE_PICTURE := "ChoiceB.jpg";	301	modulepar charstring PX_MMS_VOTE_PICTURE := "ChoiceB.jpg";
313		302	
314	/**	303	/**
315	*	304	*
316	* @desc	305	* @desc
317	* Specifies the file name of a text file	306	* Specifies the file name of a text file
318	*/	307	*/
319	modulepar charstring PX_MMS_TEXT_FILE := "Instructions.txt";	308	modulepar charstring PX_MMS_TEXT_FILE := "Instructions.txt";
320	}	309	}
321		310	
322	group XHTML {	311	group XHTML {
323	/**	312	/**
324	*	313	*
325	* @desc	314	* @desc
326	* Specifies the file name of the XHTML file	315	* Specifies the file name of the XHTML file
327	*/	316	*/
328	modulepar charstring PX_XHTML_FILE_ID := "vote-xhtml";	317	modulepar charstring PX_XHTML_FILE_ID := "vote-xhtml";
329		318	
330	/**	319	/**
331	*	320	*
332	* @desc	321	* @desc
333	* Specifies the content type of the XHTML file	322	* Specifies the content type of the XHTML file
334	*/	323	*/
335	modulepar charstring PX_XHTML_CONTENT_TYPE :=	324	modulepar charstring PX_XHTML_CONTENT_TYPE :=
336	"application/vnd.wap.xhtml+xml";	325	"application/vnd.wap.xhtml+xml";
337	}	326	}
338		327	
339	group FileDelivery {	328	group FileDelivery {
340	// change 34 (WK18): rename "TVChannelIcon" SDP session into "Data_Session", rename respect	329	// change 34 (WK18): rename "TVChannelIcon" SDP session into "Data_Session", rename respect
341	/**	330	/**
342	*	331	*
343	* @desc	332	* @desc
344	* Specifies the file name of a picture, e.g., file1.jpg	333	* Specifies the file name of a picture, e.g., file1.jpg
345	*/	334	*/
346	modulepar charstring PX_DATA_SESSION := "TvChannelIcon.jpg";	335	modulepar charstring PX_DATA_SESSION := "TvChannelIcon.jpg";
347		336	
348	/**	337	/**
349	*	338	*
350	* @desc	339	* @desc
351	* Specifies the file name containing video, e.g., video1.mov	340	* Specifies the file name containing video, e.g., video1.mov
352	*/	341	*/
353	modulepar charstring PX_FILE_VIDEO_PROG1 := "video1.mov";	342	modulepar charstring PX_FILE_VIDEO_PROG1 := "video1.mov";
354		343	
355	/**	344	/**

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

357	* @desc		345	*
358	* Specifies the file name containing video, e.g., video2.mov		346	* @desc
359	*/		347	* Specifies the file name containing video, e.g., video2.mov
360	modulepar charstring PX_FILE_VIDEO_PROG2 := "video2.mov";		348	*/
361			349	modulepar charstring PX_FILE_VIDEO_PROG2 := "video2.mov";
362	/**		350	
363	*		351	/**
364	* @desc		352	*
365	* Specifies the file name containing audio, e.g., audio1.mp3		353	* @desc
366	*/		354	* Specifies the file name containing audio, e.g., audio1.mp3
367	modulepar charstring PX_FILE_AUDIO_PROG_1 := "audio1.mp3";		355	*/
368			356	modulepar charstring PX_FILE_AUDIO_PROG_1 := "audio1.mp3";
369	/**		357	
370	*		358	/**
371	* @desc		359	*
372	* Specifies the file name containing audio, e.g., audio2.mp3		360	* @desc
373	*/		361	* Specifies the file name containing audio, e.g., audio2.mp3
374	modulepar charstring PX_FILE_AUDIO_PROG_2 := "audio2.mp3";		362	*/
375			363	modulepar charstring PX_FILE_AUDIO_PROG_2 := "audio2.mp3";
376	/**		364	
377	*		365	/**
378	* @desc		366	*
379	* Specifies the file name containing audio, e.g., audio_eng.mp3		367	* @desc
380	*/		368	* Specifies the file name containing audio, e.g., audio_eng.mp3
381	modulepar charstring PX_FILE_AUDIO_ENG := "audio_eng.mp3";		369	*/
382			370	modulepar charstring PX_FILE_AUDIO_ENG := "audio_eng.mp3";
383	/**		371	
384	*		372	/**
385	* @desc		373	*
386	* Specifies the file name containing audio, e.g., audio_ger.mp3		374	* @desc
387	*/		375	* Specifies the file name containing audio, e.g., audio_ger.mp3
388	modulepar charstring PX_FILE_AUDIO_GER := "audio_ger.mp3";		376	*/
389			377	modulepar charstring PX_FILE_AUDIO_GER := "audio_ger.mp3";
390	/**		378	
391	*		379	/**
392	* @desc		380	*
393	* Specifies the time in seconds for the video to play		381	* @desc
394	*/		382	* Specifies the time in seconds for the video to play
395	modulepar integer PX_PLAY_TIME := 300;		383	*/
396			384	modulepar integer PX_PLAY_TIME := 300;
397	// change 5 (WK18): update and documentation of ics/ixit settings		385	
398	/**		386	// change 5 (WK18): update and documentation of ics/ixit settings
399	* @desc		387	/**
400	* streaming server sends data to this IP address		388	* @desc
401	*/		389	* streaming server sends data to this IP address
402	modulepar charstring PX_DATA_STREAMING_DESTINATION_IP_1 := "232.0.3.0";		390	*/
403			391	modulepar charstring PX_DATA_STREAMING_DESTINATION_IP_1 := "232.0.3.0";
404	// change 5 (WK18): update and documentation of ics/ixit settings		392	
405	/**		393	// change 5 (WK18): update and documentation of ics/ixit settings
406	* @desc		394	/**
407	* streaming server sends data to this IP address		395	* @desc
408	*/		396	* streaming server sends data to this IP address
409	modulepar charstring PX_DATA_STREAMING_DESTINATION_IP_2 := "232.0.4.0";		397	*/
410			398	modulepar charstring PX_DATA_STREAMING_DESTINATION_IP_2 := "232.0.4.0";
411	}		399	
412			400	}
413	group FluteSessionParameters {	// change 02 (W	401	
414			402	group FluteSessionParameters {
415	/**		403	
416	*		404	/**
417	* @desc		405	*
418	* Specifies the IP address for the DVB-H bootstrapping session		406	* @desc
419	*/		407	* Specifies the IP address for the DVB-H bootstrapping session
420	modulepar charstring PX_DVB_H_FLUTE_SESSION_IP := "224.00.23.14";	// change 0	408	*/
421			409	modulepar charstring PX_DVB_H_FLUTE_SESSION_IP := "224.00.23.14";
422	/**		410	
423	* @desc Specifies the port for the DVB-H bootstrapping session		411	/**
424	*/		412	* @desc Specifies the port for the DVB-H bootstrapping session
425	modulepar integer PX_DVB_H_FLUTE_SESSION_PORT := 9214;	// change 0	413	*/
426			414	modulepar integer PX_DVB_H_FLUTE_SESSION_PORT := 9214;
			415	

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

427	/**	416	/**
428	* @desc Specifies the IP address for the Flute SGDD session	417	* @desc Specifies the IP address for the Flute SGDD session
429	*/	418	*/
430	modulepar charstring PX_SGDD_FLUTE_SESSION_IP := "232.0.1.0";	419	modulepar charstring PX_SGDD_FLUTE_SESSION_IP := "232.0.1.0";
431		420	
432	/**	421	/**
433	* @desc Specifies the port for the Flute SGDD session	422	* @desc Specifies the port for the Flute SGDD session
434	*/	423	*/
435	modulepar integer PX_SGDD_FLUTE_SESSION_PORT := 9000;	424	modulepar integer PX_SGDD_FLUTE_SESSION_PORT := 9000;
436		425	
437	/**	426	/**
438	* @desc Specifies the IP address for the Flute SGDU session	427	* @desc Specifies the IP address for the Flute SGDU session
439	*/	428	*/
440	modulepar charstring PX_SGDU_FLUTE_SESSION_IP := "232.0.1.0";	429	modulepar charstring PX_SGDU_FLUTE_SESSION_IP := "232.0.1.0";
441		430	
442	/**	431	/**
443	* @desc Specifies the port for the Flute SGDU session	432	* @desc Specifies the port for the Flute SGDU session
444	*/	433	*/
445	modulepar integer PX_SGDU_FLUTE_SESSION_PORT := 9003;	434	modulepar integer PX_SGDU_FLUTE_SESSION_PORT := 9003;
446		435	
447	/**	436	/**
448	* @desc Specifies the IP address for a Flute data session	437	* @desc Specifies the IP address for a Flute data session
449	*/	438	*/
450	modulepar charstring PX_DATA_FLUTE_SESSION_IP := "232.0.7.0";	439	modulepar charstring PX_DATA_FLUTE_SESSION_IP := "232.0.7.0";
451		440	
452	/**	441	/**
453	* @desc Specifies the port for a Flute data session	442	* @desc Specifies the port for a Flute data session
454	*/	443	*/
455	modulepar integer PX_DATA_FLUTE_SESSION_PORT := 10080;	444	modulepar integer PX_DATA_FLUTE_SESSION_PORT := 10080;
456	}	445	}
457		446	
458	group SessionDescription {	447	group SessionDescription {
459	/**	448	/**
460	* @desc Specifies the ID of the SDP	449	* @desc Specifies the ID of the SDP
461	*/	450	*/
462	modulepar charstring PX_SDP_VIDEO_PROG_1_REF_ID := "myFirstSDP";	451	modulepar charstring PX_SDP_VIDEO_PROG_1_REF_ID := "myFirstSDP";
463		452	
464	/**	453	/**
465	* @desc Specifies the ID of the SDP	454	* @desc Specifies the ID of the SDP
466	*/	455	*/
467	modulepar charstring PX_SDP_VIDEO_PROG_2_REF_ID := "mySecondSDP";	456	modulepar charstring PX_SDP_VIDEO_PROG_2_REF_ID := "mySecondSDP";
468		457	
469	/**	458	/**
470	* @desc Specifies the ID of the SDP	459	* @desc Specifies the ID of the SDP
471	*/	460	*/
472	// change 34 (WK18): rename "TVChannelIcon" SDP session into "Data_Session", rename respect	461	// change 34 (WK18): rename "TVChannelIcon" SDP session into "Data_Session", rename respect
473	modulepar charstring PX_SDP_DATA_SESSION_REF_ID := "DataSession";	462	modulepar charstring PX_SDP_DATA_SESSION_REF_ID := "DataSession";
474		463	
475	/**	464	/**
476	*	465	*
477	* @desc	466	* @desc
478	Specifies an SDP with the default settings	467	Specifies an SDP with the default settings
479	*/	468	*/
480	modulepar charstring PX_SDP_VIDEO_PROG_1 :=	469	modulepar charstring PX_SDP_VIDEO_PROG_1 :=
481	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	470	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
482		471	
483	/**	472	/**
484	*	473	*
485	* @desc	474	* @desc
486	Specifies an SDP with the default settings	475	Specifies an SDP with the default settings
487	*/	476	*/
488	modulepar charstring PX_SDP_VIDEO_PROG_2 :=	477	modulepar charstring PX_SDP_VIDEO_PROG_2 :=
489	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	478	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
490		479	
491	/**	480	/**
492	*	481	*
493	* @desc	482	* @desc
494	Specifies an SDP with a particular language	483	Specifies an SDP with a particular language
495	*/	484	*/
496	modulepar charstring PX_SDP_AUDIO_ENG :=	485	modulepar charstring PX_SDP_AUDIO_ENG :=
497	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	486	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

498		487	
499	/**	488	/**
500	*	489	*
501	* @desc	490	* @desc
502	* Specifies an SDP with a particular language	491	* Specifies an SDP with a particular language
503	*/	492	*/
504	modulepar charstring PX_SDP_AUDIO_GER :=	493	modulepar charstring PX_SDP_AUDIO_GER :=
505	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	494	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
506		495	
507	/**	496	/**
508	*	497	*
509	* @desc	498	* @desc
510	* Specifies SDP for audio stream of TV Program 1	499	* Specifies SDP for audio stream of TV Program 1
511	*/	500	*/
512	modulepar charstring PX_SDP_AUDIO_PROG_1 :=	501	modulepar charstring PX_SDP_AUDIO_PROG_1 :=
513	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	502	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
514		503	
515	/**	504	/**
516	*	505	*
517	* @desc	506	* @desc
518	* Specifies SDP for audio stream of TV Program 2	507	* Specifies SDP for audio stream of TV Program 2
519	*/	508	*/
520	modulepar charstring PX_SDP_AUDIO_PROG_2 :=	509	modulepar charstring PX_SDP_AUDIO_PROG_2 :=
521	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	510	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
522		511	
523	// change 34 (WK18): rename "TVChannelIcon" SDP session into "Data_Session", rename respect	512	// change 34 (WK18): rename "TVChannelIcon" SDP session into "Data_Session", rename respect
524	/**	513	/**
525	*	514	*
526	* @desc	515	* @desc
527	* Specifies the SDP to be used for file and Interactivity Media Document delivery over	516	* Specifies the SDP to be used for file and Interactivity Media Document delivery over
528	*/	517	*/
529	modulepar charstring PX_SDP_DATA_SESSION :=	518	modulepar charstring PX_SDP_DATA_SESSION :=
530	"v=0\r\no=user123 2890844526 2890842807 IN IP6 2201:056D::112E:144A:1E24\r\ns= Exam	519	"v=0\r\no=user123 2890844526 2890842807 IN IP6 2201:056D::112E:144A:1E24\r\ns= Exam
531		520	
532	/**	521	/**
533	*	522	*
534	* @desc	523	* @desc
535	* Specifies the SDP to be used for the Content Protection test cases using IPSec, e.g.	524	* Specifies the SDP to be used for the Content Protection test cases using IPSec, e.g.
536	*/	525	*/
537	modulepar charstring PX_SDP_SERVICE_PROTECTION_IPSEC :=	526	modulepar charstring PX_SDP_SERVICE_PROTECTION_IPSEC :=
538	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	527	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
539		528	
540	/**	529	/**
541	*	530	*
542	* @desc	531	* @desc
543	* Specifies the SDP to be used for the Content Protection test cases using SRTP, e.g.,	532	* Specifies the SDP to be used for the Content Protection test cases using SRTP, e.g.,
544	*/	533	*/
545	modulepar charstring PX_SDP_SERVICE_PROTECTION_SRTP :=	534	modulepar charstring PX_SDP_SERVICE_PROTECTION_SRTP :=
546	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	535	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
547		536	
548	/**	537	/**
549	*	538	*
550	* @desc	539	* @desc
551	* Specifies the SDP to be used for the Content Protection test cases using ISMACrypt, e	540	* Specifies the SDP to be used for the Content Protection test cases using ISMACrypt, e
552	*/	541	*/
553	modulepar charstring PX_SDP_SERVICE_PROTECTION_ISMA :=	542	modulepar charstring PX_SDP_SERVICE_PROTECTION_ISMA :=
554	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV	543	"v=0\r\no=- 424 3292855200 IN IP6 FF15:0:0:0:0:81:1BC\r\ns=Unencrypted Mobile TV
555	}	544	}
556		545	
557	group Misc {	546	group Misc {
558	/**	547	/**
559	*	548	*
560	* @desc	549	* @desc
561	* On true, test case runs in simulate mode.	550	* On true, test case runs in simulate mode.
562	*/	551	*/
563	modulepar boolean PX_SIMULATE_TC := false;	552	modulepar boolean PX_SIMULATE_TC := false;
564	}	553	}
565		554	
566	// change 1 (WK18): for content protection tests	555	// change 1 (WK18): for content protection tests
567	group GBA {	556	group GBA {
568	/**	557	/**

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

[illegible]

```

610
611 // change 2 (WK23): GBA authentication/authorization extensions for content protection test
612 /**
613 *
614 * @desc Specifies the supported GBA algorithm in the DUT
615 */
616 modulepar GBAType PX_GBA := e_gba_u;
617
618 // change 3 (WK34): Service Protection: secure streaming service
619 /**
620 *
621 * @desc Specifies the lower bound of the STKM Timestamp
622 */
623 modulepar octetstring PX_TS_LOW := '00'O;
624
625 /**
626 *
627 * @desc Specifies the upper bound of the STKM Timestamp
628 */
629 modulepar octetstring PX_TS_HIGH := '0f'O;
630
631 /**
632 *
633 * @desc Specifies the key utilized for encrypting the key included in LTKM KEMAC
634 */
635 modulepar octetstring PX_LTKM := 'ff'O;
636
637 /**
638 *
639 * @desc Specifies the initial ID of the MBMS Transport Key (MTK)

```

[illegible]

Datei: AtsBCast_ModuleParameters.ttcn (Fortsetzung)

640	*/			
641	modulepar octetstring PX_MTK_ID_START := '0001'O;			
642				
643	/**			
644	*			
645	* @desc Specifies the initial key of the traffic encryption master key			
646	*/			
647	modulepar octetstring PX_SRTP_MASTER_KEY_START := '000000000000000000000000000001'O; // 1			
648				
649	/**			
650	*			
651	* @desc Specifies the lenght of the traffic encryption salt key			
652	*/			
653	modulepar integer PX_SALT_KEY_LENGTH := 112; // length in bits			
654	}	=	599	}
655			600	
656	// change 1 (WK18): for content protection tests		601	// change 1 (WK18): for content protection tests
657	group BSM {		602	group BSM {
658			603	
659	/**		604	/**
660	*		605	*
661	* @desc		606	* @desc
662	* Specifies the BCAST Subscription Management (BSM) Server FQDN		607	* Specifies the BCAST Subscription Management (BSM) Server FQDN
663	*/		608	*/
664	modulepar charstring PX_BSM_FQDN := "bmsc.rs.de";		609	modulepar charstring PX_BSM_FQDN := "bmsc.rs.de";
665			610	
666	/**		611	/**
667	*		612	*
668	* @desc		613	* @desc
669	* Specifies the Mobile Country Code		614	* Specifies the Mobile Country Code
670	*/		615	*/
671	modulepar Hex3 PX_MCC := '001'H; // 1.5 bytes long		616	modulepar Hex3 PX_MCC := '001'H; // 1.5 bytes long
672			617	
673	/**		618	/**
674	*		619	*
675	* @desc		620	* @desc
676	* Specifies the Mobile Network Code		621	* Specifies the Mobile Network Code
677	*/		622	*/
678	modulepar Hex3 PX_MNC := '001'H; // 1.5 bytes long		623	modulepar Hex3 PX_MNC := '001'H; // 1.5 bytes long
679			624	
680	/**		625	/**
681	*		626	*
682	* @desc		627	* @desc
683	* Specifies the a group of SEK/PEKs that are identified by the same Key group part of		628	* Specifies the a group of SEK/PEKs that are identified by the same Key group part of
684	*/		629	*/
685	modulepar Hex4 PX_KEY_GROUP := '0001'H; // 2 bytes long		630	modulepar Hex4 PX_KEY_GROUP := '0001'H; // 2 bytes long
686			631	
687	/**		632	/**
688	*		633	*
689	* @desc		634	* @desc
690	* Specifies within a key group, which SEK/PEK is used		635	* Specifies within a key group, which SEK/PEK is used
691	*/		636	*/
692	modulepar Hex4 PX_KEY_NUMBER := '0000'H; // 2 bytes long		637	modulepar Hex4 PX_KEY_NUMBER := '0000'H; // 2 bytes long
693	}		638	}
694			639	
695			640	
696	} // end module AtsBCast_ModuleParameters		641	} // end module AtsBCast_ModuleParameters

Datei: AtsBCast_ServiceProtectionTests.ttcn

1	// change 1 (WK18): for content protection tests	=	1	// change 1 (WK18): for content protection tests
2	module AtsBCast_ServiceProtectionTests {		2	module AtsBCast_ServiceProtectionTests {
3	import from AtsBCast_TestConfiguration_Functions all;		3	import from AtsBCast_TestConfiguration_Functions all;
4	import from AtsBCast_TestSystem all;		4	import from AtsBCast_TestSystem all;
5	import from AtsBCast_Main_Functions all;		5	import from AtsBCast_Main_Functions all;
6	import from LibBCast_ServiceGuide_TypesAndValues all;		6	import from LibBCast_ServiceGuide_TypesAndValues all;
7	import from LibCommon_BasicTypesAndValues all;		7	import from LibCommon_BasicTypesAndValues all;
8	import from LibBCast_Common_TypesAndValues all;		8	import from LibBCast_Common_TypesAndValues all;
9	import from AtsBCast_ServiceGuide_Functions all;		9	import from AtsBCast_ServiceGuide_Functions all;
10	import from LibBCast_ServiceGuide_Templates all;		10	import from LibBCast_ServiceGuide_Templates all;
11	import from AtsBCast_ModuleParameters all;		11	import from AtsBCast_ModuleParameters all;
12	import from LibBCast_ModuleParameters all;		12	import from LibBCast_ModuleParameters all;
13	import from LibCommon_GBA_BSF {		13	import from LibCommon_GBA_BSF {
14	group externalFunctions;		14	group externalFunctions;
15	function f_bsf_generateBtid; // change 2 (WK23): GBA authentication/authorization extensions for cc	+-		
16	}	=	15	}
17	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)	+-		
18	import from LibCommon_Mikey_Templates all;			
19	import from LibCommon_Mikey_TypesAndValues all;			
20	import from LibCommon_Mikey all;			
21	// change 3 (WK34): Service Protection: secure streaming service			
22	import from LibBCast_ServicePrimitives_TypesAndValues {			
23	group streamingServerPrimitives;			
24	}			
25	import from LibCommon_DataStrings all;			
26		=	16	
27	group bcastConformanceTestCases {		17	group bcastConformanceTestCases {
28	group clientConformanceTestCases {		18	group clientConformanceTestCases {
29			19	
30	/**		20	/**
31	*		21	*
32	* @desc		22	* @desc
33	* <p>Registration</p>		23	* <p>Registration</p>
34	* <p>Test Case Description</p>		24	* <p>Test Case Description</p>
35	* <p>When the BCAST Client is started in the terminal that initiates the		25	* <p>When the BCAST Client is started in the terminal that initiates the
36	* MBMS User Service Registration procedure.</p>		26	* MBMS User Service Registration procedure.</p>
37	* <p>Specification Reference</p>		27	* <p>Specification Reference</p>
38	* <p>[BCAST10-Services] Section 5.1.6.7</p>		28	* <p>[BCAST10-Services] Section 5.1.6.7</p>
39	* <p>Preconditions</p>		29	* <p>Preconditions</p>
40	* <p>The service guide cache of the terminal is erased.		30	* <p>The service guide cache of the terminal is erased.
41	* The value for the ServiceID field (valid concatenation of a GlobalPurchaseItemID		31	* The value for the ServiceID field (valid concatenation of a GlobalPurchaseItemID
42	* and a PurchaseDataReference) is known by the Terminal		32	* and a PurchaseDataReference) is known by the Terminal
43	* UICC contains Key management function: GBA_U and (MBMS or BCAST) key management		33	* UICC contains Key management function: GBA_U and (MBMS or BCAST) key management
44	* UICC contains credentials and algorithms according to 34.108 [NEW REF] chapter 8		34	* UICC contains credentials and algorithms according to 34.108 [NEW REF] chapter 8
45	* UICC SQN value is always constant.		35	* UICC SQN value is always constant.
46	* There is a Service Guide available. For the Service Guide instantiation details see 5.		36	* There is a Service Guide available. For the Service Guide instantiation details see 5.
47	* Can be tested at the same time as: 5.10.1.3 - Purchasing information.		37	* Can be tested at the same time as: 5.10.1.3 - Purchasing information.
48	* </p>		38	* </p>
49	* <p>Test Procedure</p>		39	* <p>Test Procedure</p>
50	* <p>		40	* <p>
51	* 		41	*
52	* Activate the BCAST application on the terminal.	<>	42	* Activate the BCAST application on the terminal.
53	* Terminal initiates the MBMS User Service Registration procedure with User Service		43	* Terminal initiates the MBMS User Service Registration procedure with User Service
54	* 	=	44	*
55	* a. The Terminal sends an initial HTTP POST (Registration indication) with		45	* a. The Terminal sends an initial HTTP POST (Registration indication) with
56	* b. Test tool replies with HTTP 401 Unauthorized response with Digest chall		46	* b. Test tool replies with HTTP 401 Unauthorized response with Digest chall
57	* c. Terminal uses MRK and B-TID derived from the valid bootstrapping sessio		47	* c. Terminal uses MRK and B-TID derived from the valid bootstrapping sessio
58	* 		48	*
59	* 		49	*
60	* The test tool replies with HTTP 200 OK.		50	* The test tool replies with HTTP 200 OK.
61	* 		51	*
62	* <p>Note1: The use of test data proposed by the [3GPP TS 35.207-700 (Implementer's Test		52	* <p>Note1: The use of test data proposed by the [3GPP TS 35.207-700 (Implementer's Test
63	* <p>Note2: In case there is no valid bootstrapping context, the terminal runs bootstrapp		53	* <p>Note2: In case there is no valid bootstrapping context, the terminal runs bootstrapp
64	* </p>		54	* </p>
65	* <p>Pass-Criteria</p>		55	* <p>Pass-Criteria</p>
66	* <p>2a. The terminal sends the Registration indication</p>		56	* <p>2a. The terminal sends the Registration indication</p>
67	* <p>2c. The second POST request is properly formatted and contains the authentication head		57	* <p>2c. The second POST request is properly formatted and contains the authentication head
68	* @verdict		58	* @verdict
69	* pass in case the client pass the conformance test.		59	* pass in case the client pass the conformance test.
70	* @verdict		60	* @verdict
71	* fail in case the client does not pass the conformance test.		61	* fail in case the client does not pass the conformance test.

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

72	* @verdict		62	* @verdict
73	* inconc in case a guard timer expires.		63	* inconc in case a guard timer expires.
74	*/		64	*/
75	testcase TC_BCAST_SERVPROT_101a() runs on BCastMainComponent system BCastPlatformComponent {	<>	65	testcase TC_BCAST_SERVPROT_101a() runs on BCastMainComponent system BCastPlatformComponent {
76	// init components and ports		66	// init components and ports
77	f_createComponents();		67	f_createComponents();
78	f_cf_DNS_up();		68	f_cf_DNS_up();
79	f_cf_BSM_up();		69	f_cf_BSM_up();
80	f_cf_UpperTester_up();		70	f_cf_UpperTester_up();
81			71	
82	// preamble		72	// preamble
83	f_cf_enable_DNS();		73	f_cf_enable_DNS();
84	f_startCommonUtPreamble();		74	f_startCommonUtPreamble();
85			75	
86	// change 2 (WK23): GBA authentication/authorization extensions for content protection test			
87	f_main_ctrl_InitProtectionKeysAndValues();			
88	f_main_ctrl_SetBootstrappingTransactionId(f_bsf_generateBtid());			
89				
90	// change 1 (WK23): Service Request extensions for content protection tests			
91	// Service Protection: simplified Service Request			
92	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registrat		76	// Bcast Subscription Managemant acts as test component for MBMS User Service Registration
93	// and Service Request/Response is not allowed			
94	f_main_ctrl_BSM_Procedures(true, false, false, false);		77	f_main_ctrl_MBMS_User_Service_Procedure(false);
95			78	
96	// postamble		79	// postamble
97	f_main_ut_PowerOff();		80	f_main_ut_PowerOff();
98			81	
99	f_cf_UpperTester_down();		82	f_cf_UpperTester_down();
100	f_cf_BSM_down();		83	f_cf_BSM_down();
101	f_cf_DNS_down();		84	f_cf_DNS_down();
102	f_terminateComponents();		85	f_terminateComponents();
103	}	=	86	}
104			87	
105	/**		88	/**
106	*		89	*
107	* @desc		90	* @desc
108	* <p>GBA-U Bootstrapping USIM</p>		91	* <p>GBA-U Bootstrapping USIM</p>
109	* <p>Test Case Description</p>		92	* <p>Test Case Description</p>
110	* <p>When the terminal needs to do authentication and there is no existing		93	* <p>When the terminal needs to do authentication and there is no existing
111	* bootstrapping context that initiates the bootstrapping flow.</p>		94	* bootstrapping context that initiates the bootstrapping flow.</p>
112	* <p>Specification Reference</p>		95	* <p>Specification Reference</p>
113	* <p>[BCAST10-ServContProt] Section 6.5.1</p>	<>	96	* <p>[BCAST10-ServContProt] Section 6.5.1</p>
114	* <p>Preconditions</p>	=	97	* <p>Preconditions</p>
115	* <p>No bootstrapping context exists between the terminal and the test tool.		98	* <p>No bootstrapping context exists between the terminal and the test tool.
116	* All existing credentials are marked as invalid.		99	* All existing credentials are marked as invalid.
117	* 		100	*
118	* The value for the ServiceID field (valid concatenation of a GlobalPurchaseItemID		101	* The value for the ServiceID field (valid concatenation of a GlobalPurchaseItemID
119	* UICC contains Key management function: GBA_U and MBMS or BCAST key management.		102	* UICC contains Key management function: GBA_U and MBMS or BCAST key management.
120	* 		103	*
121	* UICC contains credentials and algorithms according to 34.108 [NEW REF] chapter 8		104	* UICC contains credentials and algorithms according to 34.108 [NEW REF] chapter 8
122	* UICC SQN value is always constant.		105	* UICC SQN value is always constant.
123	* BSF address is set up in the terminal		106	* BSF address is set up in the terminal
124	* Can be tested at the same time as: 5.10.1.1 - Registration</p>		107	* Can be tested at the same time as: 5.10.1.1 - Registration</p>
125	* <p>Test Procedure</p>		108	* <p>Test Procedure</p>
126	* <p>		109	* <p>
127	* 		110	*
128	* Terminal retrieves from the Service Guide the permissionIssuerURI, and extracts f		111	* Terminal retrieves from the Service Guide the permissionIssuerURI, and extracts f
129	* Terminal detects that a bootstrapping procedure is needed (no valid SRK available		112	* Terminal detects that a bootstrapping procedure is needed (no valid SRK available
130	* The Terminal runs the bootstrapping procedure		113	* The Terminal runs the bootstrapping procedure
131	* 		114	*
132	* a. The Terminal sends an initial POST request (HTTP request) containing th		115	* a. The Terminal sends an initial POST request (HTTP request) containing th
133	* b. Test tool replies with HTTP 401 Unauthorized response with Digest chall		116	* b. Test tool replies with HTTP 401 Unauthorized response with Digest chall
134	* c. RAND and AUTN are used by USIM to generate RES authentication challenge		117	* c. RAND and AUTN are used by USIM to generate RES authentication challenge
135	* d. Test tool generates B-TID from the IMPI and sends a 200OK message inclu		118	* d. Test tool generates B-TID from the IMPI and sends a 200OK message inclu
136	* e. The terminal stores B-TID and key lifetime in the USIM EFGBABP file		119	* e. The terminal stores B-TID and key lifetime in the USIM EFGBABP file
137	* f. The terminal sends the NAF_ID received in step 1 (FQDN and UA security		120	* f. The terminal sends the NAF_ID received in step 1 (FQDN and UA security
138	* 		121	*
139	* 		122	*
140	* At this time the test tool and the USIM share the bootstrap Key material Ks_int_NA		123	* At this time the test tool and the USIM share the bootstrap Key material Ks_int_NA
141	* 		124	*
142	* </p>		125	* </p>

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

143	* <p>Pass-Criteria</p>	126	* <p>Pass-Criteria</p>
144	* <p>4a. The terminal sends a POST request with the appropriate IMPI.</p>	127	* <p>4a. The terminal sends a POST request with the appropriate IMPI.</p>
145	* <p>4c. RES corresponds to the XRES in the test tool.</p>	128	* <p>4c. RES corresponds to the XRES in the test tool.</p>
146	* @verdict	129	* @verdict
147	* pass in case the client pass the conformance test.	130	* pass in case the client pass the conformance test.
148	* @verdict	131	* @verdict
149	* fail in case the client does not pass the conformance test.	132	* fail in case the client does not pass the conformance test.
150	* @verdict	133	* @verdict
151	* inconc in case a guard timer expires.	134	* inconc in case a guard timer expires.
152	*/	135	*/
153	testcase TC_BCAST_SERVPROT_101b() runs on BCastMainComponent system BCastPlatformComponent {	136	testcase TC_BCAST_SERVPROT_101b() runs on BCastMainComponent system BCastPlatformComponent {
154	// init components and ports	137	// init components and ports
155	f_createComponents();	138	f_createComponents();
156	f_cf_DNS_up();	139	f_cf_DNS_up();
157	f_cf_BSM_up();	140	f_cf_BSM_up();
158	f_cf_BSF_up();	141	f_cf_BSF_up();
159	f_cf_UpperTester_up();	142	f_cf_UpperTester_up();
160		143	
161	// preamble	144	// preamble
162	f_cf_enable_DNS();	145	f_cf_enable_DNS();
163	f_startCommonUtPreamble();	146	f_startCommonUtPreamble();
164		147	
165	// change 2 (WK23): GBA authentication/authorization extensions for content protection test		
166	f_main_ctrl_InitProtectionKeysAndValues();		
167	f_main_ctrl_SetBootstrappingTransactionId(f_bsf_generateBtid());		
168			
169	// change 1 (WK23): Service Request extensions for content protection tests		
170	// Service Protection: simplified Service Request		
171	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registrat	148	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registrat
172	// and Service Request/Response is not allowed		
173	f_main_ctrl_BSM_Procedures(true, true, false, true);	149	f_main_ctrl_MBMS_User_Service_Procedure(true);
174	// GBA/BSF acts as test component	150	// GBA/BSF acts as test component
175	f_main_ctrl_GBA_U_Bootstrapping(false);	151	f_main_ctrl_GBA_U_Bootstrapping(false);
176		152	
177	// postamble	153	// postamble
178	f_main_ut_PowerOff();	154	f_main_ut_PowerOff();
179		155	
180	f_cf_UpperTester_down();	156	f_cf_UpperTester_down();
181	f_cf_BSF_down();	157	f_cf_BSF_down();
182	f_cf_BSM_down();	158	f_cf_BSM_down();
183	f_cf_DNS_down();	159	f_cf_DNS_down();
184	f_terminateComponents();	160	f_terminateComponents();
185	}	161	}
186		162	
187	testcase TC_BCAST_SERVPROT_101c() runs on BCastMainComponent system BCastPlatformComponent {	163	testcase TC_BCAST_SERVPROT_101c() runs on BCastMainComponent system BCastPlatformComponent {
188	// init components and ports	164	// init components and ports
189	f_createComponents();	165	f_createComponents();
190	f_cf_Broadcast_up();	166	f_cf_Broadcast_up();
191	f_cf_UpperTester_up();	167	f_cf_UpperTester_up();
192	f_cf_DNS_up();	168	f_cf_DNS_up();
193	f_cf_BSM_up();	169	f_cf_BSM_up();
194	f_cf_BSF_up();	170	f_cf_BSF_up();
195		171	
196	// preamble	172	// preamble
197	f_cf_enable_DNS();	173	f_cf_enable_DNS();
198	f_startCommonUtPreamble();	174	f_startCommonUtPreamble();
199		175	
200	// change 2 (WK23): GBA authentication/authorization extensions for content protection test		
201	var hexstring protectionKeyId := PX_MCC & PX_MNC & PX_KEY_GROUP & PX_KEY_NUMBER;		
202	f_main_ctrl_InitProtectionKeysAndValues();		
203	f_main_ctrl_SetProtectionKeyId(protectionKeyId);		
204	f_main_ctrl_SetBootstrappingTransactionId(f_bsf_generateBtid());		
205	// GBA/BSF as parallel test component		
206	f_main_ctrl_GBA_U_Bootstrapping(true);		
207			
208	// in order to create a purchase items	176	// in order to create a purchase items
209	// ESG with purchase part	177	// ESG with purchase part
210	var ServiceGuideDeliveryUnit v_SGDU := {};	178	var ServiceGuideDeliveryUnit v_SGDU := {};
211	var UInt32 v_Version := 1;	179	var UInt32 v_Version := 1;
212		180	
213	f_main_ctrl_InitStartEndTime(0, c_one_hour);	181	f_main_ctrl_InitStartEndTime(0, c_one_hour);

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

214			182	
215	var DateTime v_DateTime_StartTime := f_getDateTime(v_startTime);		183	var DateTime v_DateTime_StartTime := f_getDateTime(v_startTime);
216	var DateTime v_DateTime_EndTime := f_getDateTime(v_endTime);		184	var DateTime v_DateTime_EndTime := f_getDateTime(v_endTime);
217			185	
218	// TODO: Session Description fragment containing SDP for "Programme" and the service protection		186	// TODO: Session Description fragment containing SDP for "Programme" and the service protection
219	var SdpFragment v_Sdp := valueof(m_def_SdpFragment(187	var SdpFragment v_Sdp := valueof(m_def_SdpFragment(
220	PX_SDP_VIDEO_PROG_1_REF_ID,	=	188	PX_SDP_VIDEO_PROG_1_REF_ID,
221	PX_SDP_SERVICE_PROTECTION_S RTP		189	PX_SDP_SERVICE_PROTECTION_S RTP
222));	<>	190));
223			191	
224	var AccessTypeType v_AccessType := valueof(m_def_AccessType_bc_sdp(192	var AccessTypeType v_AccessType := valueof(m_def_AccessType_bc_sdp(
225	f_getAccessType(PX_BROADCAST_NETWORK_BEARER),	=	193	f_getAccessType(PX_BROADCAST_NETWORK_BEARER),
226	PX_SDP_VIDEO_PROG_1_REF_ID		194	PX_SDP_VIDEO_PROG_1_REF_ID
227));	<>	195));
228			196	
229	var PriceInfoType v_PriceInfo := valueof(m_PriceInfo(197	var PriceInfoType v_PriceInfo := valueof(m_PriceInfo(
230	1,	=	198	1,
231	PX_MONETARY_PRICE,		199	PX_MONETARY_PRICE,
232	PX_CURRENCY		200	PX_CURRENCY
233));	<>	201));
234			202	
235	var ServiceType v_Service := valueof(m_def_Service(203	var ServiceType v_Service := valueof(m_def_Service(
236	PX_SGDU_SERVICE_ID,	=	204	PX_SGDU_SERVICE_ID,
237	v_Version,		205	v_Version,
238	"PayTvChannel",		206	"PayTvChannel",
239	c_basicTv_ServiceType		207	c_basicTv_ServiceType
240));	<>	208));
241			209	
242	var ContentType v_Content := valueof(m_def_Content(210	var ContentType v_Content := valueof(m_def_Content(
243	PX_SGDU_CONTENT_ID_PROG_1,	=	211	PX_SGDU_CONTENT_ID_PROG_1,
244	v_Version,		212	v_Version,
245	"Programme",		213	"Programme",
246	PX_SGDU_SERVICE_ID,		214	PX_SGDU_SERVICE_ID,
247	v_DateTime_StartTime,		215	v_DateTime_StartTime,
248	v_DateTime_EndTime		216	v_DateTime_EndTime
249));	<>	217));
250			218	
251	var ScheduleType v_Schedule := valueof(m_def_Schedule(219	var ScheduleType v_Schedule := valueof(m_def_Schedule(
252	PX_SGDU_SCHEDULE_ID_PROG_1,	=	220	PX_SGDU_SCHEDULE_ID_PROG_1,
253	v_Version,		221	v_Version,
254	PX_SGDU_SERVICE_ID,		222	PX_SGDU_SERVICE_ID,
255	PX_SGDU_CONTENT_ID_PROG_1,		223	PX_SGDU_CONTENT_ID_PROG_1,
256	v_startTime, // change 01 (WK 6)		224	v_startTime, // change 01 (WK 6)
257	v_endTime // change 01 (WK 6)		225	v_endTime // change 01 (WK 6)
258));	<>	226));
259			227	
260	var AccessType v_Access := valueof(m_def_Access(228	var AccessType v_Access := valueof(m_def_Access(
261	PX_SGDU_ACCESS_ID_PROG_1,	=	229	PX_SGDU_ACCESS_ID_PROG_1,
262	v_Version,		230	v_Version,
263	v_AccessType,		231	v_AccessType,
264	PX_SGDU_SERVICE_ID,		232	PX_SGDU_SERVICE_ID,
265	PX_SGDU_SCHEDULE_ID_PROG_1,		233	PX_SGDU_SCHEDULE_ID_PROG_1,
266	c_class_SG		234	c_class_SG
267));	<>	235));
268	v_Access.KeyManagementSystems := {		236	v_Access.KeyManagementSystems := {
269	{		237	{
270	kmsType := 1, // 1 = oma-bcast-gba_u-mbms		238	kmsType := 1, // 1 = oma-bcast-gba_u-mbms
271	protectionType := 0, // Content protection only, as defined by the LTKM		239	protectionType := 0, // Content protection only, as defined by the LTKM
272	PermissionsIssuerURI := {		240	PermissionsIssuerURI := {
273	text_ := "http://" & PX_BSM_FQDN		241	text_ := "http://" & PX_BSM_FQDN
274	},		242	},
275	ProtectionKeyIDs := {		243	ProtectionKeyIDs := {
276	{		244	{
277	type_ := 0, // ProtectionKeyID = Key Domain ID concatenated with SEK/PEK ID		245	type_ := 0, // ProtectionKeyID = Key Domain ID concatenated with SEK/PEK ID
278	// key domain id = MCC MNC		246	// key domain id = MCC MNC
279	// sek id = key group key number		247	// sek id = key group key number
280	text_ := fx_bitstring2Base64(hex2bit(protectionKeyId)) // Base64 encoding		248	text_ := fx_bitstring2Base64(hex2bit(PX_MCC & PX_MNC & PX_KEY_GROUP & PX_P
281)		249)
282	},		250	},
283	TerminalBindingKeyID := omit		251	TerminalBindingKeyID := omit
284	}	=	252	}

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

285	};		253	};
286			254	
287	var PurchaseItemType v_PurchaseItem := valueof(m_def_PurchaseItem(255	var PurchaseItemType v_PurchaseItem := valueof(m_def_PurchaseItem(
288	PX_SGDU_PURCHASE_ITEM_ID,		256	PX_SGDU_PURCHASE_ITEM_ID,
289	v_Version,		257	v_Version,
290	PX_SGDU_PURCHASE_DATA_ID,		258	PX_SGDU_PURCHASE_DATA_ID,
291	PX_SGDU_SERVICE_ID,		259	PX_SGDU_SERVICE_ID,
292	"PurchaseItem1"		260	"PurchaseItem1"
293));		261));
294	v_PurchaseItem.Descriptions := {{lang := omit, text_:"Purchasable SRTPEncrypted service"}};		262	v_PurchaseItem.Descriptions := {{lang := omit, text_:"Purchasable SRTPEncrypted service"}};
295			263	
296	var PurchaseDataType v_PurchaseData := valueof(m_def_PurchaseData(264	var PurchaseDataType v_PurchaseData := valueof(m_def_PurchaseData(
297	PX_SGDU_PURCHASE_DATA_ID,		265	PX_SGDU_PURCHASE_DATA_ID,
298	v_Version,		266	v_Version,
299	{{omit,"Discount price available"}},		267	{{omit,"Discount price available"}},
300	PX_SGDU_PURCHASE_ITEM_ID,		268	PX_SGDU_PURCHASE_ITEM_ID,
301	PX_SGDU_PURCHASE_CHANNEL_ID,		269	PX_SGDU_PURCHASE_CHANNEL_ID,
302	v_PriceInfo		270	v_PriceInfo
303));		271));
304			272	
305	var PurchaseChannelType v_PurchaseChannel := valueof(m_def_PurchaseChannel(273	var PurchaseChannelType v_PurchaseChannel := valueof(m_def_PurchaseChannel(
306	PX_SGDU_PURCHASE_CHANNEL_ID,		274	PX_SGDU_PURCHASE_CHANNEL_ID,
307	v_Version,		275	v_Version,
308	"PurchaseChannel",		276	"PurchaseChannel",
309	PX_PURCHASE_URL //hint: should have this format -> "http://" & PX_BSM_FQDN & "/bsm/purc	<>	277	PX_PURCHASE_URL
310));	=	278));
311			279	
312	// CR 26/8: generic function for adding XML fragments		280	// CR 26/8: generic function for adding XML fragments
313	f_addXMLFragment(v_SGDU, {Service := v_Service});		281	f_addXMLFragment(v_SGDU, {Service := v_Service});
314	f_addXMLFragment(v_SGDU, {Content := v_Content});		282	f_addXMLFragment(v_SGDU, {Content := v_Content});
315	f_addXMLFragment(v_SGDU, {Schedule := v_Schedule});		283	f_addXMLFragment(v_SGDU, {Schedule := v_Schedule});
316	f_addXMLFragment(v_SGDU, {Access := v_Access});		284	f_addXMLFragment(v_SGDU, {Access := v_Access});
317	f_addXMLFragment(v_SGDU, {PurchaseItem := v_PurchaseItem});		285	f_addXMLFragment(v_SGDU, {PurchaseItem := v_PurchaseItem});
318	f_addXMLFragment(v_SGDU, {PurchaseChannel := v_PurchaseChannel});		286	f_addXMLFragment(v_SGDU, {PurchaseChannel := v_PurchaseChannel});
319	f_addXMLFragment(v_SGDU, {PurchaseData := v_PurchaseData});		287	f_addXMLFragment(v_SGDU, {PurchaseData := v_PurchaseData});
320	f_addSdpFragment(v_SGDU, v_Version, v_Sdp);		288	f_addSdpFragment(v_SGDU, v_Version, v_Sdp);
321			289	
322	f_main_ctrl_broadcastServiceGuide(v_SGDU, omit, vc_FluteUpdateID);	<>	290	f_main_ctrl_broadcastServiceGuide(v_SGDU, omit);
323		=	291	
324	f_main_ut_RunBCastApplication();		292	f_main_ut_RunBCastApplication();
325			293	
326	f_main_ut_CheckService("PayTvChannel");	<>		
327	f_main_ut_SelectService("PayTvChannel");			
328	f_main_ut_CheckContent("Programme with price information", v_DateTime_StartTime, v_DateTime			
329				
330	// change 2 (WK23): GBA authentication/authorization extensions for content protection test			
331	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registrat	=	294	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registrat
332	// and Service Request/Response is not allowed	<>		
333	f_main_ctrl_BSM_Procedures(true, true, false, true);		295	f_main_ctrl_MBMS_User_Service_Procedure(true);
334				
335	// postamble			
336	f_main_ut_PowerOff();			
337				
338	f_cf_BSF_down();			
339	f_cf_BSM_down();			
340	f_cf_DNS_down();			
341	f_cf_Broadcast_down();			
342	f_cf_UpperTester_down();			
343	f_terminateComponents();			
344	}			
345				
346	// change 3 (WK34): Service Protection: secure streaming service			
347	// change 6 (WK23): new service protection testcase			
348	testcase TC_BCAST_SERVPROT_101d() runs on BCastMainComponent system BCastPlatformComponent {			
349	// init components and ports			
350	f_createComponents();			
351	f_cf_Broadcast_up();			
352	f_cf_UpperTester_up();			
353	f_cf_DNS_up();			
354	f_cf_BSM_up();			
355	f_cf_BSF_up();			

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

356				
357	// preamble			
358	f_cf_enable_DNS();			
359	f_startCommonUtPreamble();			
360				
361	var hexstring protectionKeyId := PX_MCC & PX_MNC & PX_KEY_GROUP & PX_KEY_NUMBER;			
362	f_main_ctrl_InitProtectionKeysAndValues();			
363	f_main_ctrl_SetProtectionKeyId(protectionKeyId);			
364	f_main_ctrl_SetBootstrappingTransactionId(f_bsf_generateBtid());			
365	// GBA/BSF as parallel test component	=	296	// GBA/BSF as parallel test component
366	f_main_ctrl_GBA_U_Bootstrapping(true);		297	f_main_ctrl_GBA_U_Bootstrapping(true);
367			298	
368	// in order to create a purchase items	+-		
369	// ESG with purchase part			
370	var ServiceGuideDeliveryUnit v_SGDU := {};			
371	var UInt32 v_Version := 1;			
372				
373	f_main_ctrl_InitStartTime(0, c_one_hour);			
374				
375	var DateTime v_DateTime_StartTime := f_getDateTime(v_startTime);			
376	var DateTime v_DateTime_EndTime := f_getDateTime(v_endTime);			
377				
378	// TODO: Session Description fragment containing SDP for "Programme" and the service protection			
379	var SdpFragment v_Sdp := valueof(m_def_SdpFragment(
380	PX_SDP_VIDEO_PROG_1_REF_ID,			
381	PX_SDP_SERVICE_PROTECTION_S RTP			
382));			
383				
384	var AccessTypeType v_AccessType := valueof(m_def_AccessType_bc_sdp(
385	f_getAccessType(PX_BROADCAST_NETWORK_BEARER),			
386	PX_SDP_VIDEO_PROG_1_REF_ID			
387));			
388				
389	var PriceInfoType v_PriceInfo := valueof(m_PriceInfo(
390	1,			
391	PX_MONETARY_PRICE,			
392	PX_CURRENCY			
393));			
394				
395	var ServiceType v_Service := valueof(m_def_Service(
396	PX_SGDU_SERVICE_ID,			
397	v_Version,			
398	"PayTvChannel",			
399	c_basicTv_ServiceType			
400));			
401				
402	var ContentType v_Content := valueof(m_def_Content(
403	PX_SGDU_CONTENT_ID_PROG_1,			
404	v_Version,			
405	"Programme",			
406	PX_SGDU_SERVICE_ID,			
407	v_DateTime_StartTime,			
408	v_DateTime_EndTime			
409));			
410				
411	var ScheduleType v_Schedule := valueof(m_def_Schedule(
412	PX_SGDU_SCHEDULE_ID_PROG_1,			
413	v_Version,			
414	PX_SGDU_SERVICE_ID,			
415	PX_SGDU_CONTENT_ID_PROG_1,			
416	v_startTime,			// change 01 (WK 6)
417	v_endTime			// change 01 (WK 6)
418));			
419				
420	var AccessType v_Access := valueof(m_def_Access(
421	PX_SGDU_ACCESS_ID_PROG_1,			
422	v_Version,			
423	v_AccessType,			
424	PX_SGDU_SERVICE_ID,			
425	PX_SGDU_SCHEDULE_ID_PROG_1,			
426	c_class_SG			

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

427));			
428	v_Access.KeyManagementSystems := {			
429	{			
430	kmsType := 1, // 1 = oma-bcast-gba_u-mbms			
431	protectionType := 0, // Content protection only, as defined by the LTKM			
432	PermissionsIssuerURI := {			
433	text_ := PX_BSM_FQDN			
434	},			
435	ProtectionKeyIDs := {			
436	{			
437	type_ := 0, // ProtectionKeyID = Key Domain ID concatenated with SEK/PEK ID			
438	// key domain id = MCC MNC			
439	// sek id = key group key number			
440	text_ := fx_bitstring2Base64(hex2bit(protectionKeyId)) // Base64 encoding			
441	}			
442	},			
443	TerminalBindingKeyID := omit			
444	}			
445	};			
446				
447	var PurchaseItemType v_PurchaseItem := valueof(m_def_PurchaseItem(
448	PX_SGDU_PURCHASE_ITEM_ID,			
449	v_Version,			
450	PX_SGDU_PURCHASE_DATA_ID,			
451	PX_SGDU_SERVICE_ID,			
452	"PurchaseItem1"			
453));			
454	v_PurchaseItem.Descriptions := {{lang := omit, text_:"Purchasable SRTPencrypted service"}}			
455				
456	var PurchaseDataType v_PurchaseData := valueof(m_def_PurchaseData(
457	PX_SGDU_PURCHASE_DATA_ID,			
458	v_Version,			
459	{{omit,"Discount price available"}},			
460	PX_SGDU_PURCHASE_ITEM_ID,			
461	PX_SGDU_PURCHASE_CHANNEL_ID,			
462	v_PriceInfo			
463));			
464				
465	var PurchaseChannelType v_PurchaseChannel := valueof(m_def_PurchaseChannel(
466	PX_SGDU_PURCHASE_CHANNEL_ID,			
467	v_Version,			
468	"PurchaseChannel",			
469	"http://" & PX_BSM_FQDN & "/bsm/purchaseURL"			
470));			
471				
472	// CR 26/8: generic function for adding XML fragments			
473	f_addXMLFragment(v_SGDU, {Service := v_Service});			
474	f_addXMLFragment(v_SGDU, {Content := v_Content});			
475	f_addXMLFragment(v_SGDU, {Schedule := v_Schedule});			
476	f_addXMLFragment(v_SGDU, {Access := v_Access});			
477	f_addXMLFragment(v_SGDU, {PurchaseItem := v_PurchaseItem});			
478	f_addXMLFragment(v_SGDU, {PurchaseChannel := v_PurchaseChannel});			
479	f_addXMLFragment(v_SGDU, {PurchaseData := v_PurchaseData});			
480	f_addSdpFragment(v_SGDU, v_Version, v_Sdp);			
481				
482	f_main_ctrl_broadcastServiceGuide(v_SGDU, omit, vc_FluteUpdateID);			
483				
484	f_main_ut_RunBCastApplication();			
485				
486	f_main_ut_CheckService("PayTvChannel");	=	299	f_main_ut_CheckService("PayTvChannel");
487	f_main_ut_SelectService("PayTvChannel");		300	f_main_ut_SelectService("PayTvChannel");
488	f_main_ut_CheckContent("Programme with price information", v_DateTime_StartTime, v_DateTime_EndTime);		301	f_main_ut_CheckContent("Programme with price information", v_DateTime_StartTime, v_DateTime_EndTime);
489	f_main_ut_PurchaseService("PayTvChannel");	+-		
490				
491	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registration			
492	// and Service Request and Service Response are test purpose			
493	f_main_ctrl_BSM_Procedures(true, true, true, false);			
494		=	302	
495	// postamble		303	// postamble
496	f_main_ut_PowerOff();		304	f_main_ut_PowerOff();
497			305	

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

498	f_cf_BSF_down();		306	f_cf_BSF_down();
499	f_cf_BSM_down();		307	f_cf_BSM_down();
500	f_cf_DNS_down();		308	f_cf_DNS_down();
501	f_cf_Broadcast_down();		309	f_cf_Broadcast_down();
502	f_cf_UpperTester_down();		310	f_cf_UpperTester_down();
503	f_terminateComponents();		311	f_terminateComponents();
504	}	<>	312	}
505				
506	// change 3 (WK34): Service Protection: secure streaming service			
507	// change 6 (WK23): new service protection testcase			
508	testcase TC_BCAST_SERVPROT_101e() runs on BCastMainComponent system BCastPlatformComponent {			
509	// init components and ports			
510	f_createComponents();			
511	f_cf_Broadcast_up();			
512	f_cf_UpperTester_up();			
513	f_cf_DNS_up();			
514	f_cf_BSM_up();			
515	f_cf_BSF_up();			
516	f_cf_Mikey_up();			
517				
518	// preamble			
519	f_cf_enable_DNS();			
520	f_startCommonUtPreamble();			
521				
522	var hexstring v_keyDomainId := PX_MCC & PX_MNC;			
523	var hexstring v_mskId := PX_KEY_GROUP & PX_KEY_NUMBER;			
524	var hexstring v_protectionKeyId := v_keyDomainId & v_mskId;			
525	var octetstring v_ks_NAF := f_main_ctrl_InitProtectionKeysAndValues();			
526	f_main_ctrl_SetProtectionKeyId(v_protectionKeyId);			
527	var charstring v_btId := f_bsf_generateBtid();			
528	f_main_ctrl_SetBootstrappingTransactionId(v_btId);			
529	// Bcast Subscription Managemant as parallel test component for MBMS User Service Registrat			
530	// and Service Request/Response			
531	f_main_ctrl_BSM_Procedures(true, true, true, true);			
532	// GBA/BSF as parallel test component			
533	f_main_ctrl_GBA_U_Bootstrapping(true);			
534				
535	// in order to create a purchase items			
536	// ESG with purchase part			
537	var ServiceGuideDeliveryUnit v_SGDU := {};			
538	var UInt32 v_Version := 1;			
539				
540	f_main_ctrl_InitStartEndTime(0, c_one_hour);			
541				
542	var DateTime v_DateTime_StartTime := f_getDateTime(v_startTime);			
543	var DateTime v_DateTime_EndTime := f_getDateTime(v_endTime);			
544				
545	// TODO: Session Description fragment containing SDP for "Programme" and the service protec			
546	var SdpFragment v_Sdp := valueof(m_def_SdpFragment(
547	PX_SDP_VIDEO_PROG_1_REF_ID,			
548	PX_SDP_SERVICE_PROTECTION_SRTP			
549));			
550				
551	var AccessTypeType v_AccessType := valueof(m_def_AccessType_bc_sdp(
552	f_getAccessType(PX_BROADCAST_NETWORK_BEARER),			
553	PX_SDP_VIDEO_PROG_1_REF_ID			
554));			
555				
556	var PriceInfoType v_PriceInfo := valueof(m_PriceInfo(
557	1,			
558	PX_MONETARY_PRICE,			
559	PX_CURRENCY			
560));			
561				
562	var ServiceType v_Service := valueof(m_def_Service(
563	PX_SGDU_SERVICE_ID,			
564	v_Version,			
565	"PayTvChannel",			
566	c_basicTv_ServiceType			
567));			
568				

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

```
569     var ContentType v_Content := valueof(m_def_Content(  
570         PX_SGDU_CONTENT_ID_PROG_1,  
571         v_Version,  
572         "Programme",  
573         PX_SGDU_SERVICE_ID,  
574         v_DateTime_StartTime,  
575         v_DateTime_EndTime  
576     ));  
577  
578     var ScheduleType v_Schedule := valueof(m_def_Schedule(  
579         PX_SGDU_SCHEDULE_ID_PROG_1,  
580         v_Version,  
581         PX_SGDU_SERVICE_ID,  
582         PX_SGDU_CONTENT_ID_PROG_1,  
583         v_startTime, // change 01 (WK 6  
584         v_endTime // change 01 (WK 6  
585     ));  
586  
587     var AccessType v_Access := valueof(m_def_Access(  
588         PX_SGDU_ACCESS_ID_PROG_1,  
589         v_Version,  
590         v_AccessType,  
591         PX_SGDU_SERVICE_ID,  
592         PX_SGDU_SCHEDULE_ID_PROG_1,  
593         c_class_SG  
594     ));  
595     v_Access.KeyManagementSystems := {  
596     {  
597         kmsType := 1, // 1 = oma-bcast-gba_u-mbms  
598         protectionType := 0, // Content protection only, as defined by the LTKM  
599         PermissionsIssuerURI := {  
600             text_ := "http://" & PX_BSM_FQDN  
601         },  
602         ProtectionKeyIDs := {  
603             {  
604                 type_ := 0, // ProtectionKeyID = Key Domain ID concatenated with SEK/PEK ID  
605                 // key domain id = MCC || MNC  
606                 // sek id = key group || key number  
607                 text_ := fx_bitstring2Base64(hex2bit(v_protectionKeyId)) // Base64 encoding  
608             }  
609         },  
610         TerminalBindingKeyID := omit  
611     }  
612 };  
613  
614     var PurchaseItemType v_PurchaseItem := valueof(m_def_PurchaseItem(  
615         PX_SGDU_PURCHASE_ITEM_ID,  
616         v_Version,  
617         PX_SGDU_PURCHASE_DATA_ID,  
618         PX_SGDU_SERVICE_ID,  
619         "PurchaseItem1"  
620     ));  
621     v_PurchaseItem.Descriptions := {{lang := omit, text_ := "Purchasable SRTPencrypted service"}}  
622  
623     var PurchaseDataType v_PurchaseData := valueof(m_def_PurchaseData(  
624         PX_SGDU_PURCHASE_DATA_ID,  
625         v_Version,  
626         {{omit, "Discount price available"}},  
627         PX_SGDU_PURCHASE_ITEM_ID,  
628         PX_SGDU_PURCHASE_CHANNEL_ID,  
629         v_PriceInfo  
630     ));  
631  
632     var PurchaseChannelType v_PurchaseChannel := valueof(m_def_PurchaseChannel(  
633         PX_SGDU_PURCHASE_CHANNEL_ID,  
634         v_Version,  
635         "PurchaseChannel",  
636         "http://" & PX_BSM_FQDN & "/bsm/purchaseURL"  
637     ));  
638  
639     f_addXMLFragment(v_SGDU, {Service := v_Service});
```

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

640	f_addXMLFragment(v_SGDU, {Content := v_Content});	
641	f_addXMLFragment(v_SGDU, {Schedule := v_Schedule});	
642	f_addXMLFragment(v_SGDU, {Access := v_Access});	
643	f_addXMLFragment(v_SGDU, {PurchaseItem := v_PurchaseItem});	
644	f_addXMLFragment(v_SGDU, {PurchaseChannel := v_PurchaseChannel});	
645	f_addXMLFragment(v_SGDU, {PurchaseData := v_PurchaseData});	
646	f_addSdpFragment(v_SGDU, v_Version, v_Sdp);	
647		
648	f_main_ctrl_broadcastServiceGuide(v_SGDU, omit, vc_FluteUpdateID);	
649		
650	f_main_ut_RunBCastApplication();	
651		
652	f_main_ut_CheckService("PayTvChannel");	
653	f_main_ut_SelectService("PayTvChannel");	
654	f_main_ut_CheckContent("Programme with price information", v_DateTime_StartTime, v_DateTime_EndTime);	
655	f_main_ut_PurchaseService("PayTvChannel");	
656		
657	// send via UDP the LTKM	
658	var UInt8 v_PolicyNumber := 0;	
659	var octetstring v_randValue := '6813b3758c1cfe138d8b2ba886d3f449'O;	
660	var octetstring v_csbId := '00000000'O;	
661	var Oct4 v_ssrc := '7BE6AC2E'O;	
662		
663	var ListOfSrtpId v_SrtpIds := {}; // empty list	
664	var SrtpId v_SrtpId := valueof(m_def_SrtpId(v_PolicyNumber, v_ssrc, '00000000'O));	
665	f_addSrtpId(v_SrtpIds, v_SrtpId);	
666	//v_SrtpId := valueof(m_def_SrtpId(v_PolicyNumber, '00000000'O, '00000000'O));	
667	//f_addSrtpId(v_SrtpIds, v_SrtpId);	
668		
669	var MikeyHeaderPayload v_mikeyheader := valueof(m_def_LTKMHeader(GeneralExt, v_csbId, v_SrtpId));	
670		
671	var ListOfPayload v_payloads := {}; // empty list	
672		
673	var KeyId v_keyIdKeyDomianId := valueof(m_def_KeyId_KeyDomainId(hex2oct(v_keyDomainId)));	
674	var KeyId v_keyIdMskId := valueof(m_def_KeyId_MSKId(hex2oct(v_mskId)));	
675	var GeneralExtension v_ExtMBMS := valueof(m_def_GeneralExtension(KEY_ID, {keyIds := {v_keyIdKeyDomianId, v_keyIdMskId}}));	
676	f_addMikeyPayload(v_payloads, {generalExtension := v_ExtMBMS});	
677		
678	var BcastExtension v_bcastExt := valueof(m_def_BcastExtension_LTKM(m_def_LTKMManagementData));	
679	var GeneralExtension v_ExtBCAST := valueof(m_def_GeneralExtension(OMA_BCAST, {bcastExtension := v_bcastExt}));	
680	f_addMikeyPayload(v_payloads, {generalExtension := v_ExtBCAST});	
681		
682	var integer v_ntpTimestamp := f_getNTPTime();	
683	var Timestamp v_Timestamp := valueof(m_def_TimestampPayloadData(NTP.UTC, v_ntpTimestamp));	
684	f_addMikeyPayload(v_payloads, {timestamp := v_Timestamp});	
685		
686	var Rand v_Rand := valueof(m_def_RandPayloadData(v_randValue));	
687	f_addMikeyPayload(v_payloads, {rand := v_Rand});	
688		
689	var Id v_Idi := valueof(m_def_IdPayloadData(URI, PX_BSM_FQDN));	
690	f_addMikeyPayload(v_payloads, {id := v_Idi});	
691		
692	var Id v_Idr := valueof(m_def_IdPayloadData(NAI, v_btId));	
693	f_addMikeyPayload(v_payloads, {id := v_Idr});	
694		
695	//	var ListOfSrtpPolicyParam v_SrtpPolicyParams := {}; //empty list
696	//	var SrtpPolicyParameter v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_ENC_OFF_ON, 1, {srtpPrf := v_SrtpPrf}));
697	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
698	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SESSION_ENCR_KEY_LEN, 1, {srtpPrf := v_SrtpPrf}));
699	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
700	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_AUTH_ALGORITHM, 1, {srtpPrf := v_SrtpPrf}));
701	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
702	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SESSION_AUTH_KEY_LEN, 1, {srtpPrf := v_SrtpPrf}));
703	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
704	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SESSION_SALT_KEY_LEN, 1, {srtpPrf := v_SrtpPrf}));
705	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
706	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_PRF, 1, {srtpPrf := v_SrtpPrf}));
707	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
708	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(KEY_DERIVATION_RATE, 1, {srtpPrf := v_SrtpPrf}));
709	//	f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
710	//	v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_ENCR_OFF_ON, 1, {srtpPrf := v_SrtpPrf}));

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

```
711 //      f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
712 //      v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTCP_ENCR_OFF_ON, 1);
713 //      f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
714 //      v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SENDER_FEC_ORDER, 1);
715 //      f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
716 //      v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_AUTH_OFF_ON, 1);
717 //      f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
718 //      v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(AUTH_TAG_LENGTH, 1);
719 //      f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
720 //      v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_PREFIX_LENGTH, 1);
721 //      f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
722 //      var SecurityPolicy v_SecurityPolicy := valueof(m_def_SecurityPolicy4Srtp(v_SrtpPolicyParams));
723 //      f_addMikeyPayload(v_payloads, {securityPolicy := v_SecurityPolicy});
724
725 // preparations for encrypting of LTKM included in KEMAC
726 var Interval v_interval := valueof(m_def_Interval('00'O, '0f'O));
727 var Key v_keyData := valueof(m_def_KeyData4LTKM(KV_INTERVAL, PX_LTKM, {interval := v_interval}));
728 var MikeyPayload v_keyDataPayload := valueof(m_def_Payload(LastPayload, {key := v_keyData}));
729 // PSK is MUK is ks_(ext/int)_NAF (depending of used GBA algorithm and UICC supports MBMS)
730 var octetstring v_encrKeyData := fx_encrypt(v_csId, v_randValue, v_ntpTimestamp, fx_mikeyHeader);
731 // hash calculation is done in the Mikey Codec
732 var KeyDataTransport v_kemac := valueof(m_def_KemacPayload(AES_CM_128, v_encrKeyData, HMAC_SHA_1));
733 f_addMikeyPayload(v_payloads, {keyDataTransport := v_kemac});
734
735 // PSK needed for hash calculation
736 var MikeyMessage v_sendMikey := valueof(m_def_Mikey(v_mikeyheader, v_payloads, {usedPSK := v_mikeyHeader}));
737
738 f_main_ctrl_sendMikey(v_sendMikey);
739
740 // postamble
741 f_main_ut_PowerOff();
742
743 f_cf_Mikey_down();
744 f_cf_BSF_down();
745 f_cf_BSM_down();
746 f_cf_DNS_down();
747 f_cf_Broadcast_down();
748 f_cf_UpperTester_down();
749 f_terminateComponents();
750 }
751
752 // change 3 (WK34): Service Protection: secure streaming service
753 testcase TC_BCAST_SERVPROT_101f() runs on BCastMainComponent system BCastPlatformComponent {
754   // init components and ports
755   f_createComponents();
756   f_cf_Broadcast_up();
757   f_cf_Secure_Data_up();
758   f_cf_Data_up();
759   f_cf_UpperTester_up();
760   f_cf_DNS_up();
761   f_cf_BSM_up();
762   f_cf_BSF_up();
763   f_cf_Mikey_up();
764
765   // preamble
766   f_cf_enable_DNS();
767   f_startCommonUtPreamble();
768
769   var charstring dstIP := PX_DATA_STREAMING_DESTINATION_IP_1;
770
771   var hexstring v_keyDomainId := PX_MCC & PX_MNC;
772   var hexstring v_mskId := PX_KEY_GROUP & PX_KEY_NUMBER;
773   var hexstring v_protectionKeyId := v_keyDomainId & v_mskId;
774   var octetstring v_ks_NAF := f_main_ctrl_InitProtectionKeysAndValues();
775   f_main_ctrl_SetProtectionKeyId(v_protectionKeyId);
776   var charstring v_btid := f_bsf_generateBtid();
777   f_main_ctrl_SetBootstrappingTransactionId(v_btid);
778   // Bcast Subscription Managemant as parallel test component for MBMS User Service Registration
779   // and Service Request/Response
780   f_main_ctrl_BSM_Procedures(true, true, true, true);
781   // GBA/BSF as parallel test component
```

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

782	f_main_ctrl_GBA_U_Bootstrapping(true);	
783		
784	// in order to create a purchase items	
785	// ESG with purchase part	
786	var ServiceGuideDeliveryUnit v_SGDU := {};	
787	var UInt32 v_Version := 1;	
788		
789	f_main_ctrl_InitStartEndTime(0, c_one_hour);	
790		
791	var DateTime v_DateTime_StartTime := f_getDateTime(v_startTime);	
792	var DateTime v_DateTime_EndTime := f_getDateTime(v_endTime);	
793		
794	// TODO: SGDD should be extended with BSM selector > SDP extension for STKM need the	
795	// reference to the BSM selector as declared in the SGDD	
796	// SGDU access fragment need the reference to thd BSM selector as well	
797		
798	var SdpFragment v_Sdp := valueof(m_def_SdpFragment(
799	PX_SDP_VIDEO_PROG_1_REF_ID,	
800	PX_SDP_SERVICE_PROTECTION_S RTP	
801));	
802		
803	var AccessTypeType v_AccessType := valueof(m_def_AccessType_bc_sdp(
804	f_getAccessType(PX_BROADCAST_NETWORK_BEARER),	
805	PX_SDP_VIDEO_PROG_1_REF_ID	
806));	
807		
808	var PriceInfoType v_PriceInfo := valueof(m_PriceInfo(
809	1,	
810	PX_MONETARY_PRICE,	
811	PX_CURRENCY	
812));	
813		
814	var ServiceType v_Service := valueof(m_def_Service(
815	PX_SGDU_SERVICE_ID,	
816	v_Version,	
817	"PayTvChannel",	
818	c_basicTv_ServiceType	
819));	
820		
821	var ContentType v_Content := valueof(m_def_Content(
822	PX_SGDU_CONTENT_ID_PROG_1,	
823	v_Version,	
824	"Programme",	
825	PX_SGDU_SERVICE_ID,	
826	v_DateTime_StartTime,	
827	v_DateTime_EndTime	
828));	
829		
830	var ScheduleType v_Schedule := valueof(m_def_Schedule(
831	PX_SGDU_SCHEDULE_ID_PROG_1,	
832	v_Version,	
833	PX_SGDU_SERVICE_ID,	
834	PX_SGDU_CONTENT_ID_PROG_1,	
835	v_startTime,	// change 01 (WK 6)
836	v_endTime	// change 01 (WK 6)
837));	
838		
839	var AccessType v_Access := valueof(m_def_Access(
840	PX_SGDU_ACCESS_ID_PROG_1,	
841	v_Version,	
842	v_AccessType,	
843	PX_SGDU_SERVICE_ID,	
844	PX_SGDU_SCHEDULE_ID_PROG_1,	
845	c_class_SG	
846));	
847	v_Access.KeyManagementSystems := {	
848	{	
849	kmsType := 1, // 1 = oma-bcast-gba_u-mbms	
850	protectionType := 0, // Content protection only, as defined by the LTKM	
851	PermissionsIssuerURI := {	
852	text_ := "http://" & PX_BSM_FQDN	

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

853	},	
854	ProtectionKeyIDs := {	
855	{	
856	type_ := 0, // ProtectionKeyID = Key Domain ID concatenated with SEK/PEK ID	
857	// key domain id = MCC MNC	
858	// sek id = key group key number	
859	text_ := fx_bitstring2Base64(hex2bit(v_protectionKeyId)) // Base64 encoding	
860	}	
861	},	
862	TerminalBindingKeyID := omit	
863	}	
864	};	
865	// TODO: set BSM selector like in SGDD	
866	// v_Access.ServiceClass.ReferredSGInfo.BSMSelector	
867		
868	var PurchaseItemType v_PurchaseItem := valueof(m_def_PurchaseItem(
869	PX_SGDU_PURCHASE_ITEM_ID,	
870	v_Version,	
871	PX_SGDU_PURCHASE_DATA_ID,	
872	PX_SGDU_SERVICE_ID,	
873	"PurchaseItem1"	
874));	
875	v_PurchaseItem.Descriptions := {{lang := omit, text_:"Purchasable SRTPencrypted service"}}	
876		
877	var PurchaseDataType v_PurchaseData := valueof(m_def_PurchaseData(
878	PX_SGDU_PURCHASE_DATA_ID,	
879	v_Version,	
880	{{omit,"Discount price available"}},	
881	PX_SGDU_PURCHASE_ITEM_ID,	
882	PX_SGDU_PURCHASE_CHANNEL_ID,	
883	v_PriceInfo	
884));	
885		
886	var PurchaseChannelType v_PurchaseChannel := valueof(m_def_PurchaseChannel(
887	PX_SGDU_PURCHASE_CHANNEL_ID,	
888	v_Version,	
889	"PurchaseChannel",	
890	"http://" & PX_BSM_FQDN & "/bsm/purchaseURL"	
891));	
892		
893	f_addXMLFragment(v_SGDU, {Service := v_Service});	
894	f_addXMLFragment(v_SGDU, {Content := v_Content});	
895	f_addXMLFragment(v_SGDU, {Schedule := v_Schedule});	
896	f_addXMLFragment(v_SGDU, {Access := v_Access});	
897	f_addXMLFragment(v_SGDU, {PurchaseItem := v_PurchaseItem});	
898	f_addXMLFragment(v_SGDU, {PurchaseChannel := v_PurchaseChannel});	
899	f_addXMLFragment(v_SGDU, {PurchaseData := v_PurchaseData});	
900	f_addSdpFragment(v_SGDU, v_Version, v_Sdp);	
901		
902	f_main_ctrl_broadcastServiceGuide(v_SGDU, omit, vc_FluteUpdateID);	
903		
904	f_main_ut_RunBCastApplication();	
905		
906	f_main_ut_CheckService("PayTvChannel");	
907	f_main_ut_SelectService("PayTvChannel");	
908	f_main_ut_CheckContent("Programme with price information", v_DateTime_StartTime, v_DateTime_EndTime);	
909	f_main_ut_PurchaseService("PayTvChannel");	
910		
911	// preparation for LTKM	
912	var UInt8 v_PolicyNumber := 0;	
913	var octetstring v_randValue := '6813b3758c1cfe138d8b2ba886d3f449'O;	
914	var octetstring v_csbId := '00000000'O;	
915	var Oct4 v_ssrc := '7BE6AC2E'O;	
916		
917	var ListOfSrtpId v_SrtpIds := {}; // empty list	
918	var SrtpId v_SrtpId := valueof(m_def_SrtpId(v_PolicyNumber, v_ssrc, '00000000'O));	
919	f_addSrtpId(v_SrtpIds, v_SrtpId);	
920		
921	var MikeyHeaderPayload v_mikeyheader := valueof(m_def_LTKMHeader(GeneralExt, v_csbId, v_SrtpId));	
922		
923	var ListOfPayload v_payloads4ltkm := {}; // empty list	

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

```
924
925     var KeyId v_keyIdKeyDomianId := valueof(m_def_KeyId_KeyDomainId(hex2oct(v_keyDomainId)));
926     var KeyId v_keyIdMskId := valueof(m_def_KeyId_MSKId(hex2oct(v_mskId)));
927     var GeneralExtension v_ExtMBMS := valueof(m_def_GeneralExtension(KEY_ID, {keyIds := {v_keyIdKeyDomianId, v_keyIdMskId}}));
928     f_addMikeyPayload(v_payloads4ltkm, {generalExtension := v_ExtMBMS});
929
930     var BcastExtension v_bcastExt := valueof(m_def_BcastExtension_LTKM(m_def_LTKMManagementData));
931     var GeneralExtension v_ExtBCAST := valueof(m_def_GeneralExtension(OMA_BCAST, {bcastExtension := v_bcastExt}));
932     f_addMikeyPayload(v_payloads4ltkm, {generalExtension := v_ExtBCAST});
933
934     var integer v_ntpTimestamp := f_getNTPTime();
935     var Timestamp v_Timestamp := valueof(m_def_TimestampPayloadData(NTP.UTC, v_ntpTimestamp));
936     f_addMikeyPayload(v_payloads4ltkm, {timestamp := v_Timestamp});
937
938     var Rand v_Rand := valueof(m_def_RandPayloadData(v_randValue));
939     f_addMikeyPayload(v_payloads4ltkm, {rand := v_Rand});
940
941     var Id v_Idi := valueof(m_def_IdPayloadData(URI, PX_BSM_FQDN));
942     f_addMikeyPayload(v_payloads4ltkm, {id := v_Idi});
943
944     var Id v_Idr := valueof(m_def_IdPayloadData(NAI, v_btId));
945     f_addMikeyPayload(v_payloads4ltkm, {id := v_Idr});
946
947     // var ListOfSrtpPolicyParam v_SrtpPolicyParams := {}; //empty list
948     // var SrtpPolicyParameter v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter);
949     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
950     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SESSION_ENCR_KEY_LENGTH, 1, {srtpEncrKeyLength := 1}));
951     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
952     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_AUTH_ALGORITHM, 1, {srtpAuthAlgorithm := 1}));
953     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
954     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SESSION_AUTH_KEY_LENGTH, 1, {srtpAuthKeyLength := 1}));
955     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
956     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SESSION_SALT_KEY_LENGTH, 1, {srtpSaltKeyLength := 1}));
957     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
958     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_PR_F, 1, {srtpPrF := 1}));
959     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
960     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(KEY_DERIVATION_RATE, 1, {keyDerivationRate := 1}));
961     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
962     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_ENCR_OFF_ON, 1, {srtpEncrOffOn := 1}));
963     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
964     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_ENCR_OFF_ON, 1, {srtpEncrOffOn := 1}));
965     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
966     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SENDER_FEC_ORDER, 1, {senderFecOrder := 1}));
967     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
968     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_AUTH_OFF_ON, 1, {srtpAuthOffOn := 1}));
969     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
970     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(AUTH_TAG_LENGTH, 1, {authTagLength := 1}));
971     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
972     // v_SrtpPolicyParam := valueof(m_def_SrtpPolicyParameter(SRTP_PREFIX_LENGTH, 1, {srtpPrefixLength := 1}));
973     // f_addSrtpPolicyParameter(v_SrtpPolicyParams, v_SrtpPolicyParam);
974     // var SecurityPolicy v_SecurityPolicy := valueof(m_def_SecurityPolicy4Srtp);
975     // f_addMikeyPayload(v_payloads, {securityPolicy := v_SecurityPolicy});
976
977     // preparations for encrypting of LTKM included in KEMAC
978     var Interval v_interval := valueof(m_def_Interval(PX_TS_LOW, PX_TS_HIGH));
979     var Key v_keyData := valueof(m_def_KeyData4LTKM(KV_INTERVAL, PX_LTKM, {interval := v_interval}));
980     var MikeyPayload v_keyDataPayload := valueof(m_def_Payload(LastPayload, {key := v_keyData}));
981     // PSK is MUK is ks_(ext/int)_NAF (depending of used GBA algorithm and UICC supports MBMS)
982     var octetstring v_encrKeyData := fx_encrypt(v_csbId, v_randValue, v_ntpTimestamp, fx_mikeyHeader, v_keyDataPayload);
983     // hash calculation is done in the Mikey Codec
984     var KeyDataTransport v_kemac := valueof(m_def_KemacPayload(AES_CM_128, v_encrKeyData, HMAC_SHA_1));
985     f_addMikeyPayload(v_payloads4ltkm, {keyDataTransport := v_kemac});
986
987     // PSK needed for hash calculation
988     var MikeyMessage v_sendMikeyLTKM := valueof(m_def_Mikey(v_mikeyheader, v_payloads4ltkm, {useLTKM := true}));
989
990     // preparation for STKMs
991     v_mikeyheader := valueof(m_def_STKMHeader(GeneralExt, v_csbId));
992
993     var ListOfPayload v_payloads4stkm := {}; // empty list
994
```

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

995	var octetstring v_mtkId := PX_MTK_ID_START;	
996	var KeyId v_keyIdMtkId := valueof(m_def_KeyId MTKId(v_mtkId));	
997	v_ExtMBMS := valueof(m_def_GeneralExtension(KEY_ID, {keyIds := {v_keyIdKeyDomianId, v_keyId	
998	f_addMikeyPayload(v_payloads4stkm, {generalExtension := v_ExtMBMS});	
999		
1000	v_bcastExt := valueof(m_def_BcastExtension_STKM(m_def_STKMManagementData()));	
1001	v_ExtBCAST := valueof(m_def_GeneralExtension(OMA_BCAST, {bcastExtension := v_bcastExt}));	
1002	f_addMikeyPayload(v_payloads4stkm, {generalExtension := v_ExtBCAST});	
1003		
1004	v_Timestamp := valueof(m_def_TimestampPayloadData(COUNTER, oct2int(PX_TS_LOW) + 1));	
1005	f_addMikeyPayload(v_payloads4stkm, {timestamp := v_Timestamp});	
1006		
1007	// preparations for encrypting of STKM included in KEMAC	
1008	// PSK is key of LTKM	
1009	var octetstring v_salt := fx_mikeySaltKey(v_csbId, v_randValue, PX_LTKM, PX_SALT_KEY_LENGTH	
1010	v_keyData := valueof(m_def_KeyData4STKM(PX_S RTP_MASTER_KEY_START, v_salt));	
1011	v_keyDataPayload := valueof(m_def_Payload(LastPayload, {key := v_keyData}));	
1012		
1013	v_encrKeyData := fx_encrypt(v_csbId, v_randValue, v_ntpTimestamp, fx_mikeyPayload2Oct(v_key	
1014	// hash calculation is done in the Mikey Codec	
1015	v_kemac := valueof(m_def_KemacPayload(AES_CM_128, v_encrKeyData, HMAC_SHA_1_160));	
1016	f_addMikeyPayload(v_payloads4stkm, {keyDataTransport := v_kemac});	
1017		
1018	// PSK and RAND from LTKM needed for hash calculation	
1019	// RAND payload is not send again in STKM	
1020	var MikeyMessage v_sendMikeySTKM := valueof(m_def_Mikey(v_mikeyheader, v_payloads4stkm, {us	
1021		
1022	// generates all keys delivered in STKM	
1023	var ListOfSTKMKeys v_stkmKeys := {};	
1024	var integer numOfKeys := 15;	
1025	for (var integer i := 0; i < numOfKeys; i := i + 1) {	
1026	v_stkmKeys[i] := {int2oct((oct2int(PX_S RTP_MASTER_KEY_START) + i), 16), v_salt}	
1027	}	
1028		
1029	// generates remaining STKMs (updates the timestamp in every STKM; replaces the master key	
1030	var ListOfMikeyMessage v_stkms := {v_sendMikeySTKM};	
1031	var integer renewalOfKeys := 10; // the master key changes every 10th STKM	
1032	for (var integer i := 1; i < numOfKeys * renewalOfKeys; i := i + 1) {	
1033	v_stkms[i] := f_updateTSinSTKM(v_stkms[i - 1]);	
1034	if (i mod renewalOfKeys == 0) {	
1035	v_stkms[i] := f_updateKEYinSTKM(v_csbId, v_ntpTimestamp, v_stkmKeys[i / renewalOfKe	
1036	}	
1037	}	
1038		
1039	// sends LTKM	
1040	f_main_crtl_sendMikey(v_sendMikeyLTKM);	
1041		
1042	f_main_data_secure_setKeys(v_stkmKeys);	
1043		
1044	f_main_data_secure_startStreamingFile(0, PX_FILE_VIDEO_PROG1, PX_SDP_SERVICE_PROTECTION_SRT	
1045		
1046	// sends STKMs	
1047	timer v_nextSTKMTimer := 1.0;	
1048	var integer v_currentKeyNumber := 0;	
1049	for (var integer i := 0; i < numOfKeys * renewalOfKeys; i := i + 1) {	
1050	// after sending 10 STKMs the next key must be used	
1051	if (i mod 10 == 0) {	
1052	f_main_data_secure_activateKey(v_stkmKeys[v_currentKeyNumber], v_currentKeyNumber);	
1053	v_currentKeyNumber := v_currentKeyNumber + 1;	
1054	}	
1055	v_nextSTKMTimer.start;	
1056	f_main_crtl_broadcastMikey(v_stkms[i]);	
1057	v_nextSTKMTimer.timeout;	
1058	}	
1059		
1060	f_main_data_secure_stopStreaming(0);	
1061		
1062	// postamble	
1063	f_main_ut_PowerOff();	
1064		
1065	f_cf_Mikey_down();	

Datei: AtsBCast_ServiceProtectionTests.ttcn (Fortsetzung)

1066	f_cf_BSF_down();			
1067	f_cf_BSM_down();			
1068	f_cf_DNS_down();			
1069	f_cf_Data_down();			
1070	f_cf_Secure_Data_down();			
1071	f_cf_Broadcast_down();			
1072	f_cf_UpperTester_down();			
1073	f_terminateComponents();			
1074	}			
1075	}	=	313	}
1076		<>		
1077	// change 3 (WK34): Service Protection: secure streaming service			
1078	group Function {			
1079	/**			
1080	*			
1081	* @desc increases TS value by 1 in STKM			
1082	* @param p_currentKey			
1083	* @param p_stkm			
1084	* @return			
1085	* @verdict			
1086	*/			
1087	function f_updateTSinSTKM(in MikeyMessage p_stkm) return MikeyMessage {			
1088	var integer v_size := sizeof(p_stkm.payloads);			
1089	var MikeyMessage v_mikey := p_stkm;			
1090				
1091	// updates the timestamp			
1092	for (var integer i := 0; i < v_size; i := i + 1) {			
1093	if (ischosen(v_mikey.payloads[i].payloadData.timestamp)) {			
1094	v_mikey.payloads[i].payloadData.timestamp.tsValue := v_mikey.payloadData			
1095	i := v_size + 1; // stops the loop			
1096	}			
1097	}			
1098				
1099	return v_mikey;			
1100	}			
1101				
1102	/**			
1103	*			
1104	* @desc replaces the Key Data in STKM			
1105	* @param p_currentKey			
1106	* @param p_stkm			
1107	* @return			
1108	* @verdict			
1109	*/			
1110	function f_updateKEYinSTKM(in octetstring p_csbId, in integer p_timestamp, in SecureStreamingKe			
1111	// the key to be replaced is stored in the encrypted key data of the KEMAC payload			
1112	// KEMAC payload is always the last payload in an STKM			
1113	p_stkm.payloads[sizeof(p_stkm.payloads) - 1].payloadData.keyDataTransport.encrData := fx_er			
1114	return p_stkm;			
1115	}			
1116	}			
1117				
1118	// change 3 (WK34): Service Protection: secure streaming service			
1119	group externalFunctions {			
1120	external function fx_mikeyPayload2Oct(in MikeyPayload p_payload) return octetstring;			
1121	external function fx_encrypt(in octetstring p_csbId, in octetstring p_rand, in integer p_timest			
1122	external function fx_decrypt(in octetstring p_csbId, in octetstring p_rand, in integer p_timest			
1123	external function fx_mikeySaltKey(in octetstring p_csbid, in octetstring p_rand, in octetstring			
1124	external function fx_oct2mikeyPayload(in octetstring p_mikeyPayload) return MikeyPayload;			
1125	}			
1126	}	=	314	}
1127	}		315	}

Datei: AtsBCast_TestConfiguration_Functions.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies functions for the test configuratoin 9 */ 10 module AtsBCast_TestConfiguration_Functions{ 11 import from AtsBCast_Main_Functions all; 12 import from AtsBCast_TestSystem { 13 type BCastMainComponent; 14 } 15 import from LibBCast_Interface { 16 type UpperTesterComponent; 17 type BCastNetworkControlComponent; 18 type BCastNetworkDataComponent; 19 } 20 import from LibBCast_ModuleParameters { 21 modulepar PX_BROADCAST_NETWORK_BEARER; 22 modulepar PX_INTERACTION_NETWORK_BEARER; 23 } 24 // change 1 (WK18): for content protection tests 25 import from LibCommon_DNS { 26 group DNS_Configuration_and_Components; 27 group DNS_Functions; 28 } 29 // change 1 (WK18): for content protection tests 30 // hint: (parallel) component for GBA 31 import from LibCommon_GBA_BSF { 32 group BSF_Configuration_and_Components; 33 group BSF_Functions; 34 } 35 // change 1 (WK18): for content protection tests 36 // hint: (parallel) component for MBMS user service registration 37 import from LibCommon_BSM { 38 group BSM_Configuration_and_Components; 39 group BSM_Functions; 40 } 41 // change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...) 42 import from LibCommon_Mikey { 43 group Mikey_Configuration_and_Components; 44 }</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies functions for the test configuratoin 9 */ 10 module AtsBCast_TestConfiguration_Functions{ 11 import from AtsBCast_Main_Functions all; 12 import from AtsBCast_TestSystem { 13 type BCastMainComponent; 14 } 15 import from LibBCast_Interface { 16 type UpperTesterComponent; 17 type BCastNetworkControlComponent; 18 type BCastNetworkDataComponent; 19 } 20 import from LibBCast_ModuleParameters { 21 modulepar PX_BROADCAST_NETWORK_BEARER; 22 modulepar PX_INTERACTION_NETWORK_BEARER; 23 } 24 // change 1 (WK18): for content protection tests 25 import from LibCommon_DNS { 26 group DNS_Configuration_and_Components; 27 group DNS_Functions; 28 } 29 // change 1 (WK18): for content protection tests 30 // hint: (parallel) component for GBA 31 import from LibCommon_GBA_BSF { 32 group BSF_Configuration_and_Components; 33 group BSF_Functions; 34 } 35 // change 1 (WK18): for content protection tests 36 // hint: (parallel) component for MBMS user service registration 37 import from LibCommon_BSM { 38 group BSM_Configuration_and_Components; 39 group BSM_Functions; 40 }</pre>
<pre>45 46 /** 47 * 48 * @desc 49 * This functions creates all test components that may be potentially used in BCast test 50 * suite. Note that that the execution of behavior on these test components is handled 51 * by the ATS main functions, i.e., after their creation test components do not yet produce 52 * any test behavior. 53 * @see 54 * AtsBCast_TestSystem.BCastMainComponent 55 * @see 56 * AtsBCast_TestSystem.BCastNetworkControlComponent 57 * @see 58 * AtsBCast_TestSystem.UpperTesterComponent 59 * @see 60 * AtsBCast_TestSystem.SimulationComponent 61 */ 62 function f_createComponents() runs on BCastMainComponent { 63 vc_CTRL := BCastNetworkControlComponent.create ("BCast Network Control") alive; 64 vc_DATA := BCastNetworkDataComponent.create ("BCast Network Data") alive; 65 vc_UTC := UpperTesterComponent.create ("Upper Tester") alive; 66 67 // change 1 (WK18): for content protection tests 68 vc_DNS := DNSServerComponent.create("DNS Server Simulation") alive; 69 70 // change 1 (WK18): for content protection tests 71 vc_BSF := BSFServerComponent.create("BSF Server Simulation") alive;</pre>	=	<pre>41 42 /** 43 * 44 * @desc 45 * This functions creates all test components that may be potentially used in BCast test 46 * suite. Note that that the execution of behavior on these test components is handled 47 * by the ATS main functions, i.e., after their creation test components do not yet produce 48 * any test behavior. 49 * @see 50 * AtsBCast_TestSystem.BCastMainComponent 51 * @see 52 * AtsBCast_TestSystem.BCastNetworkControlComponent 53 * @see 54 * AtsBCast_TestSystem.UpperTesterComponent 55 * @see 56 * AtsBCast_TestSystem.SimulationComponent 57 */ 58 function f_createComponents() runs on BCastMainComponent { 59 vc_CTRL := BCastNetworkControlComponent.create ("BCast Network Control") alive; 60 vc_DATA := BCastNetworkDataComponent.create ("BCast Network Data") alive; 61 vc_UTC := UpperTesterComponent.create ("Upper Tester") alive; 62 63 // change 1 (WK18): for content protection tests 64 vc_DNS := DNSServerComponent.create("DNS Server Simulation") alive; 65 66 // change 1 (WK18): for content protection tests 67 vc_BSF := BSFServerComponent.create("BSF Server Simulation") alive;</pre>

Datei: AtsBCast_TestConfiguration_Functions.ttcn
(Fortsetzung)

72			68	
73	// change 1 (WK18): for content protection tests		69	// change 1 (WK18): for content protection tests
74	vc_BSM := BSMServerComponent.create("BSM Server Simulation") alive;		70	vc_BSM := BSMServerComponent.create("BSM Server Simulation") alive;
75		+-		
76	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)			
77	vc_Mikey := MikeyComponent.create("Mikey Component") alive;			
78	}	=	71	}
79			72	
80	// change 1 (WK18): for content protection tests		73	// change 1 (WK18): for content protection tests
81	/**		74	/**
82	* @desc Starts a DNS server simulation as a parallel component.		75	* @desc Starts a DNS server simulation as a parallel component.
83	*/		76	*/
84	function f_cf_DNS_up() runs on BCastMainComponent {		77	function f_cf_DNS_up() runs on BCastMainComponent {
85	map(vc_DNS: dns, system: dns);		78	map(vc_DNS: dns, system: dns);
86	}		79	}
87			80	
88	function f_cf_enable_DNS() runs on BCastMainComponent {		81	function f_cf_enable_DNS() runs on BCastMainComponent {
89	vc_DNS.start(DNS_ServerSimulation());		82	vc_DNS.start(DNS_ServerSimulation());
90	}		83	}
91			84	
92	// change 1 (WK18): for content protection tests		85	// change 1 (WK18): for content protection tests
93	/**		86	/**
94	* @desc Stops a DNS server simulation.		87	* @desc Stops a DNS server simulation.
95	*/		88	*/
96	function f_cf_DNS_down() runs on BCastMainComponent {		89	function f_cf_DNS_down() runs on BCastMainComponent {
97	unmap(vc_DNS: dns, system: dns);		90	unmap(vc_DNS: dns, system: dns);
98	vc_DNS.stop;		91	vc_DNS.stop;
99	}		92	}
100			93	
101	// change 1 (WK18): for content protection tests		94	// change 1 (WK18): for content protection tests
102	/**		95	/**
103	* @desc Starts a BSF server simulation as a parallel component.		96	* @desc Starts a BSF server simulation as a parallel component.
104	*/		97	*/
105	function f_cf_BSF_up() runs on BCastMainComponent {		98	function f_cf_BSF_up() runs on BCastMainComponent {
106	map(vc_BSF: bsf, system: bsf);		99	map(vc_BSF: bsf, system: bsf);
107	}		100	}
108			101	
109	// change 1 (WK18): for content protection tests		102	// change 1 (WK18): for content protection tests
110	/**		103	/**
111	* @desc Stops a BSF server simulation.		104	* @desc Stops a BSF server simulation.
112	*/		105	*/
113	function f_cf_BSF_down() runs on BCastMainComponent {		106	function f_cf_BSF_down() runs on BCastMainComponent {
114	unmap(vc_BSF: bsf, system: bsf);		107	unmap(vc_BSF: bsf, system: bsf);
115	vc_BSF.stop;		108	vc_BSF.stop;
116	}		109	}
117			110	
118	// change 1 (WK18): for content protection tests		111	// change 1 (WK18): for content protection tests
119	// hint: Bcast Subscription Management Server for MBMS User Service Registration		112	// hint: Bcast Subscription Management Server for MBMS User Service Registration
120	/**		113	/**
121	* @desc Starts a BSM server simulation as a parallel component.		114	* @desc Starts a BSM server simulation as a parallel component.
122	*/		115	*/
123	function f_cf_BSM_up() runs on BCastMainComponent {		116	function f_cf_BSM_up() runs on BCastMainComponent {
124	map(vc_BSM: bsm, system: bsm);		117	map(vc_BSM: bsm, system: bsm);
125	}		118	}
126			119	
127	// change 1 (WK18): for content protection tests		120	// change 1 (WK18): for content protection tests
128	/**		121	/**
129	* @desc Stops a BSM server simulation.		122	* @desc Stops a BSM server simulation.
130	*/		123	*/
131	function f_cf_BSM_down() runs on BCastMainComponent {		124	function f_cf_BSM_down() runs on BCastMainComponent {
132	unmap(vc_BSM: bsm, system: bsm);		125	unmap(vc_BSM: bsm, system: bsm);
133	vc_BSM.stop;		126	vc_BSM.stop;
134	}		127	}
135		+-		
136	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)			
137	function f_cf_Mikey_up() runs on BCastMainComponent {			
138	map(vc_Mikey: mk, system: mk);			
139	map(vc_Mikey: bmk, system: bmk);			
140	}			
141				

Datei: AtsBCast_TestConfiguration_Functions.ttcn
 (Fortsetzung)

142	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)			
143	function f_cf_Mikey_down() runs on BCastMainComponent {			
144	unmap(vc_Mikey: mk, system: mk);			
145	unmap(vc_Mikey: bmk, system: bmk);			
146	vc_Mikey.stop;			
147	}			
148		=	128	
149	/**		129	/**
150	*		130	*
151	* @desc		131	* @desc
152	* This function map the ports for the broadcast		132	* This function map the ports for the broadcast
153	* channel.		133	* channel.
154	*/		134	*/
155	function f_cf_Broadcast_up() runs on BCastMainComponent {		135	function f_cf_Broadcast_up() runs on BCastMainComponent {
156	map (vc_CTRL: ac, system: ac);		136	map (vc_CTRL: ac, system: ac);
157	map (vc_CTRL: dc, system: dc);		137	map (vc_CTRL: dc, system: dc);
158	}		138	}
159			139	
160	/**		140	/**
161	*		141	*
162	* @desc		142	* @desc
163	* This function unmap the ports for the broadcast		143	* This function unmap the ports for the broadcast
164	* channel.		144	* channel.
165	*/		145	*/
166	function f_cf_Broadcast_down() runs on BCastMainComponent {		146	function f_cf_Broadcast_down() runs on BCastMainComponent {
167	unmap (vc_CTRL: ac, system: ac);		147	unmap (vc_CTRL: ac, system: ac);
168	unmap (vc_CTRL: dc, system: dc);		148	unmap (vc_CTRL: dc, system: dc);
169	}		149	}
170			150	
171	/**		151	/**
172	*		152	*
173	* @desc This function map the ports for the file streaming		153	* @desc This function map the ports for the file streaming
174	*/		154	*/
175	function f_cf_Data_up() runs on BCastMainComponent {		155	function f_cf_Data_up() runs on BCastMainComponent {
176	map (vc_DATA: ssc, system: ssc);		156	map (vc_DATA: ssc, system: ssc);
177	map (vc_DATA: fsc, system: fsc);		157	map (vc_DATA: fsc, system: fsc);
178	}		158	}
179			159	
180	// change 3 (WK34): Service Protection: secure streaming service	+ -		
181	/**	=	160	/**
182	*		161	*
183	* @desc This function map the ports for the secure file streaming	< >		
184	*/			
185	function f_cf_Secure_Data_up() runs on BCastMainComponent {			
186	map (vc_DATA: sc, system: sc);			
187	}			
188				
189	/**			
190	*			
191	* @desc This function unmap/disconnect the ports for the file streaming		162	* @desc This function nmap/disconnect the ports for the file streaming
192	*/	=	163	*/
193	function f_cf_Data_down() runs on BCastMainComponent {		164	function f_cf_Data_down() runs on BCastMainComponent {
194	unmap (vc_DATA: ssc, system: ssc);		165	unmap (vc_DATA: ssc, system: ssc);
195	unmap (vc_DATA: fsc, system: fsc);		166	unmap (vc_DATA: fsc, system: fsc);
196	}	+ -		
197				
198	// change 3 (WK34): Service Protection: secure streaming service			
199	/**			
200	*			
201	* @desc This function unmap/disconnect the ports for the secure file streaming			
202	*/			
203	function f_cf_Secure_Data_down() runs on BCastMainComponent {			
204	unmap (vc_DATA: sc, system: sc);			
205	}	=	167	}
206			168	
207	/**		169	/**
208	*		170	*
209	* @desc This function map the ports for mms.		171	* @desc This function map the ports for mms.
210	*/		172	*/
211	function f_cf_MMS_up() runs on BCastMainComponent {		173	function f_cf_MMS_up() runs on BCastMainComponent {

Datei: AtsBCast_TestConfiguration_Functions.ttcn
(Fortsetzung)

212	map (vc_CTRL: mms, system: mms);		174	map (vc_CTRL: mms, system: mms);
213	}		175	}
214			176	
215	/**		177	/**
216	*		178	*
217	* @desc This function unmap the ports for mms.		179	* @desc This function unmap the ports for mms.
218	*/		180	*/
219	function f_cf_MMS_down() runs on BCastMainComponent {		181	function f_cf_MMS_down() runs on BCastMainComponent {
220	unmap (vc_CTRL: mms, system: mms);		182	unmap (vc_CTRL: mms, system: mms);
221	}		183	}
222			184	
223	/**		185	/**
224	*		186	*
225	* @desc This function map the ports for sms.	<>	187	* @desc This function map the ports for sms.
226	*/	=	188	*/
227	function f_cf_SMS_up() runs on BCastMainComponent {		189	function f_cf_SMS_up() runs on BCastMainComponent {
228	map (vc_CTRL: sms, system: sms);		190	map (vc_CTRL: sms, system: sms);
229	}		191	}
230			192	
231			193	
232	/**		194	/**
233	*		195	*
234	* @desc This function unmap the ports for sms.		196	* @desc This function unmap the ports for sms.
235	*/		197	*/
236	function f_cf_SMS_down() runs on BCastMainComponent {		198	function f_cf_SMS_down() runs on BCastMainComponent {
237	unmap (vc_CTRL: sms, system: sms);		199	unmap (vc_CTRL: sms, system: sms);
238	}		200	}
239			201	
240	/**		202	/**
241	*		203	*
242	* @desc		204	* @desc
243	* This function map the ports for the interaction		205	* This function map the ports for the interaction
244	* channel		206	* channel
245	*/		207	*/
246	function f_cf_Interaction_up() runs on BCastMainComponent {		208	function f_cf_Interaction_up() runs on BCastMainComponent {
247	map (vc_CTRL: ic, system: ic);		209	map (vc_CTRL: ic, system: ic);
248	}		210	}
249			211	
250	/**		212	/**
251	*		213	*
252	* @desc		214	* @desc
253	* This function unmap the ports for the interaction		215	* This function unmap the ports for the interaction
254	* channel		216	* channel
255	*/		217	*/
256	function f_cf_Interaction_down() runs on BCastMainComponent {		218	function f_cf_Interaction_down() runs on BCastMainComponent {
257	unmap (vc_CTRL: ic, system: ic);		219	unmap (vc_CTRL: ic, system: ic);
258	}		220	}
259			221	
260	/**		222	/**
261	*		223	*
262	* @desc This function mapps/connect the ports for the upper test comunication.		224	* @desc This function mapps/connect the ports for the upper test comunication.
263	*/		225	*/
264	function f_cf_UpperTester_up() runs on BCastMainComponent {		226	function f_cf_UpperTester_up() runs on BCastMainComponent {
265	map (vc_UTC: utp, system: utp);		227	map (vc_UTC: utp, system: utp);
266	}		228	}
267			229	
268	/**		230	/**
269	*		231	*
270	* @desc This function unmap the ports for the upper test comunication.		232	* @desc This function unmap the ports for the upper test comunication.
271	*/		233	*/
272	function f_cf_UpperTester_down() runs on BCastMainComponent {		234	function f_cf_UpperTester_down() runs on BCastMainComponent {
273	unmap (vc_UTC: utp, system: utp);		235	unmap (vc_UTC: utp, system: utp);
274	}		236	}
275			237	
276			238	
277	/**		239	/**
278	*		240	*
279	* @desc This function set up the preambles for BCast.		241	* @desc This function set up the preambles for BCast.
280	*/		242	*/
281	function f_startCommonUtPreamble() runs on BCastMainComponent {		243	function f_startCommonUtPreamble() runs on BCastMainComponent {

Datei: AtsBCast_TestConfiguration_Functions.ttcn
(Fortsetzung)

282	f_main_ut_PowerOn();	244	f_main_ut_PowerOn();
283	f_main_ut_ClearServiceGuide();	245	f_main_ut_ClearServiceGuide();
284	}	246	}
285		247	
286	// change 1 (WK18): for content protection tests	248	// change 1 (WK18): for content protection tests
287	/**	249	/**
288	* @desc Stops all compontens. Sufficient and feasible for all tests with	250	* @desc Stops all compontens. Sufficient and feasible for all tests with
289	* parallel components (e.g. Protection Testcases).	251	* parallel components (e.g. Protection Testcases).
290	*/	252	*/
291	function f_terminateComponents() {	253	function f_terminateComponents() {
292	all component.stop	254	all component.stop
293	}	255	}
294		256	
295	}// end module AtsBCast_TestConfiguration	257	}// end module AtsBCast_TestConfiguration

Datei: AtsBCast_TestControl.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies a fairly static test execution control. Via 9 * module parameters for test area or specific test case identifiers 10 * execution can be controlled. The BCAST test suite can also be 11 * compiled without this module. In that case test case execution must 12 * be implemented via the TCI-TM interface. 13 */ 14 module AtsBCast_TestControl { 15 import from AtsBCast_ModuleParameters all; 16 import from AtsBCast_ContentProtectionTests all; 17 import from AtsBCast_FileAndStreamDistributionTests all; 18 import from AtsBCast_ServiceGuideTests all; 19 import from AtsBCast_ServiceInteractionTests all; 20 import from AtsBCast_ServiceProvisioningTests all;</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies a fairly static test execution control. Via 9 * module parameters for test area or specific test case identifiers 10 * execution can be controlled. The BCAST test suite can also be 11 * compiled without this module. In that case test case execution must 12 * be implemented via the TCI-TM interface. 13 */ 14 module AtsBCast_TestControl { 15 import from AtsBCast_ModuleParameters all; 16 import from AtsBCast_ContentProtectionTests all; 17 import from AtsBCast_FileAndStreamDistributionTests all; 18 import from AtsBCast_ServiceGuideTests all; 19 import from AtsBCast_ServiceInteractionTests all; 20 import from AtsBCast_ServiceProvisioningTests all;</pre>
<pre>21 import from AtsBCast_ServiceProtectionTests all;</pre>	+ -	
<pre>22 23 // test case selection switches 24 control { 25 log ("Control part started..."); 26 if (PX_ALL_TCS) { 27 log ("Running all test cases"); 28 } 29 if (PX_ALL_TCS or PX_ALL_SP_TCS) { 30 log ("Running all service provisoring test cases"); 31 execute (TC_BCAST_PROV_CONF_101()); 32 execute (TC_BCAST_PROV_CONF_102()); 33 } 34 if (PX_ALL_TCS or PX_ALL_SG_TCS) { 35 log ("Running all service guide test cases"); 36 execute (TC_BCAST_ESG_CONF_101()); 37 execute (TC_BCAST_ESG_CONF_102()); 38 execute (TC_BCAST_ESG_CONF_103()); 39 execute (TC_BCAST_ESG_CONF_104()); 40 execute (TC_BCAST_ESG_CONF_105()); 41 execute (TC_BCAST_ESG_CONF_106()); 42 execute (TC_BCAST_ESG_CONF_107()); 43 execute (TC_BCAST_ESG_CONF_108()); 44 } 45 if (PX_ALL_TCS or PX_ALL_FD_TCS) { 46 log ("Running all file distribution test cases"); 47 execute (TC_BCAST_DIST_CONF_102()); 48 } 49 if (PX_ALL_TCS or PX_ALL_SI_TCS) { 50 log ("Running all service interaction test cases"); 51 execute (TC_BCAST_INTER_CONF_101()); 52 execute (TC_BCAST_INTER_CONF_102()); 53 execute (TC_BCAST_INTER_CONF_103()); 54 } 55 if (PX_ALL_TCS or PX_ALL_CP_TCS) { 56 log ("Running all content protection test cases"); 57 execute (TC_BCAST_CONTPROT_CONF_101()); 58 execute (TC_BCAST_CONTPROT_CONF_102()); 59 execute (TC_BCAST_CONTPROT_CONF_103()); 60 } 61 62 if (PX_ALL_TCS or PX_ALL_SProt_TCS) { 63 log ("Running all Service protection test cases"); 64 execute (TC_BCAST_SERVPROT_101a()); 65 execute (TC_BCAST_SERVPROT_101b()); 66 execute (TC_BCAST_SERVPROT_101c()); 67 execute (TC_BCAST_SERVPROT_101d()); 68 execute (TC_BCAST_SERVPROT_101e()); 69 execute (TC_BCAST_SERVPROT_101f()); 70 } 71 }</pre>	=	<pre>22 // test case selection switches 23 control { 24 log ("Control part started..."); 25 if (PX_ALL_TCS) { 26 log ("Running all test cases"); 27 } 28 if (PX_ALL_TCS or PX_ALL_SP_TCS) { 29 log ("Running all service provisoring test cases"); 30 execute (TC_BCAST_PROV_CONF_101()); 31 execute (TC_BCAST_PROV_CONF_102()); 32 } 33 if (PX_ALL_TCS or PX_ALL_SG_TCS) { 34 log ("Running all service guide test cases"); 35 execute (TC_BCAST_ESG_CONF_101()); 36 execute (TC_BCAST_ESG_CONF_102()); 37 execute (TC_BCAST_ESG_CONF_103()); 38 execute (TC_BCAST_ESG_CONF_104()); 39 execute (TC_BCAST_ESG_CONF_105()); 40 execute (TC_BCAST_ESG_CONF_106()); 41 execute (TC_BCAST_ESG_CONF_107()); 42 execute (TC_BCAST_ESG_CONF_108()); 43 } 44 if (PX_ALL_TCS or PX_ALL_FD_TCS) { 45 log ("Running all file distribution test cases"); 46 execute (TC_BCAST_DIST_CONF_102()); 47 } 48 if (PX_ALL_TCS or PX_ALL_SI_TCS) { 49 log ("Running all service interaction test cases"); 50 execute (TC_BCAST_INTER_CONF_101()); 51 execute (TC_BCAST_INTER_CONF_102()); 52 execute (TC_BCAST_INTER_CONF_103()); 53 } 54 if (PX_ALL_TCS or PX_ALL_CP_TCS) { 55 log ("Running all content protection test cases"); 56 execute (TC_BCAST_CONTPROT_CONF_101()); 57 execute (TC_BCAST_CONTPROT_CONF_102()); 58 execute (TC_BCAST_CONTPROT_CONF_103()); 59 } 60 61 }</pre>
<pre>61 if (PX_ALL_TCS or PX_ALL_SProt_TCS) { 62 log ("Running all Service protection test cases"); 63 execute (TC_BCAST_SERVPROT_101a()); 64 execute (TC_BCAST_SERVPROT_101b()); 65 execute (TC_BCAST_SERVPROT_101c()); 66 execute (TC_BCAST_SERVPROT_101d()); 67 execute (TC_BCAST_SERVPROT_101e()); 68 execute (TC_BCAST_SERVPROT_101f()); 69 }</pre>	+ -	
<pre>70 } 71 } // end module AtsBCast_TestControl</pre>	=	<pre>60 } 61 } // end module AtsBCast_TestControl</pre>

Datei: AtsBCast_TestSystem.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies component types for test components used in the 9 * BCAST test suite. The system component types is defined in the BCAST 10 * platform interface module 11 * @see 12 * LibBCast_Interface.BCastPlatformComponent 13 */ 14 module AtsBCast_TestSystem { 15 import from LibBCast_Interface all; 16 import from LibCommon_BasicTypesAndValues { 17 type UInt32; 18 type UInt; 19 } 20 // change 1 (WK18): for content protection tests 21 import from LibCommon_DNS { 22 group DNS_Configuration_and_Components 23 } 24 // change 1 (WK18): for content protection tests 25 import from LibCommon_GBA_BSF { 26 group BSF_Configuration_and_Components; 27 } 28 // change 1 (WK18): for content protection tests 29 import from LibCommon_BSM { 30 group BSM_Configuration_and_Components; 31 } 32 33 // change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...) 34 import from LibCommon_Mikey { 35 group Mikey_Configuration_and_Components; 36 } 37 38 group atsComponentTypes { 39 /** 40 * 41 * @desc 42 * This component type is used for the Main Test Component (MTC) 43 * which coordinates the creation and execution of BCAST control, 44 * BCAST data and Upper Tester components. 45 */ 46 type component BCastMainComponent { 47 var BCastNetworkControlComponent vc_CTRL; 48 var BCastNetworkDataComponent vc_DATA; 49 var UpperTesterComponent vc_UTC; 50 51 // change 1 (WK18): for content protection tests 52 var DNSServerComponent vc_DNS; 53 var BSFServerComponent vc_BSF; 54 var BSMServerComponent vc_BSM; 55 56 // change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...) 57 var MikeyComponent vc_Mikey; 58 59 var UInt32 vc_FluteUpdateID := 15; // CR (WK 34/2008): Flute Update Sign 60 var UInt32 vc_BCASTTransportID := 1; // change 05 (WK 6): renaming to vc_F 61 var UInt v_startTime := 0; // change 01 (WK 6): global time defi 62 var UInt v_endTime := 0; // change 01 (WK 6): global time defi 63 } 64 } 65 66 group testSystemComponents { 67 /** 68 * @desc 69 * This component type defines the interface for BCAST tests to the 70 * SUT Adapter. It shall be used as the system component in any BCAST 71 * test case statement</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies component types for test components used in the 9 * BCAST test suite. The system component types is defined in the BCAST 10 * platform interface module 11 * @see 12 * LibBCast_Interface.BCastPlatformComponent 13 */ 14 module AtsBCast_TestSystem { 15 import from LibBCast_Interface all; 16 import from LibCommon_BasicTypesAndValues { 17 type UInt32; 18 type UInt; 19 } 20 // change 1 (WK18): for content protection tests 21 import from LibCommon_DNS { 22 group DNS_Configuration_and_Components 23 } 24 // change 1 (WK18): for content protection tests 25 import from LibCommon_GBA_BSF { 26 group BSF_Configuration_and_Components; 27 } 28 // change 1 (WK18): for content protection tests 29 import from LibCommon_BSM { 30 group BSM_Configuration_and_Components; 31 } 32 33 // change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...) 34 import from LibCommon_Mikey { 35 group Mikey_Configuration_and_Components; 36 } 37 38 group atsComponentTypes { 39 /** 40 * 41 * @desc 42 * This component type is used for the Main Test Component (MTC) 43 * which coordinates the creation and execution of BCAST control, 44 * BCAST data and Upper Tester components. 45 */ 46 type component BCastMainComponent { 47 var BCastNetworkControlComponent vc_CTRL; 48 var BCastNetworkDataComponent vc_DATA; 49 var UpperTesterComponent vc_UTC; 50 51 // change 1 (WK18): for content protection tests 52 var DNSServerComponent vc_DNS; 53 var BSFServerComponent vc_BSF; 54 var BSMServerComponent vc_BSM; 55 56 // change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...) 57 var MikeyComponent vc_Mikey; 58 59 var UInt32 vc_FluteUpdateID := 15; // CR (WK 34/2008): Flute Update Sign 60 var UInt32 vc_BCASTTransportID := 1; // change 05 (WK 6): renaming to vc_F 61 var UInt v_startTime := 0; // change 01 (WK 6): global time defi 62 var UInt v_endTime := 0; // change 01 (WK 6): global time defi 63 } 64 } 65 66 group testSystemComponents { 67 /** 68 * @desc 69 * This component type defines the interface for BCAST tests to the 70 * SUT Adapter. It shall be used as the system component in any BCAST 71 * test case statement</pre>	+ -	
	=			
	+ -			
	=			
	+ -			
	=			

Datei: AtsBCast_TestSystem.ttcn (Fortsetzung)

72	*/		65	*/
73	type component BCastPlatformComponent {		66	type component BCastPlatformComponent {
74	port UtpPort utp;		67	port UtpPort utp;
75	port BroadcastPort ac;		68	port BroadcastPort ac;
76	port BroadcastPort dc;		69	port BroadcastPort dc;
77	port InteractionPort ic;		70	port InteractionPort ic;
78	port StreamServerCtrlPort ssc;		71	port StreamServerCtrlPort ssc;
79	// change 3 (WK34): Service Protection: secure streaming service	+-		
80	port SecureStreamServerCtrlPort sc;			
81	port SmsPort sms;	=	72	port SmsPort sms;
82	port MmsPort mms;		73	port MmsPort mms;
83	port FileServerControlPort fsc;		74	port FileServerControlPort fsc;
84	// change 1 (WK18): for content protection tests		75	// change 1 (WK18): for content protection tests
85	port BSFPort bsf;		76	port BSFPort bsf;
86	port DNSPort dns;		77	port DNSPort dns;
87	port BSMPort bsm;		78	port BSMPort bsm;
88	// change 1 (WK34): Service Protection: Mikey (definition, sending, receiving ...)	+-		
89	port MikeyPort mk;			
90	port MikeyPort mmk;			
91	}	=	79	}
92			80	
93			81	
94	} // end module AtsBCast_TestSystem		82	} // end module AtsBCast_TestSystem

Datei: LibBCast_BCastNetworkData_Functions.ttcn

<pre> 1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies some function which can be used by the BCastNetworkDataComponent component. 9 */ 10 module LibBCast_BCastNetworkData_Functions { 11 12 import from LibCommon_Time { 13 modulepar PX_TWAIT 14 } 15 // change 12 (WK18): (additional to 11) in order to broadcast data files over IP and port for FLUTE 16 import from AtsBCast_ModuleParameters all; 17 import from LibBCast_Common_Templates { 18 template m_def_StartStreamingRequest; 19 template m_def_StopStreamingRequest; 20 template m_def_StartFileTransferRequest; 21 template m_def_StopFileTransferRequest; </pre>	=	<pre> 1 /** 2 * 3 * @author 4 * ETSI CTI BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies some function which can be used by the BCastNetworkDataComponent component. 9 */ 10 module LibBCast_BCastNetworkData_Functions { 11 12 import from LibCommon_Time { 13 modulepar PX_TWAIT 14 } 15 // change 12 (WK18): (additional to 11) in order to broadcast data files over IP and port for FLUTE 16 import from AtsBCast_ModuleParameters all; 17 import from LibBCast_Common_Templates { 18 template m_def_StartStreamingRequest; 19 template m_def_StopStreamingRequest; 20 template m_def_StartFileTransferRequest; 21 template m_def_StopFileTransferRequest; </pre>
<pre> 22 // change 3 (WK34): Service Protection: secure streaming service 23 template m_def_UpdateKeyRequest; 24 template m_def_ActivateKeyRequest; 25 template m_def_StartSecureStreamingRequest; 26 template m_def_StopSecureStreamingRequest; </pre>	+-	
<pre> 27 } 28 import from LibBCast_ServicePrimitives_TypesAndValues all; 29 // change 13 (WK18): (additional to 11) dynamic setting of content location and expiry time 30 import from LibBCast_ServiceGuide_Templates { 31 template m_def_FileType; 32 template m_def_FDT_Instance; 33 } 34 import from LibCommon_BasicTypesAndValues { 35 type UInt, UInt32 36 } 37 import from LibBCast_Interface all; </pre>	=	<pre> 22 } 23 import from LibBCast_ServicePrimitives_TypesAndValues all; 24 // change 13 (WK18): (additional to 11) dynamic setting of content location and expiry time 25 import from LibBCast_ServiceGuide_Templates { 26 template m_def_FileType; 27 template m_def_FDT_Instance; 28 } 29 import from LibCommon_BasicTypesAndValues { 30 type UInt, UInt32 31 } 32 import from LibBCast_Interface all; </pre>
<pre> 38 // change 3 (WK34): Service Protection: secure streaming service 39 import from LibCommon_DataStrings all; </pre>	+-	
<pre> 40 41 group subFunctions { 42 group behaviorFunctions { 43 44 // CR (WK 34/2008): Flute Update Signalling 45 /** 46 * 47 * @desc This function sends a File Transfer Request 48 * @param p_TransferId The transfer id 49 * @param p_FluteUpdateID The Flute Instance ID of the Flute Session. 50 * @param p_Sdp the used SDP 51 * @param p_FileList The list of files which should delivered 52 * @verdict 53 * pass In case that the file transfer request was successful 54 * @verdict 55 * fail A wrong message was received. 56 * @verdict 57 * inconc A guard timer expires. 58 */ 59 function f_data_startFileTransfer(UInt p_TransferId, 60 UInt32 p_FluteUpdateID, 61 charstring p_Sdp, 62 TransferFileList p_FileList) runs on BCastNetworkDataComponent { 63 var FDT_InstanceType v_fdt := f_createFDT4FileTransfer(p_FileList); // charstring 64 65 var StartFileTransferRequest v_Request := valueof(m_def_StartFileTransferRequest { 66 p_TransferId, 67 PX_DATA_FLUTE_SESSION_IP, // change 12 (WK18): (additional to 11) 68 PX_DATA_FLUTE_SESSION_PORT, // change 12 (WK18): (additional to 11) 69 p_FluteUpdateID, // CR (WK 34/2008): Flute Update Signalling 70 v_fdt, 71 p_FileList, </pre>	=	<pre> 33 34 group subFunctions { 35 group behaviorFunctions { 36 37 // CR (WK 34/2008): Flute Update Signalling 38 /** 39 * 40 * @desc This function sends a File Transfer Request 41 * @param p_TransferId The transfer id 42 * @param p_FluteUpdateID The Flute Instance ID of the Flute Session. 43 * @param p_Sdp the used SDP 44 * @param p_FileList The list of files which should delivered 45 * @verdict 46 * pass In case that the file transfer request was successful 47 * @verdict 48 * fail A wrong message was received. 49 * @verdict 50 * inconc A guard timer expires. 51 */ 52 function f_data_startFileTransfer(UInt p_TransferId, 53 UInt32 p_FluteUpdateID, 54 charstring p_Sdp, 55 TransferFileList p_FileList) runs on BCastNetworkDataComponent { 56 var FDT_InstanceType v_fdt := f_createFDT4FileTransfer(p_FileList); // charstring 57 58 var StartFileTransferRequest v_Request := valueof(m_def_StartFileTransferRequest { 59 p_TransferId, 60 PX_DATA_FLUTE_SESSION_IP, // change 12 (WK18): (additional to 11) 61 PX_DATA_FLUTE_SESSION_PORT, // change 12 (WK18): (additional to 11) 62 p_FluteUpdateID, // CR (WK 34/2008): Flute Update Signalling 63 v_fdt, 64 p_FileList, </pre>

Datei: LibBCast_BCastNetworkData_Functions.ttcn (Fortsetzung)

72	p_Sdp	65	p_Sdp
73));	66));
74		67	
75	fsc.send(v_Request);	68	fsc.send(v_Request);
76		69	
77	t_wait.start (PX_TWAIT);	70	t_wait.start (PX_TWAIT);
78		71	
79	alt {	72	alt {
80	[] fsc.receive (StartFileTransferResponse: { 0, * }) {	73	[] fsc.receive (StartFileTransferResponse: { 0, * }) {
81	t_wait.stop;	74	t_wait.stop;
82	setverdict (pass);	75	setverdict (pass);
83	}	76	}
84	[] fsc.receive {	77	[] fsc.receive {
85	t_wait.stop;	78	t_wait.stop;
86	setverdict (fail);	79	setverdict (fail);
87	}	80	}
88	[] t_wait.timeout {	81	[] t_wait.timeout {
89	setverdict (inconc);	82	setverdict (inconc);
90	}	83	}
91	}	84	}
92	}	85	}
93		86	
94	/**	87	/**
95	*	88	*
96	* @desc This function sends a stop file transfer request to the file server	89	* @desc This function sends a stop file transfer request to the file server
97	* @param p_StreamId The id of the file transfer which should stopped	90	* @param p_StreamId The id of the file transfer which should stopped
98	* @verdict	91	* @verdict
99	* pass on receipt of a successful StopFileTransferResponse	92	* pass on receipt of a successful StopFileTransferResponse
100	* @verdict	93	* @verdict
101	* inconc on receipt of a wrong message or in case the guard	94	* inconc on receipt of a wrong message or in case the guard
102	* timer expires.	95	* timer expires.
103	*/	96	*/
104	function f_data_stopFileTransfer(UInt p_StreamId)	97	function f_data_stopFileTransfer(UInt p_StreamId)
105	runs on BCastNetworkDataComponent {	98	runs on BCastNetworkDataComponent {
106		99	
107	var StopFileTransferRequest v_Request :=	100	var StopFileTransferRequest v_Request :=
108	valueOf(m_def_StopFileTransferRequest(p_StreamId));	101	valueOf(m_def_StopFileTransferRequest(p_StreamId));
109		102	
110	fsc.send (StopFileTransferRequest:v_Request);	103	fsc.send (StopFileTransferRequest:v_Request);
111	t_wait.start (PX_TWAIT);	104	t_wait.start (PX_TWAIT);
112		105	
113	alt {	106	alt {
114	[] fsc.receive (StopFileTransferResponse: { 0, * }) {	107	[] fsc.receive (StopFileTransferResponse: { 0, * }) {
115	t_wait.stop;	108	t_wait.stop;
116	setverdict (pass);	109	setverdict (pass);
117	}	110	}
118	[] fsc.receive {	111	[] fsc.receive {
119	t_wait.stop;	112	t_wait.stop;
120	setverdict (fail);	113	setverdict (fail);
121	}	114	}
122	[] t_wait.timeout {	115	[] t_wait.timeout {
123	setverdict (inconc);	116	setverdict (inconc);
124	}	117	}
125	}	118	}
126	}	119	}
127		120	
128	// change 13 (WK18): (additional to 11) dynamic setting of content location and exp	121	// change 13 (WK18): (additional to 11) dynamic setting of content location and exp
129	/**	122	/**
130	*	123	*
131	* @desc	124	* @desc
132	* Creates a File Dilivery Table Instance (FDT) for a Service	125	* Creates a File Dilivery Table Instance (FDT) for a Service
133	* Guide Delivery Descriptor (SGDD).	126	* Guide Delivery Descriptor (SGDD).
134	* @param	127	* @param
135	* p_Content_Location the Service Guide Delivery Descriptor id.	128	* p_Content_Location the Service Guide Delivery Descriptor id.
136	* @param	129	* @param
137	* p_ContentEncoding the encoding of the SGDD	130	* p_ContentEncoding the encoding of the SGDD
138	* @param	131	* @param
139	* p_Expires the expire time how long this FDT is vaild.	132	* p_Expires the expire time how long this FDT is vaild.
140	* @return	133	* @return
141	* the File Delivery Table Instance (FDT)	134	* the File Delivery Table Instance (FDT)
142	*/	135	*/

Datei: LibBCast_BCastNetworkData_Functions.ttcn (Fortsetzung)

143	function f_createFDT4FileTransfer (TransferFileList p_TransferFileList)	136	function f_createFDT4FileTransfer (TransferFileList p_TransferFileList)
144	runs on BCastNetworkDataComponent return FDT_InstanceType {	137	runs on BCastNetworkDataComponent return FDT_InstanceType {
145	var LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;	138	var LibBCast_ServicePrimitives_TypesAndValues.FileList v_FileList;
146	var LibBCast_ServicePrimitives_TypesAndValues.FileType v_File;	139	var LibBCast_ServicePrimitives_TypesAndValues.FileType v_File;
147		140	
148	for (var integer i := 0; i < sizeof (p_TransferFileList); i := i + 1) {	141	for (var integer i := 0; i < sizeof (p_TransferFileList); i := i + 1) {
149	v_File := valueof(m_def_FileType(142	v_File := valueof(m_def_FileType(
150	p_TransferFileList[i].Content_Location,	143	p_TransferFileList[i].Content_Location,
151	i + 1,	144	i + 1,
152	p_TransferFileList[i].Content_Type,	145	p_TransferFileList[i].Content_Type,
153	p_TransferFileList[i].Content_Encoding));	146	p_TransferFileList[i].Content_Encoding));
154	v_FileList[i] := v_File;	147	v_FileList[i] := v_File;
155	}	148	}
156		149	
157	var FDT_InstanceType v_FDT :=	150	var FDT_InstanceType v_FDT :=
158	valueof(m_def_FDT_Instance(v_FileList, int2str(v_endTime), omit, 0))	151	valueof(m_def_FDT_Instance(v_FileList, int2str(v_endTime), omit, 0))
159		152	
160	return v_FDT;	153	return v_FDT;
161	}	154	}
162		155	
163	/**	156	/**
164	*	157	*
165	* @desc This function sends a start streaming request to the streaming server.	158	* @desc This function sends a start streaming request to the streaming server.
166	* @param p_StreamId The stream id	159	* @param p_StreamId The stream id
167	* @param p_FileName The file which should be streamed	160	* @param p_FileName The file which should be streamed
168	* @param p_SDP The used SDP	161	* @param p_SDP The used SDP
169	* @param p_DstIP The destination IP address// change 14 (WK18): dynamic setting of	162	* @param p_DstIP The destination IP address// change 14 (WK18): dynamic setting of
170	* @verdict	163	* @verdict
171	* pass In case that the streaming request was successful	164	* pass In case that the streaming request was successful
172	* @verdict	165	* @verdict
173	* fail A wrong message was received.	166	* fail A wrong message was received.
174	* @verdict	167	* @verdict
175	* inconc A guard timer expires.	168	* inconc A guard timer expires.
176	*/	169	*/
177	function f_data_startStreaming(170	function f_data_startStreaming(
178	UInt p_StreamId,	171	UInt p_StreamId,
179	charstring p_FileName,	172	charstring p_FileName,
180	charstring p_SDP,	173	charstring p_SDP,
181	charstring p_DstIP)	174	charstring p_DstIP)
182	runs on BCastNetworkDataComponent {	175	runs on BCastNetworkDataComponent {
183		176	
184	var StartStreamingRequest v_Request :=	177	var StartStreamingRequest v_Request :=
185	valueof(m_def_StartStreamingRequest(p_StreamId, p_FileName, p_SDP,	178	valueof(m_def_StartStreamingRequest(p_StreamId, p_FileName, p_SDP,
186		179	
187	ssc.send (v_Request);	180	ssc.send (v_Request);
188	t_wait.start (PX_TWAIT);	181	t_wait.start (PX_TWAIT);
189		182	
190	alt {	183	alt {
191	[] ssc.receive (StartStreamingResponse: {0, *}) {	184	[] ssc.receive (StartStreamingResponse: {0, *}) {
192	t_wait.stop;	185	t_wait.stop;
193	setverdict (pass);	186	setverdict (pass);
194	}	187	}
195	[] ssc.receive {	188	[] ssc.receive {
196	t_wait.stop;	189	t_wait.stop;
197	setverdict (fail);	190	setverdict (fail);
198	}	191	}
199	[] t_wait.timeout {	192	[] t_wait.timeout {
200	setverdict (inconc);	193	setverdict (inconc);
201	}	194	}
202	}	195	}
203	}	196	}
204		197	
205	/**	198	/**
206	*	199	*
207	* @desc This function sends a stop streaming request to the streaming server	200	* @desc This function sends a stop streaming request to the streaming server
208	* @param p_StreamId The id of the stream which should stopped	201	* @param p_StreamId The id of the stream which should stopped
209	* @verdict	202	* @verdict
210	* pass on receipt of a successful StopStreamingResponse	203	* pass on receipt of a successful StopStreamingResponse
211	* @verdict	204	* @verdict
212	* inconc on receipt of a wrong message or in case the guard	205	* inconc on receipt of a wrong message or in case the guard
213	* timer expires.	206	* timer expires.

Datei: LibBCast_BCastNetworkData_Functions.ttcn (Fortsetzung)

214	*/	207	*/
215	function f_data_stopStreaming(UInt p_StreamId)	208	function f_data_stopStreaming(UInt p_StreamId)
216	runs on BCastNetworkDataComponent {	209	runs on BCastNetworkDataComponent {
217		210	
218	var StopStreamingRequest v_Request :=	211	var StopStreamingRequest v_Request :=
219	valueof(m_def_StopStreamingRequest(p_StreamId));	212	valueof(m_def_StopStreamingRequest(p_StreamId));
220		213	
221	ssc.send (v_Request);	214	ssc.send (v_Request);
222	t_wait.start (PX_TWAIT);	215	t_wait.start (PX_TWAIT);
223		216	
224	alt {	217	alt {
225	[] ssc.receive (StopStreamingResponse: { 0, * }) {	218	[] ssc.receive (StopStreamingResponse: { 0, * }) {
226	t_wait.stop;	219	t_wait.stop;
227	setverdict (pass);	220	setverdict (pass);
228	}	221	}
229	[] ssc.receive {	222	[] ssc.receive {
230	t_wait.stop;	223	t_wait.stop;
231	setverdict (fail);	224	setverdict (fail);
232	}	225	}
233	[] t_wait.timeout {	226	[] t_wait.timeout {
234	setverdict (inconc);	227	setverdict (inconc);
235	}	228	}
236	}	229	}
237	}	230	}
238			
239	// change 3 (WK34): Service Protection: secure streaming service		
240	/**		
241	*		
242	* @desc This function sends a start secure streaming request to the secure streami		
243	* @param p_DstIP The destination IP address		
244	* @param p_ssrc The secure source ID		
245	* @verdict		
246	* pass In case that the streaming request was successful		
247	* @verdict		
248	* fail A wrong message was received.		
249	* @verdict		
250	* inconc A guard timer expires.		
251	*/		
252	function f_data_secure_startStreaming(charstring p_DstIP, Oct4 p_ssrc) runs on BCas		
253	var StartSecureStreamingRequest v_Request :=		
254	valueof(m_def_StartSecureStreamingRequest(p_DstIP, p_ssrc));		
255			
256	sc.send(v_Request);		
257	t_wait.start(PX_TWAIT);		
258			
259	alt {		
260	[] sc.receive(StartSecureStreamingResponse: {0, *}) {		
261	t_wait.stop;		
262	log("Verdict Info: The secure streaming service sends a Sta		
263	setverdict(pass);		
264	}		
265	[] sc.receive(StartSecureStreamingResponse: {*, *}) {		
266	t_wait.stop;		
267	log("Verdict Info: The secure streaming service does not se		
268	setverdict(fail);		
269	}		
270	[] t_wait.timeout {		
271	log("Verdict Info: The secure streaming service does not se		
272	setverdict(inconc);		
273	}		
274	}		
275	}		
276			
277	// change 3 (WK34): Service Protection: secure streaming service		
278	/**		
279	*		
280	* @desc This function sends a stop secure streaming request to the secure streami		
281	* @verdict		
282	* pass on receipt of a successful StopStreamingResponse		
283	* @verdict		
284	* fail on receipt of a wrong message		

Datei: LibBCast_BCastNetworkData_Functions.ttcn (Fortsetzung)

285	<pre> * @verdict * inconc in case the guard timer expires */ function f_data_secure_stopStreaming() runs on BCastNetworkDataComponent { var StopSecureStreamingRequest v_Request := valueof(m_def_StopSecureStreamingRequest);</pre>	
286		
287		
288		
289		
290		
291	<pre> sc.send(v_Request);</pre>	
292	<pre> t_wait.start(PX_TWAIT);</pre>	
293		
294	<pre> alt {</pre>	
295	<pre> [] sc.receive(StopSecureStreamingResponse: {0, *}) {</pre>	
296	<pre> t_wait.stop;</pre>	
297	<pre> log("Verdict Info: The secure streaming service sends a StopSecureStreamingResponse");</pre>	
298	<pre> setverdict(pass);</pre>	
299	<pre> }</pre>	
300	<pre> [] sc.receive(StopSecureStreamingResponse: {*, *}) {</pre>	
301	<pre> t_wait.stop;</pre>	
302	<pre> log("Verdict Info: The secure streaming service does not send a StopSecureStreamingResponse");</pre>	
303	<pre> setverdict(fail);</pre>	
304	<pre> }</pre>	
305	<pre> [] t_wait.timeout {</pre>	
306	<pre> log("Verdict Info: The secure streaming service does not send a StopSecureStreamingResponse within the guard time");</pre>	
307	<pre> setverdict(inconc);</pre>	
308	<pre> }</pre>	
309	<pre> }</pre>	
310	<pre> }</pre>	
311		
312	<pre>// change 3 (WK34): Service Protection: secure streaming service</pre>	
313	<pre>/**</pre>	
314	<pre> *</pre>	
315	<pre> * @desc This function sends a list of keys to the secure streaming service for later use</pre>	
316	<pre> * @param p_keys A record of keys</pre>	
317	<pre> * @verdict</pre>	
318	<pre> * pass on receipt of a successful StopStreamingResponse</pre>	
319	<pre> * @verdict</pre>	
320	<pre> * fail on receipt of a wrong message</pre>	
321	<pre> * @verdict</pre>	
322	<pre> * inconc in case the guard timer expires</pre>	
323	<pre> */</pre>	
324	<pre>function f_data_secure_setKeys(ListOfSTKMKeys p_keys) runs on BCastNetworkDataComponent {</pre>	
325	<pre> var UpdateKeyRequest v_Request := valueof(m_def_UpdateKeyRequest(p_keys));</pre>	
326		
327	<pre> sc.send(v_Request);</pre>	
328	<pre> t_wait.start(PX_TWAIT);</pre>	
329		
330	<pre> alt {</pre>	
331	<pre> [] sc.receive(UpdateKeyResponse: {0, *}) {</pre>	
332	<pre> t_wait.stop;</pre>	
333	<pre> log("Verdict Info: The secure streaming service sends a UpdateKeyResponse");</pre>	
334	<pre> setverdict(pass);</pre>	
335	<pre> }</pre>	
336	<pre> [] sc.receive(UpdateKeyResponse: {*, *}) {</pre>	
337	<pre> t_wait.stop;</pre>	
338	<pre> log("Verdict Info: The secure streaming service does not send a UpdateKeyResponse");</pre>	
339	<pre> setverdict(fail);</pre>	
340	<pre> }</pre>	
341	<pre> [] t_wait.timeout {</pre>	
342	<pre> log("Verdict Info: The secure streaming service does not send a UpdateKeyResponse within the guard time");</pre>	
343	<pre> setverdict(inconc);</pre>	
344	<pre> }</pre>	
345	<pre> }</pre>	
346	<pre>}</pre>	
347		
348	<pre>// change 3 (WK34): Service Protection: secure streaming service</pre>	
349	<pre>/**</pre>	
350	<pre> *</pre>	
351	<pre> * @desc This function activates a certain key in the key list for encoding</pre>	
352	<pre> * @param p_key A certain key</pre>	
353	<pre> * @param p_keyNumberInList A key number (based on the previously provisioned key list)</pre>	
354	<pre> * @verdict</pre>	
355	<pre> * pass on receipt of a successful StopStreamingResponse</pre>	

Datei: LibBCast_BCastNetworkData_Functions.ttcn (Fortsetzung)

356	<pre> * @verdict 357 * fail on receipt of a wrong message 358 * @verdict 359 * inconc in case the guard timer expires 360 */ 361 function f_data_secure_activateKey(in SecureStreamingKey p_key, in integer p_keyNum 362 var ActivateKeyRequest v_Request := valueof(m_def_ActivateKeyRequest(p_key)) 363 364 sc.send(v_Request); 365 t_wait.start(PX_TWAIT); 366 367 alt { 368 [] sc.receive(ActivateKeyResponse: {0, *}) { 369 t_wait.stop; 370 log("Verdict Info: The secure streaming service sends a Act 371 setverdict(pass); 372 } 373 [] sc.receive(ActivateKeyResponse: {*, *}) { 374 t_wait.stop; 375 log("Verdict Info: The secure streaming service does not se 376 setverdict(fail); 377 } 378 [] t_wait.timeout { 379 log("Verdict Info: The secure streaming service does not se 380 setverdict(inconc); 381 } 382 } 383 }</pre>			
384	<pre> } // end group behaviorFunctions 385 386 group commonFunctions { 387 // change 15 (WK18): sets start and stop times 388 /** 389 * 390 * @desc 391 * initialize the start and end time of the BCastNetworkDataComponent. 392 * @param 393 * p_startTime The Start Time which is one hour after test execution 394 * @param 395 * p_endTime The End Time which is set to a value one our after the start time 396 */ 397 function f_data_InitStartEndTime(UInt p_startTime, UInt p_endTime) runs on BCastNet 398 v_startTime := p_startTime; 399 v_endTime := p_endTime; 400 } 401 } // end group commonFunctions 402 403 } // end group subFunctions 404 }</pre>	=	231	<pre> } // end group behaviorFunctions 232 233 group commonFunctions { 234 // change 15 (WK18): sets start and stop times 235 /** 236 * 237 * @desc 238 * initialize the start and end time of the BCastNetworkDataComponent. 239 * @param 240 * p_startTime The Start Time which is one hour after test execution 241 * @param 242 * p_endTime The End Time which is set to a value one our after the start time 243 */ 244 function f_data_InitStartEndTime(UInt p_startTime, UInt p_endTime) runs on BCastNet 245 v_startTime := p_startTime; 246 v_endTime := p_endTime; 247 } 248 } // end group commonFunctions 249 250 } // end group subFunctions 251 }</pre>

Datei: LibBCast_Common_Templates.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI OMA BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module provides common templates. 9 */ 10 module LibBCast_Common_Templates { 11 import from LibBCast_ServicePrimitives_TypesAndValues all; 12 import from LibBCast_ServiceGuide_TypesAndValues all; 13 import from LibBCast_ServicePrimitives_TypesAndValues all; 14 import from LibBCast_Common_TypesAndValues all; 15 import from LibCommon_BasicTypesAndValues all; 16 import from LibCommon_TextStrings all; // change 17 (WK 6): additional param 17 // change 3 (WK34): Service Protection: secure streaming service 18 import from LibCommon_DataStrings all;</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI OMA BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module provides common templates. 9 */ 10 module LibBCast_Common_Templates { 11 import from LibBCast_ServicePrimitives_TypesAndValues all; 12 import from LibBCast_ServiceGuide_TypesAndValues all; 13 import from LibBCast_ServicePrimitives_TypesAndValues all; 14 import from LibBCast_Common_TypesAndValues all; 15 import from LibCommon_BasicTypesAndValues all; 16 import from LibCommon_TextStrings all; // change 17 (WK 6): additional param</pre>
<pre>19 20 group flute { 21 /** 22 * 23 * @desc 24 * This template creates a StartFluteSessionRequest. 25 * @param 26 * p_Id The flute session id 27 * @param 28 * p_IP The Destination IP address 29 * @param 30 * p_Port The Destination Port 31 * @param 32 * p_FluteUpdateID The Flute Instance ID of the Flute Session. 33 * @param 34 * p_FDT The File Delivery Table (FDT) 35 * @param 36 * p_Payload The payload which should be send 37 */ 38 template StartFluteSessionRequest m_def_StartFluteSessionRequest(39 UInt p_Id, 40 charstring p_IP, // change 17 (WK 6): additional 41 UInt16 p_Port, // change 17 (WK 6): additional 42 UInt32 p_FluteUpdateID, // CR (WK 34/2008): Flute Upd 43 template FDT_InstanceType p_FDT, 44 template FluteSessionPayloadType p_Payload) := { 45 FluteSessionId := p_Id, 46 FluteSessionIp := p_IP, // change 17 (WK 6): additional 47 FluteSessionIpType := IPv4, // change 17 (WK 6): additional 48 FluteSessionPort := p_Port, // change 17 (WK 6): additional 49 FluteUpdateID := p_FluteUpdateID, // CR (WK 34/2008): Flute Upd 50 FDT_Instance := p_FDT, 51 FluteSessionPayload := p_Payload 52 } 53 54 /** 55 * 56 * @desc 57 * This template creates a Flute Session Payload 58 * @param 59 * p_SGDD The SGDD which should be included 60 */ 61 template FluteSessionPayloadType m_SGDD_FluteSessionPayload(ServiceGuideDeliveryDescriptorT 62 ServiceGuideDeliveryDescriptors := { p_SGDD } 63 } 64 65 /** 66 * 67 * @desc 68 * This template creates a Flute Session Payload. 69 * @param 70 * p_SGDU The SGDU which should be included 71 */</pre>	=	<pre>19 20 group flute { 21 /** 22 * 23 * @desc 24 * This template creates a StartFluteSessionRequest. 25 * @param 26 * p_Id The flute session id 27 * @param 28 * p_IP The Destination IP address 29 * @param 30 * p_Port The Destination Port 31 * @param 32 * p_FluteUpdateID The Flute Instance ID of the Flute Session. 33 * @param 34 * p_FDT The File Delivery Table (FDT) 35 * @param 36 * p_Payload The payload which should be send 37 */ 38 template StartFluteSessionRequest m_def_StartFluteSessionRequest(39 UInt p_Id, 40 charstring p_IP, // change 17 (WK 6): additional 41 UInt16 p_Port, // change 17 (WK 6): additional 42 UInt32 p_FluteUpdateID, // CR (WK 34/2008): Flute Upd 43 template FDT_InstanceType p_FDT, 44 template FluteSessionPayloadType p_Payload) := { 45 FluteSessionId := p_Id, 46 FluteSessionIp := p_IP, // change 17 (WK 6): additional 47 FluteSessionIpType := IPv4, // change 17 (WK 6): additional 48 FluteSessionPort := p_Port, // change 17 (WK 6): additional 49 FluteUpdateID := p_FluteUpdateID, // CR (WK 34/2008): Flute Upd 50 FDT_Instance := p_FDT, 51 FluteSessionPayload := p_Payload 52 } 53 54 /** 55 * 56 * @desc 57 * This template creates a Flute Session Payload 58 * @param 59 * p_SGDD The SGDD which should be included 60 */ 61 template FluteSessionPayloadType m_SGDD_FluteSessionPayload(ServiceGuideDeliveryDescriptorT 62 ServiceGuideDeliveryDescriptors := { p_SGDD } 63 } 64 65 /** 66 * 67 * @desc 68 * This template creates a Flute Session Payload. 69 * @param 70 * p_SGDU The SGDU which should be included 71 */</pre>

Datei: LibBCast_Common_Templates.ttcn (Fortsetzung)

72	template FluteSessionPayloadType m_SGDU_FluteSessionPayload(ServiceGuideDeliveryUnit p_SGDU	70	template FluteSessionPayloadType m_SGDU_FluteSessionPayload(ServiceGuideDeliveryUnit p_SGDU
73	ServiceGuideDeliveryUnits := { p_SGDU }	71	ServiceGuideDeliveryUnits := { p_SGDU }
74	}	72	}
75		73	
76	// change 13 (WK 6): send DVB-H bootstrap files via FLUTE	74	// change 13 (WK 6): send DVB-H bootstrap files via FLUTE
77	/**	75	/**
78	*	76	*
79	* @desc	77	* @desc
80	* This template creates a Flute Session Payload.	78	* This template creates a Flute Session Payload.
81	* @param	79	* @param
82	* p_EPDD The ESG Provider Discovery Descriptor which should be included	80	* p_EPDD The ESG Provider Discovery Descriptor which should be included
83	* @param	81	* @param
84	* p_EAD The ESG Access Descriptor which should be included	82	* p_EAD The ESG Access Descriptor which should be included
85	*/	83	*/
86	template FluteSessionPayloadType m_DVBH_ESG_Descriptor_FluteSessionPayload(ESGProviderDisc	84	template FluteSessionPayloadType m_DVBH_ESG_Descriptor_FluteSessionPayload(ESGProviderDisc
87	ESGDescriptor := {	85	ESGDescriptor := {
88	p_EPDD,	86	p_EPDD,
89	p_EAD	87	p_EAD
90	}	88	}
91	}	89	}
92	}	90	}
93		91	
94	group http {	92	group http {
95	// change 16 (WK18) templates for incoming HTTP POST over interaction channel	93	// change 16 (WK18) templates for incoming HTTP POST over interaction channel
96	template HttpPostIndication mw_HttpPostIndication(template KeyValuePair p_pair) := {	94	template HttpPostIndication mw_HttpPostIndication(template KeyValuePair p_pair) := {
97	keyValuePairs := {p_pair, *}	95	keyValuePairs := {p_pair, *}
98	}	96	}
99		97	
100	template HttpPostIndication mw_HttpPostIndication_empty := {	98	template HttpPostIndication mw_HttpPostIndication_empty := {
101	keyValuePairs := omit	99	keyValuePairs := omit
102	}	100	}
103		101	
104	template HttpPostIndication mw_HttpPostIndication_any := {	102	template HttpPostIndication mw_HttpPostIndication_any := {
105	keyValuePairs := *	103	keyValuePairs := *
106	}	104	}
107		105	
108	template HttpPostIndication mw_HttpPostIndication_unspecific_sgdd := {	106	template HttpPostIndication mw_HttpPostIndication_unspecific_sgdd := {
109	keyValuePairs := { m_KeyValuePair("type", "sgdd"),*}	107	keyValuePairs := { m_KeyValuePair("type", "sgdd"),*}
110	}	108	}
111		109	
112	template HttpPostIndication mw_HttpPostIndication_unspecific_sgdu := {	110	template HttpPostIndication mw_HttpPostIndication_unspecific_sgdu := {
113	keyValuePairs := { m_KeyValuePair("type", "sgdu"),*}	111	keyValuePairs := { m_KeyValuePair("type", "sgdu"),*}
114	}	112	}
115		113	
116	template HttpPostIndication mw_HttpPostIndication_unspecific_sgdd_sgdu := {	114	template HttpPostIndication mw_HttpPostIndication_unspecific_sgdd_sgdu := {
117	keyValuePairs := { m_KeyValuePair("type", "sgdd+sgdu"),*}	115	keyValuePairs := { m_KeyValuePair("type", "sgdd+sgdu"),*}
118	}	116	}
119		117	
120	template HttpPostIndication mw_HttpPostIndication_specific_sgdd(charstring p_sgddId) := {	118	template HttpPostIndication mw_HttpPostIndication_specific_sgdd(charstring p_sgddId) := {
121	keyValuePairs := { m_KeyValuePair("sgddID", p_sgddId),*}	119	keyValuePairs := { m_KeyValuePair("sgddID", p_sgddId),*}
122	}	120	}
123		121	
124	template KeyValuePair m_KeyValuePair (charstring p_key, charstring p_value) := {	122	template KeyValuePair m_KeyValuePair (charstring p_key, charstring p_value) := {
125	key := p_key, valuePart := p_value	123	key := p_key, valuePart := p_value
126	}	124	}
127		125	
128	// change 17 (WK18): no connection id; additional header 'Content-Type' see OMA SG 5.4.3.1	126	// change 17 (WK18): no connection id; additional header 'Content-Type' see OMA SG 5.4.3.1
129	/**	127	/**
130	* @desc This template creates a HttpResponseRequest for an HTTP POST request over interact	128	* @desc This template creates a HttpResponseRequest for an HTTP POST request over interact
131	* Should be used for sending SGDD or/and SGDU.	129	* Should be used for sending SGDD or/and SGDU.
132	*	130	*
133	* @param p_ServiceGuidePayload The payload which should be send.	131	* @param p_ServiceGuidePayload The payload which should be send.
134	*/	132	*/
135	template HttpResponseRequest m_def_HttpResponseRequest(133	template HttpResponseRequest m_def_HttpResponseRequest(
136	ServiceGuidePayload p_ServiceGuidePayload) := {	134	ServiceGuidePayload p_ServiceGuidePayload) := {
137	responseCode := 200,	135	responseCode := 200,
138	responseHeaders := {	136	responseHeaders := {
139	{name := "Content-Type", valuePart := "application/octet-stream"}	137	{name := "Content-Type", valuePart := "application/octet-stream"}
140	},	138	},
141	responsePayload := {	139	responsePayload := {
142	serviceGuidePayload := p_ServiceGuidePayload	140	serviceGuidePayload := p_ServiceGuidePayload

Datei: LibBCast_Common_Templates.ttcn (Fortsetzung)

143	}	141	}
144	}	142	}
145		143	
146	// change 18 (WK18)(additional to 17) this template replaces other 'ServiceGuidePayload' te	144	// change 18 (WK18)(additional to 17) this template replaces other 'ServiceGuidePayload' te
147	/**	145	/**
148	*	146	*
149	* @desc Default ServiceGuidePayload template wich includes only a SGDU	147	* @desc Default ServiceGuidePayload template wich includes only a SGDU
150	* @param p_SGDU the included SGDU	148	* @param p_SGDU the included SGDU
151	*/	149	*/
152	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDU(ServiceGuideDeliveryUnit p_SGDU	150	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDU(ServiceGuideDeliveryUnit p_SGDU
153	serviceGuideResponse := {	151	serviceGuideResponse := {
154	status := 0,	152	status := 0,
155	ServiceGuideDeliveryDescriptors := omit,	153	ServiceGuideDeliveryDescriptors := omit,
156	privateExt := omit },	154	privateExt := omit },
157	serviceGuideDeliveryUnit := p_SGDU	155	serviceGuideDeliveryUnit := p_SGDU
158	}	156	}
159	/**	157	/**
160	*	158	*
161	*	159	*
162	* @desc Default ServiceGuidePayload template wich includes only a SGDD	160	* @desc Default ServiceGuidePayload template wich includes only a SGDD
163	* @param p_SGDD the included SGDD	161	* @param p_SGDD the included SGDD
164	*/	162	*/
165	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDD(ServiceGuideDeliveryDescriptorT	163	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDD(ServiceGuideDeliveryDescriptorT
166	serviceGuideResponse := {	164	serviceGuideResponse := {
167	status := 0,	165	status := 0,
168	ServiceGuideDeliveryDescriptors := { p_SGDD },	166	ServiceGuideDeliveryDescriptors := { p_SGDD },
169	privateExt := omit },	167	privateExt := omit },
170	serviceGuideDeliveryUnit := omit	168	serviceGuideDeliveryUnit := omit
171	}	169	}
172	/**	170	/**
173	*	171	*
174	*	172	*
175	* @desc Default ServiceGuidePayload template wich includes a SGDU and SGDD	173	* @desc Default ServiceGuidePayload template wich includes a SGDU and SGDD
176	* @param p_SGDU the included SGDU	174	* @param p_SGDU the included SGDU
177	* @param p_SGDD the included SGDD	175	* @param p_SGDD the included SGDD
178	*/	176	*/
179	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDU_SGDD(177	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDU_SGDD(
180	ServiceGuideDeliveryUnit p_SGDU,	178	ServiceGuideDeliveryUnit p_SGDU,
181	ServiceGuideDeliveryDescriptorType p_SGDD) := {	179	ServiceGuideDeliveryDescriptorType p_SGDD) := {
182	serviceGuideResponse := {	180	serviceGuideResponse := {
183	status := 0,	181	status := 0,
184	ServiceGuideDeliveryDescriptors := { p_SGDD },	182	ServiceGuideDeliveryDescriptors := { p_SGDD },
185	privateExt := omit },	183	privateExt := omit },
186	serviceGuideDeliveryUnit := p_SGDU	184	serviceGuideDeliveryUnit := p_SGDU
187	}	185	}
188		186	
189	// change 18 (WK18)(additional to 17) this template replaces other 'ServiceGuidePayload' te	187	// change 18 (WK18)(additional to 17) this template replaces other 'ServiceGuidePayload' te
190	/**	188	/**
191	*	189	*
192	* @desc Default ServiceGuidePayload template wich includes a SGDU and SGDD	190	* @desc Default ServiceGuidePayload template wich includes a SGDU and SGDD
193	* @param p_SGDU the included SGDU	191	* @param p_SGDU the included SGDU
194	* @param p_SGDDList the included SGDD as a list	192	* @param p_SGDDList the included SGDD as a list
195	*/	193	*/
196	template ServiceGuidePayload m_def_ServiceGuidePayload_SGDD_SGDU(194	template ServiceGuidePayload m_def_ServiceGuidePayload SGDD_SGDU(
197	template ServiceGuideDeliveryDescriptorList p_SGDDList,	195	template ServiceGuideDeliveryDescriptorList p_SGDDList,
198	template ServiceGuideDeliveryUnit p_SGDU) := {	196	template ServiceGuideDeliveryUnit p_SGDU) := {
199	serviceGuideResponse := {	197	serviceGuideResponse := {
200	status := 0,	198	status := 0,
201	ServiceGuideDeliveryDescriptors := p_SGDDList,	199	ServiceGuideDeliveryDescriptors := p_SGDDList,
202	privateExt := omit },	200	privateExt := omit },
203	serviceGuideDeliveryUnit := p_SGDU	201	serviceGuideDeliveryUnit := p_SGDU
204	}	202	}
205	}	203	}
206		204	
207		205	
208	group streaming {	206	group streaming {
209		207	
210	// change 14 (WK18): dynamic setting of destination IP	208	// change 14 (WK18): dynamic setting of destination IP
211	/**	209	/**
212	*	210	*
213	* @desc This is a default template for the StartStreamingRequest.	211	* @desc This is a default template for the StartStreamingRequest.

Datei: LibBCast_Common_Templates.ttcn (Fortsetzung)

214	* @param p_StreamId The stream id		212	* @param p_StreamId The stream id
215	* @param p_FileName The name of the file which should be streamed		213	* @param p_FileName The name of the file which should be streamed
216	* @param p_SDP The used SDP		214	* @param p_SDP The used SDP
217	* @param p_DstIp The destination IP address		215	* @param p_DstIp The destination IP address
218	*/		216	*/
219	template StartStreamingRequest m_def_StartStreamingRequest(217	template StartStreamingRequest m_def_StartStreamingRequest(
220	UInt p_StreamId, charstring p_FileName, charstring p_SDP, charstring p_DstIp)		218	UInt p_StreamId, charstring p_FileName, charstring p_SDP, charstring p_DstIp)
221	streamId := p_StreamId,		219	streamId := p_StreamId,
222	fileName := p_FileName,		220	fileName := p_FileName,
223	sdp := p_SDP,		221	sdp := p_SDP,
224	destinationIpAddress := p_DstIp, // change 14 (WK18): dynamic setting of destination IP address		222	destinationIpAddress := p_DstIp, // change 14 (WK18): dynamic setting of destination IP address
225	contentProtection := omit,		223	contentProtection := omit,
226	serviceProtection := omit		224	serviceProtection := omit
227	}		225	}
228			226	
229	/**		227	/**
230	*		228	*
231	* @desc This is a default template for a StopStreamingRequest		229	* @desc This is a default template for a StopStreamingRequest
232	* @param p_StreamId The id of the stream wich should stopped		230	* @param p_StreamId The id of the stream wich should stopped
233	*/		231	*/
234	template StopStreamingRequest m_def_StopStreamingRequest(UInt p_StreamId) := {		232	template StopStreamingRequest m_def_StopStreamingRequest(UInt p_StreamId) := {
235	streamId := p_StreamId		233	streamId := p_StreamId
236	}		234	}
237		+-		
238	// change 3 (WK34): Service Protection: secure streaming service			
239	template UpdateKeyRequest m_def_UpdateKeyRequest(ListOfSTKMKeys p_keys) := {			
240	keys := p_keys			
241	}			
242				
243	// change 3 (WK34): Service Protection: secure streaming service			
244	template ActivateKeyRequest m_def_ActivateKeyRequest(integer p_numberInKeyList, SecureStreamingKeyList p_keyList)			
245	keyListNumber := p_numberInKeyList,			
246	activeKey := p_key			
247	}			
248				
249	// change 3 (WK34): Service Protection: secure streaming service			
250	/**			
251	*			
252	* @desc This is a default template for the StartSecureStreamingRequest.			
253	* @param p_DstIp The destination IP address			
254	* @param p_Ssrc The synchronization source			
255	*/			
256	template StartSecureStreamingRequest m_def_StartSecureStreamingRequest(charstring p_DstIp, integer p_Ssrc)			
257	destinationIpAddress := p_DstIp,			
258	ssrc := p_Ssrc			
259	}			
260				
261	// change 3 (WK34): Service Protection: secure streaming service			
262	/**			
263	*			
264	* @desc This is a default template for a StopSecureStreamingRequest			
265	*/			
266	template StopSecureStreamingRequest m_def_StopSecureStreamingRequest() := {			
267	}			
268	}	=	235	}
269			236	
270	group transfer {		237	group transfer {
271	/**		238	/**
272	*		239	*
273	* @desc This is a default template for a StartFileTransferRequest.		240	* @desc This is a default template for a StartFileTransferRequest.
274	* @param p_TransferId The transfer id		241	* @param p_TransferId The transfer id
275	* @param p_FileList The list of file		242	* @param p_FileList The list of file
276	* @param p_Sdp The used SDP		243	* @param p_Sdp The used SDP
277	*/		244	*/
278	template StartFileTransferRequest m_def_StartFileTransferRequest(245	template StartFileTransferRequest m_def_StartFileTransferRequest(
279	UInt p_TransferId,		246	UInt p_TransferId,
280	charstring p_IP, // change 19 (WK18): (additional to 11) additional IP address		247	charstring p_IP, // change 19 (WK18): (additional to 11) additional IP address
281	UInt16 p_Port, // change 19 (WK18): (additional to 11) additional port		248	UInt16 p_Port, // change 19 (WK18): (additional to 11) additional port
282	UInt32 p_FluteUpdateID, // CR (WK 34/2008): Flute Update Signalling		249	UInt32 p_FluteUpdateID, // CR (WK 34/2008): Flute Update Signalling
283	template FDT_InstanceType p_FDT,		250	template FDT_InstanceType p_FDT,
284	TransferFileList p_FileList,		251	TransferFileList p_FileList,

Datei: LibBCast_Common_Templates.ttcn (Fortsetzung)

285	charstring p_Sdp) := {	252	charstring p_Sdp) := {
286	transferId := p_TransferId,	253	transferId := p_TransferId,
287	FluteSessionIp := p_IP, // change 20 (WK18): (additional to 11) f	254	FluteSessionIp := p_IP, // change 20 (WK18): (additional to 11) f
288	FluteSessionIpType := IPv4, // change 20 (WK18): (additional to 11) f	255	FluteSessionIpType := IPv4, // change 20 (WK18): (additional to 11) f
289	FluteSessionPort := p_Port, // change 20 (WK18): (additional to 11) f	256	FluteSessionPort := p_Port, // change 20 (WK18): (additional to 11) f
290	FluteUpdateID := p_FluteUpdateID, // CR (WK 34/2008): Flute Update Signalli	257	FluteUpdateID := p_FluteUpdateID, // CR (WK 34/2008): Flute Update Signalli
291	FDT_Instance := p_FDT, // change 20 (WK18): (additional to 11) f	258	FDT_Instance := p_FDT, // change 20 (WK18): (additional to 11) f
292	fileList := p_FileList,	259	fileList := p_FileList,
293	sdp := p_Sdp,	260	sdp := p_Sdp,
294	contentProtection := omit,	261	contentProtection := omit,
295	serviceProtection := omit	262	serviceProtection := omit
296	}	263	}
297		264	
298	/**	265	/**
299	*	266	*
300	* @desc This is a default template for a StopFileTransferRequest	267	* @desc This is a default template for a StopFileTransferRequest
301	* @param p_TransferId The id of the stream wich should stopped	268	* @param p_TransferId The id of the stream wich should stopped
302	*/	269	*/
303	template StopFileTransferRequest m_def_StopFileTransferRequest(UInt p_TransferId) := {	270	template StopFileTransferRequest m_def_StopFileTransferRequest(UInt p_TransferId) := {
304	transferId := p_TransferId	271	transferId := p_TransferId
305	}	272	}
306		273	
307		274	
308	/**	275	/**
309	*	276	*
310	* @desc This is a default ZipFile template	277	* @desc This is a default ZipFile template
311	* @param p_Name The name of the zip file	278	* @param p_Name The name of the zip file
312	* @param p_FileList The content file list of the zip file	279	* @param p_FileList The content file list of the zip file
313	*/	280	*/
314	template ZipFile m_def_ZipFile(charstring p_Name, FileReferenceList p_FileList) := {	281	template ZipFile m_def_ZipFile(charstring p_Name, FileReferenceList p_FileList) := {
315	name := p_Name,	282	name := p_Name,
316	fileList := p_FileList	283	fileList := p_FileList
317	}	284	}
318		285	
319	}	286	}
320	}	287	}

Datei: LibBCast_Interface.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI OMA BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies port and component types that specify the 9 * interface required by BCAST library functions. 10 */ 11 module LibBCast_Interface { 12 import from LibBCast_UpperTesterPrimitives_TypesAndValues all ; 13 import from LibBCast_ServicePrimitives_TypesAndValues all ; 14 import from LibCommon_BasicTypesAndValues { // change 01 (WK 6): global time definition 15 type UInt; 16 } 17 18 group libraryComponentTypes { 19 20 /** 21 * 22 * @desc 23 * This type of component is used by library functions which 24 * specify BCAST control behavior, i.e., handling of BCAST 25 * SGDDs, SGDU, and Interactivity Media Documents. 26 */ 27 type component BCastNetworkControlComponent { 28 port BroadcastPort ac; 29 port BroadcastPort dc; 30 port InteractionPort ic; 31 var UInt v_TOI4SGDD := 10; // change 22 (WK18): (additional to 6) set default value 32 var UInt v_TOI4SGDU := 20; // change 22 (WK18): (additional to 6) set default value 33 port SmsPort sms; 34 port MmsPort mms; 35 timer t_wait; 36 var UInt v_startTime := 0; // change 01 (WK 6): global time definition 37 var UInt v_endTime := 0; // change 01 (WK 6): global time definition 38 } 39 40 /** 41 * 42 * @desc 43 * This component type is used by library functions which 44 * control streaming or file servers. 45 */ 46 type component BCastNetworkDataComponent { 47 port StreamServerCtrlPort ssc; 48 port FileServerControlPort fsc;</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI OMA BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies port and component types that specify the 9 * interface required by BCAST library functions. 10 */ 11 module LibBCast_Interface { 12 import from LibBCast_UpperTesterPrimitives_TypesAndValues all ; 13 import from LibBCast_ServicePrimitives_TypesAndValues all ; 14 import from LibCommon_BasicTypesAndValues { // change 01 (WK 6): global time definition 15 type UInt; 16 } 17 18 group libraryComponentTypes { 19 20 /** 21 * 22 * @desc 23 * This type of component is used by library functions which 24 * specify BCAST control behavior, i.e., handling of BCAST 25 * SGDDs, SGDU, and Interactivity Media Documents. 26 */ 27 type component BCastNetworkControlComponent { 28 port BroadcastPort ac; 29 port BroadcastPort dc; 30 port InteractionPort ic; 31 var UInt v_TOI4SGDD := 10; // change 22 (WK18): (additional to 6) set default value 32 var UInt v_TOI4SGDU := 20; // change 22 (WK18): (additional to 6) set default value 33 port SmsPort sms; 34 port MmsPort mms; 35 timer t_wait; 36 var UInt v_startTime := 0; // change 01 (WK 6): global time definition 37 var UInt v_endTime := 0; // change 01 (WK 6): global time definition 38 } 39 40 /** 41 * 42 * @desc 43 * This component type is used by library functions which 44 * control streaming or file servers. 45 */ 46 type component BCastNetworkDataComponent { 47 port StreamServerCtrlPort ssc; 48 port FileServerControlPort fsc;</pre>
<pre>49 // change 3 (WK34): Service Protection: secure streaming service 50 port SecureStreamServerCtrlPort sc;</pre>	+ -	
<pre>51 timer t_wait; 52 var UInt v_startTime := 0; // change 21 (WK18): remaining global time definition 53 var UInt v_endTime := 0; // change 21 (WK18): remaining global time definition 54 } 55 56 /** 57 * 58 * @desc 59 * This component type is used by library functions which 60 * specify upper tester behavior. 61 */ 62 type component UpperTesterComponent { 63 port UtpPort utp; 64 timer t_wait; 65 } 66 67 } 68 69 group portDefinitions { 70 /** 71 * @desc</pre>	=	<pre>49 timer t_wait; 50 var UInt v_startTime := 0; // change 21 (WK18): remaining global time definition 51 var UInt v_endTime := 0; // change 21 (WK18): remaining global time definition 52 } 53 54 /** 55 * 56 * @desc 57 * This component type is used by library functions which 58 * specify upper tester behavior. 59 */ 60 type component UpperTesterComponent { 61 port UtpPort utp; 62 timer t_wait; 63 } 64 65 } 66 67 group portDefinitions { 68 /** 69 * @desc</pre>

Datei: LibBCast_Interface.ttcn (Fortsetzung)

72	* This port type is used to configure and interact with the bradcast	70	* This port type is used to configure and interact with the bradcast
73	* channel.	71	* channel.
74	*/	72	*/
75	type port BroadcastPort message {	73	type port BroadcastPort message {
76	inout StartFluteSessionRequest;	74	inout StartFluteSessionRequest;
77	inout StartFluteSessionResponse;	75	inout StartFluteSessionResponse;
78	}	76	}
79		77	
80	/**	78	/**
81	* @desc	79	* @desc
82	* This port type is used to configure and interact with the interaction	80	* This port type is used to configure and interact with the interaction
83	* channel.	81	* channel.
84	*/	82	*/
85	type port InteractionPort message {	83	type port InteractionPort message {
86	inout HttpResponseRequest;	84	inout HttpResponseRequest;
87	inout HttpResponseResponse;	85	inout HttpResponseResponse;
88	inout HttpSubscriptionRequestIndication;	86	inout HttpSubscriptionRequestIndication;
89	inout HttpPostIndication;	87	inout HttpPostIndication;
90	}	88	}
91		89	
92	/**	90	/**
93	* @desc	91	* @desc
94	* This port type is used to configure streaming servers.	92	* This port type is used to configure streaming servers.
95	*/	93	*/
96	type port StreamServerCtrlPort message {	94	type port StreamServerCtrlPort message {
97	inout StartStreamingRequest;	95	inout StartStreamingRequest;
98	inout StartStreamingResponse;	96	inout StartStreamingResponse;
99	inout StopStreamingRequest;	97	inout StopStreamingRequest;
100	inout StopStreamingResponse;	98	inout StopStreamingResponse;
101	}		
102			
103	// change 3 (WK34): Service Protection: secure streaming service		
104	/**		
105	* @desc		
106	* This port type is used to configure the secure streaming instance.		
107	*/		
108	type port SecureStreamServerCtrlPort message {		
109	out StartSecureStreamingRequest;		
110	in StartSecureStreamingResponse;		
111	out StopSecureStreamingRequest;		
112	in StopSecureStreamingResponse;		
113	out UpdateKeyRequest;		
114	in UpdateKeyResponse;		
115	out ActivateKeyRequest;		
116	in ActivateKeyResponse;		
117	}		
118			
119	/**		
120	* @desc		
121	* This port type is used to interact with BCAST upper testers.		
122	*/		
123	type port UtpPort message {		
124	inout PowerOnTerminalRequest;		
125	inout PowerOffTerminalRequest;		
126	inout RunBCastApplicationRequest;		
127	inout GetServiceGuideRequest;		
128	inout UpdateServiceGuideRequest;		
129	inout CheckServiceRequest;		
130	inout SelectServiceRequest;		
131	inout CheckContentPresenceRequest;		
132	inout SelectContentRequest;		
133	inout SelectLanguageRequest;		
134	inout ClearServiceGuideCacheRequest;		
135	inout GenericUtsResponse;		
136	inout CheckBrowserRequest;		
137	inout CheckPreviewRequest;		
138	inout CheckFileRequest;		
139	inout CheckVideoRequest;		
140	inout CheckPurchaseInfoRequest;		
141	inout CheckLanguageRequest;		
142	inout CheckInteractivityRequest;		

Datei: LibBCast_Interface.ttcn (Fortsetzung)

143	inout CheckInteractivityChoicesRequest;	125	inout CheckInteractivityChoicesRequest;
144	inout PurchaseServiceRequest;	126	inout PurchaseServiceRequest;
145	inout SelectFileRequest;	127	inout SelectFileRequest;
146	inout SelectInteractivityRequest;	128	inout SelectInteractivityRequest;
147	inout CheckFileReceivedRequest;	129	inout CheckFileReceivedRequest;
148	inout SelectInteractivityChoiceRequest;	130	inout SelectInteractivityChoiceRequest;
149	inout UseServiceRequest;	131	inout UseServiceRequest;
150	}	132	}
151		133	
152	/**	134	/**
153	*	135	*
154	* @desc This port type is used to interact with sms centers	136	* @desc This port type is used to interact with sms centers
155	*/	137	*/
156	type port SmsPort message {	138	type port SmsPort message {
157	inout SmsGetIndication;	139	inout SmsGetIndication;
158	}	140	}
159		141	
160	/**	142	/**
161	* @desc This port type is used to interact with mms centers	143	* @desc This port type is used to interact with mms centers
162	*/	144	*/
163	type port MmsPort message {	145	type port MmsPort message {
164	inout MmsGetIndication;	146	inout MmsGetIndication;
165	}	147	}
166		148	
167	/**	149	/**
168	*	150	*
169	* @desc This port type is used to interact with the file servers in order	151	* @desc This port type is used to interact with the file servers in order
170	* to deliver files to a terminal	152	* to deliver files to a terminal
171	*/	153	*/
172	type port FileServerControlPort message {	154	type port FileServerControlPort message {
173	inout StartFileTransferRequest;	155	inout StartFileTransferRequest;
174	inout StartFileTransferResponse;	156	inout StartFileTransferResponse;
175	inout StopFileTransferRequest;	157	inout StopFileTransferRequest;
176	inout StopFileTransferResponse;	158	inout StopFileTransferResponse;
177	}	159	}
178	}	160	}
179	} // end module LibBCast_Interface	161	} // end module LibBCast_Interface

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn

<pre>1 /** 2 * 3 * @author 4 * ETSI CTI OMA BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies message types and values that are used to 9 * sconfigure the transport of BCAST messages. There are no requirements 10 * regarding the encoding of these primitives. 11 * @remark 12 * End users should be aware that any changes made to this file may 13 * negatively affect the interworking of test suite code with the BCast 14 * SUT Adapter during test execution. 15 */ 16 module LibBCast_ServicePrimitives_TypesAndValues { 17 import from LibBCast_FileStreamDistribution_TypesAndValues all; 18 import from LibBCast_ServiceGuide_TypesAndValues all; 19 import from LibCommon_BasicTypesAndValues all; 20 import from LibBCast_FileStreamDistribution_TypesAndValues all; 21 import from LibBCast_Common_TypesAndValues all; 22 import from LibCommon_TextStrings all;</pre>	=	<pre>1 /** 2 * 3 * @author 4 * ETSI CTI OMA BCAST CON Project 5 * @version 6 * \$Id\$ 7 * @desc 8 * This module specifies message types and values that are used to 9 * sconfigure the transport of BCAST messages. There are no requirements 10 * regarding the encoding of these primitives. 11 * @remark 12 * End users should be aware that any changes made to this file may 13 * negatively affect the interworking of test suite code with the BCast 14 * SUT Adapter during test execution. 15 */ 16 module LibBCast_ServicePrimitives_TypesAndValues { 17 import from LibBCast_FileStreamDistribution_TypesAndValues all; 18 import from LibBCast_ServiceGuide_TypesAndValues all; 19 import from LibCommon_BasicTypesAndValues all; 20 import from LibBCast_FileStreamDistribution_TypesAndValues all; 21 import from LibBCast_Common_TypesAndValues all; 22 import from LibCommon_TextStrings all;</pre>
<pre>23 // change 3 (WK34): Service Protection: secure streaming service 24 import from LibCommon_DataStrings all;</pre>	+ -	
<pre>25 26 /** 27 * 28 * @desc 29 * This group specifies information to be forwarded from TTCN-3 to 30 * the FLUTE application 31 */ 32 group fluteConfigPrimitives { 33 34 /** 35 * 36 * @desc 37 * Send to the SA to request the start of a flute session 38 * @member 39 * fluteSessionId The flute session identifier 40 * @member 41 * FDT_Instance The used FDT 42 * @member 43 * fluteSessionPayload Includes a SGDU or SGDD 44 */ 45 type record StartFluteSessionRequest { 46 UInt FluteSessionId, 47 charstring FluteSessionIp, // change 22 (WK 6): further struct 48 IPAddressType FluteSessionIpType, // change 22 (WK 6): further struct 49 UInt16 FluteSessionPort, // change 22 (WK 6): further struct 50 UInt32 FluteUpdateID, // CR (WK 34/2008): Flute Update S</pre>	=	<pre>23 24 /** 25 * 26 * @desc 27 * This group specifies information to be forwarded from TTCN-3 to 28 * the FLUTE application 29 */ 30 group fluteConfigPrimitives { 31 32 /** 33 * 34 * @desc 35 * Send to the SA to request the start of a flute session 36 * @member 37 * fluteSessionId The flute session identifier 38 * @member 39 * FDT_Instance The used FDT 40 * @member 41 * fluteSessionPayload Includes a SGDU or SGDD 42 */ 43 type record StartFluteSessionRequest { 44 UInt FluteSessionId, 45 charstring FluteSessionIp, // change 22 (WK 6): further struct 46 IPAddressType FluteSessionIpType, // change 22 (WK 6): further struct 47 UInt16 FluteSessionPort, // change 22 (WK 6): further struct 48 UInt32 FluteUpdateID, // CR (WK 34/2008): Flute Update S</pre>
<pre>51 FDT_InstanceType FDT_Instance,</pre>	< >	<pre>49 FDT_InstanceType FDT_Instance,</pre>
<pre>52 FluteSessionPayloadType FluteSessionPayload 53 } 54 55 type union FluteSessionPayloadType { 56 LibBCast_ServicePrimitives_TypesAndValues.ServiceGuideDeliveryUnitList ServiceGuide 57 ServiceGuideDeliveryDescriptorList ServiceGuideDeliveryDescriptors, 58 DvbhEsgDescriptor ESGDescriptor // change 13 (WK 6): send DVB-H bootstrap 59 } 60 61 type set of InteractivityMediaDocumentType InteractivityMediaDocumentList; 62 63 type set of ServiceGuideDeliveryUnit ServiceGuideDeliveryUnitList; 64 65 //type set length(1 .. infinity) of ServiceGuideDeliveryDescriptor ServiceGuideDeliveryDesc 66 67 68 type record FDT_InstanceType { 69 FileList Files, 70 MBMS_Session_Identity_Expiry_List MBMS_Session_Identity_Expirys optional, 71 charstring Expires,</pre>	=	<pre>50 FluteSessionPayloadType FluteSessionPayload 51 } 52 53 type union FluteSessionPayloadType { 54 LibBCast_ServicePrimitives_TypesAndValues.ServiceGuideDeliveryUnitList ServiceGuide 55 ServiceGuideDeliveryDescriptorList ServiceGuideDeliveryDescriptors, 56 DvbhEsgDescriptor ESGDescriptor // change 13 (WK 6): send DVB-H bootstrap 57 } 58 59 type set of InteractivityMediaDocumentType InteractivityMediaDocumentList; 60 61 type set of ServiceGuideDeliveryUnit ServiceGuideDeliveryUnitList; 62 63 //type set length(1 .. infinity) of ServiceGuideDeliveryDescriptor ServiceGuideDeliveryDesc 64 65 66 type record FDT_InstanceType { 67 FileList Files, 68 MBMS_Session_Identity_Expiry_List MBMS_Session_Identity_Expirys optional, 69 charstring Expires,</pre>

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

72	boolean Complete optional,	70	boolean Complete optional,
73	charstring Content_Type optional,	71	charstring Content_Type optional,
74	charstring Content_Encoding optional,	72	charstring Content_Encoding optional,
75	UInt8 FEC_OTI_FEC_Encoding_ID optional,	73	UInt8 FEC_OTI_FEC_Encoding_ID optional,
76	UInt32 FEC_OTI_FEC_Instance_ID optional,	74	UInt32 FEC_OTI_FEC_Instance_ID optional,
77	UInt32 FEC_OTI_Maximum_Source_Block_Length optional,	75	UInt32 FEC_OTI_Maximum_Source_Block_Length optional,
78	UInt32 FEC_OTI_Encoding_Symbol_Length optional,	76	UInt32 FEC_OTI_Encoding_Symbol_Length optional,
79	UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,	77	UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,
80	charstring FEC_OTI_Scheme_Specific_Info optional,	78	charstring FEC_OTI_Scheme_Specific_Info optional,
81	boolean FullFDT optional,	79	boolean FullFDT optional,
82	UInt32 Version_ID_Length optional	80	UInt32 Version_ID_Length optional
83	} with { // change 25 (WK 6): variant extension for definitions of XML namespaces	81	} with { // change 25 (WK 6): variant extension for definitions of XML namespaces
84	variant "replace '_' with '-' && namespacedef nsprefix = '' nsuri = 'urn:oma:xml:bc	82	variant "replace '_' with '-' && namespacedef nsprefix = '' nsuri = 'urn:oma:xml:bc
85	}	83	}
86		84	
87		85	
88	type set length (1 .. infinity) of FileType FileList;	86	type set length (1 .. infinity) of FileType FileList;
89		87	
90	type record FileType {	88	type record FileType {
91	MBMS_Session_Identity_List MBMS_Session_Identityys optional,	89	MBMS_Session_Identity_List MBMS_Session_Identityys optional,
92	AnyUri Content_Location,	90	AnyUri Content_Location,
93	UInt TOI,	91	UInt TOI,
94	UInt32 Content_Length optional,	92	UInt32 Content_Length optional,
95	UInt32 Transfer_Length optional,	93	UInt32 Transfer_Length optional,
96	charstring Content_Type optional,	94	charstring Content_Type optional,
97	charstring Content_Encoding optional,	95	charstring Content_Encoding optional,
98	charstring Content_MD5 optional,	96	charstring Content_MD5 optional,
99	UInt8 FEC_OTI_FEC_Encoding_ID optional,	97	UInt8 FEC_OTI_FEC_Encoding_ID optional,
100	UInt32 FEC_OTI_FEC_Instance_ID optional,	98	UInt32 FEC_OTI_FEC_Instance_ID optional,
101	UInt32 FEC_OTI_Maximum_Source_Block_Length optional,	99	UInt32 FEC_OTI_Maximum_Source_Block_Length optional,
102	UInt32 FEC_OTI_Encoding_Symbol_Length optional,	100	UInt32 FEC_OTI_Encoding_Symbol_Length optional,
103	UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,	101	UInt32 FEC_OTI_Max_Number_of_Encoding_Symbols optional,
104	charstring FEC_OTI_Scheme_Specific_Info optional,	102	charstring FEC_OTI_Scheme_Specific_Info optional,
105	UInt32 Version_ID_Length optional	103	UInt32 Version_ID_Length optional
106	} with {variant "replace '_' with '-'" }	104	} with {variant "replace '_' with '-'" }
107		105	
108	type set length (1 .. infinity) of MBMS_Session_Identity_Expiry_Type MBMS_Session_Identity_	106	type set length (1 .. infinity) of MBMS_Session_Identity_Expiry_Type MBMS_Session_Identity_
109		107	
110	type set length (1 .. infinity) of MBMS_Session_Identity_Type MBMS_Session_Identity_List;	108	type set length (1 .. infinity) of MBMS_Session_Identity_Type MBMS_Session_Identity_List;
111		109	
112	type record MBMS_Session_Identity_Expiry_Type {	110	type record MBMS_Session_Identity_Expiry_Type {
113	UInt8 text_	111	UInt8 text_
114	UInt32 value_	112	UInt32 value_
115	}	113	}
116		114	
117	type record MBMS_Session_Identity_Type {	115	type record MBMS_Session_Identity_Type {
118	UInt8 text_	116	UInt8 text_
119	}	117	}
120		118	
121	/**	119	/**
122	*	120	*
123	* @desc	121	* @desc
124	* Send by the SA to indicate that a start flute session request	122	* Send by the SA to indicate that a start flute session request
125	* could be send.	123	* could be send.
126	* @member	124	* @member
127	* returnCode 0 - success 1+ - error code	125	* returnCode 0 - success 1+ - error code
128	* @member	126	* @member
129	* reason should be set if return code > 0	127	* reason should be set if return code > 0
130	*/	128	*/
131	type record StartFluteSessionResponse {	129	type record StartFluteSessionResponse {
132	UInt returnCode,	130	UInt returnCode,
133	charstring reason optional	131	charstring reason optional
134	}	132	}
135	} with {	133	} with {
136	encode "FLUTECodec"	134	encode "FLUTECodec"
137	} // End fluteConfigPrimitives	135	} // End fluteConfigPrimitives
138		136	
139	/**	137	/**
140	*	138	*
141	* @desc	139	* @desc

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

142	* These primitives extract information from or provide it to the	140	* These primitives extract information from or provide it to the
143	* upper interface of a HTTP stack	141	* upper interface of a HTTP stack
144	*/	142	*/
145	group httpConfigPrimitives {	143	group httpConfigPrimitives {
146		144	
147	/**	145	/**
148	*	146	*
149	* @desc	147	* @desc
150	* The http request which is send by the end user to subscribe	148	* The http request which is send by the end user to subscribe
151	* to a service.	149	* to a service.
152	* @member	150	* @member
153	* connectionId Contains a unique identifier for the (TCP)	151	* connectionId Contains a unique identifier for the (TCP)
154	* connection on which the POST was received	152	* connection on which the POST was received
155	* @member	153	* @member
156	* serviceName Contains the name of the service which the end	154	* serviceName Contains the name of the service which the end
157	* user want to subscribe.	155	* user want to subscribe.
158	*/	156	*/
159	type record HttpSubscriptionRequestIndication {	157	type record HttpSubscriptionRequestIndication {
160	UInt connectionId,	158	UInt connectionId,
161	charstring serviceName	159	charstring serviceName
162	}	160	}
163		161	
164	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction	162	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction
165	/**	163	/**
166	* Note: Modifications to types still likely to cover HTTP use in	164	* Note: Modifications to types still likely to cover HTTP use in
167	* service provisioning	165	* service provisioning
168	*	166	*
169	* @desc	167	* @desc
170	* Used for terminal requests on interactive channel	168	* Used for terminal requests on interactive channel
171	* @member	169	* @member
172	* keyValuePairs Indicates more detailed SG requests	170	* keyValuePairs Indicates more detailed SG requests
173	* @remark	171	* @remark
174	* if serviceGuideRequestType and keyValuePairs are both omitted	172	* if serviceGuideRequestType and keyValuePairs are both omitted
175	* then default view to service guide shall be sent (see OMA SG	173	* then default view to service guide shall be sent (see OMA SG
176	* doc 5.4.3.2)	174	* doc 5.4.3.2)
177	*/	175	*/
178	type record HttpPostIndication {	176	type record HttpPostIndication {
179	KeyValuePairList keyValuePairs optional	177	KeyValuePairList keyValuePairs optional
180	}	178	}
181		179	
182	/**	180	/**
183	*	181	*
184	* @desc	182	* @desc
185	* Used for terminal signalling of file repair or to request	183	* Used for terminal signalling of file repair or to request
186	* fragments referenced in a SGDU	184	* fragments referenced in a SGDU
187	* @member	185	* @member
188	* connectionId Contains a unique identifier for the (TCP)	186	* connectionId Contains a unique identifier for the (TCP)
189	* connection on which the GET was received	187	* connection on which the GET was received
190	* @member	188	* @member
191	* messageUri The message URI	189	* messageUri The message URI
192	*/	190	*/
193	type record HttpGetIndication {	191	type record HttpGetIndication {
194	UInt connectionId,	192	UInt connectionId,
195	AnyUri messageUri	193	AnyUri messageUri
196	}	194	}
197		195	
198	type enumerated ServiceGuideRequestType {	196	type enumerated ServiceGuideRequestType {
199	e_sgdd,	197	e_sgdd,
200	e_sgdu,	198	e_sgdu,
201	e_sgdd_sgdu	199	e_sgdd_sgdu
202	}	200	}
203		201	
204	type record length (0 .. infinity) of KeyValuePair KeyValuePairList;	202	type record length (0 .. infinity) of KeyValuePair KeyValuePairList;
205		203	
206	type record KeyValuePair {	204	type record KeyValuePair {
207	charstring key,	205	charstring key,
208	charstring valuePart optional	206	charstring valuePart optional
209	}	207	}
210		208	
211	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction	209	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

212	/**	210	/**
213	* @desc	211	* @desc
214	* Used for terminal response on interactive channel	212	* Used for terminal response on interactive channel
215	* @member	213	* @member
216	* responseCode http status code to be used, e.g., 200 (OK)	214	* responseCode http status code to be used, e.g., 200 (OK)
217	* @member	215	* @member
218	* responseHeaders additional http header	216	* responseHeaders additional http header
219	* @member	217	* @member
220	* responsePayload This field shall only be set in case of	218	* responsePayload This field shall only be set in case of
221	* file repair responses; syntax is follows the one used in the	219	* file repair responses; syntax is follows the one used in the
222	* corresponding HTTP GET URI	220	* corresponding HTTP GET URI
223	*/	221	*/
224	type record HttpResponseRequest {	222	type record HttpResponseRequest {
225	UInt responseCode,	223	UInt responseCode,
226	ResponseHeaderList responseHeaders optional,	224	ResponseHeaderList responseHeaders optional,
227	ResponsePayload responsePayload optional	225	ResponsePayload responsePayload optional
228	}	226	}
229		227	
230	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction	228	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction
231	type record length (0 .. infinity) of ResponseHeader ResponseHeaderList;	229	type record length (0 .. infinity) of ResponseHeader ResponseHeaderList;
232		230	
233	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction	231	// change 26 (WK18): (additional to 17) clearing of HTTP request/responses for Interaction
234	type record ResponseHeader {	232	type record ResponseHeader {
235	charstring name,	233	charstring name,
236	charstring valuePart optional	234	charstring valuePart optional
237	}	235	}
238		236	
239	type union ResponsePayload {	237	type union ResponsePayload {
240	ServiceGuidePayload serviceGuidePayload,	238	ServiceGuidePayload serviceGuidePayload,
241	charstring text	239	charstring text
242	}	240	}
243		241	
244	type record ServiceGuidePayload {	242	type record ServiceGuidePayload {
245	ServiceGuideResponse serviceGuideResponse,	243	ServiceGuideResponse serviceGuideResponse,
246	ServiceGuideDeliveryUnit serviceGuideDeliveryUnit optional	244	ServiceGuideDeliveryUnit serviceGuideDeliveryUnit optional
247	}	245	}
248		246	
249	/**	247	/**
250	*	248	*
251	* @desc	249	* @desc
252	* Send by the SA to indicate that a http response could be send	250	* Send by the SA to indicate that a http response could be send
253	* or not	251	* or not
254	* @member	252	* @member
255	* returnCode 0 - success 1+ - error code	253	* returnCode 0 - success 1+ - error code
256	* @member	254	* @member
257	* reason should be set if return code > 0	255	* reason should be set if return code > 0
258	*/	256	*/
259	type record HttpResponseResponse {	257	type record HttpResponseResponse {
260	UInt returnCode,	258	UInt returnCode,
261	charstring reason optional	259	charstring reason optional
262	}	260	}
263	} with {	261	} with {
264	encode "HTTPCodec" // change 26 (WK18): (additional to 17) clearing of HTTP request/respons	262	encode "HTTPCodec" // change 26 (WK18): (additional to 17) clearing of HTTP request/respons
265	} // end httpConfigPrimitives	263	} // end httpConfigPrimitives
266		264	
267		265	
268	group smsConfigPrimitives {	266	group smsConfigPrimitives {
269	/**	267	/**
270	*	268	*
271	* @desc	269	* @desc
272	* Send by the SA to indicate that a sms was received.	270	* Send by the SA to indicate that a sms was received.
273	* @member	271	* @member
274	* messageBody The message body of the sms	272	* messageBody The message body of the sms
275	*/	273	*/
276	type record SmsGetIndication {	274	type record SmsGetIndication {
277	charstring messageBody // should be further structured	275	charstring messageBody // should be further structured
278	}	276	}
279	} // end smsConfigPrimitives	277	} // end smsConfigPrimitives
280			
281			

<>

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

282	group mmsConfigPrimitives {	=	278	group mmsConfigPrimitives {
283	/**		279	/**
284	*		280	*
285	* @desc		281	* @desc
286	* Send by the SA to indicate that a mms was received.		282	* Send by the SA to indicate that a mms was received.
287	* @member		283	* @member
288	* messageBody The message body of the mms		284	* messageBody The message body of the mms
289	*/		285	*/
290	type record MmsGetIndication {		286	type record MmsGetIndication {
291	charstring messageBody // should be further structured		287	charstring messageBody // should be further structured
292	}		288	}
293	} // end mmsConfigPrimitives		289	} // end mmsConfigPrimitives
294	/**		290	/**
295	*		291	*
296	* @desc This group specifies information to be forwarded from TTCN-3 to the streaming server		292	* @desc This group specifies information to be forwarded from TTCN-3 to the streaming server
297	*/		293	*/
298	group streamingServerPrimitives {		294	group streamingServerPrimitives {
299	/**		295	/**
300	*		296	*
301	* @desc		297	* @desc
302	* Sends start streaming request to a streaming server		298	* Sends start streaming request to a streaming server
303	* @member		299	* @member
304	* streamId The stream identifier		300	* streamId The stream identifier
305	* @member		301	* @member
306	* fileName The file name reference to a file which should be		302	* fileName The file name reference to a file which should be
307	* streamed.		303	* streamed.
308	* @member		304	* @member
309	* sdp Includes IP information, codecs, rates, unicast vs		305	* sdp Includes IP information, codecs, rates, unicast vs
310	* broadcast, etc		306	* broadcast, etc
311	* @member		307	* @member
312	* contentProtection For future use		308	* contentProtection For future use
313	* @member		309	* @member
314	* serviceProtection For future use		310	* serviceProtection For future use
315	*/		311	*/
316	type record StartStreamingRequest {		312	type record StartStreamingRequest {
317	UInt streamId,		313	UInt streamId,
318	charstring fileName,		314	charstring fileName,
319	Sdp sdp,		315	Sdp sdp,
320	charstring destinationIpAddress, // change 14 (WK18): dynamic setting of destination		316	charstring destinationIpAddress, // change 14 (WK18): dynamic setting of destination
321	ContentProtection contentProtection optional, // for future use		317	ContentProtection contentProtection optional, // for future use
322	ServiceProtection serviceProtection optional // for future use		318	ServiceProtection serviceProtection optional // for future use
323	};		319	};
324			320	
325	type charstring ContentProtection;		321	type charstring ContentProtection;
326	type charstring ServiceProtection;		322	type charstring ServiceProtection;
327			323	
328	/**		324	/**
329	*		325	*
330	* @desc		326	* @desc
331	* Send by the SA to indicate that a start streaming request		327	* Send by the SA to indicate that a start streaming request
332	* could be send.		328	* could be send.
333	* @member		329	* @member
334	* returnCode 0 - success 1+ - error code		330	* returnCode 0 - success 1+ - error code
335	* @member		331	* @member
336	* reason should be set if return code > 0		332	* reason should be set if return code > 0
337	*/		333	*/
338	type record StartStreamingResponse {		334	type record StartStreamingResponse {
339	UInt returnCode,		335	UInt returnCode,
340	charstring reason optional		336	charstring reason optional
341	}		337	}
342			338	
343	/**		339	/**
344	*		340	*
345	* @desc		341	* @desc
346	* Send to a streaming server to stop streaming.		342	* Send to a streaming server to stop streaming.
347	* @member		343	* @member
348	* streamId The stream identifier		344	* streamId The stream identifier
349	*/		345	*/
350	type record StopStreamingRequest {		346	type record StopStreamingRequest {
351	UInt streamId		347	UInt streamId

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

352	};		348	};
353			349	
354	/**		350	/**
355	*		351	*
356	* @desc		352	* @desc
357	* Send by the SA to indicate that a stop streaming request		353	* Send by the SA to indicate that a stop streaming request
358	* could be send.		354	* could be send.
359	* @member		355	* @member
360	* returnCode 0 - success 1+ - error code		356	* returnCode 0 - success 1+ - error code
361	* @member		357	* @member
362	* reason should be set if return code > 0		358	* reason should be set if return code > 0
363	*/		359	*/
364	type record StopStreamingResponse {		360	type record StopStreamingResponse {
365	UInt returnCode,		361	UInt returnCode,
366	charstring reason optional		362	charstring reason optional
367	}		363	}
368	// change 3 (WK34): Service Protection: secure streaming service	+-		
369	group secureStreamingPrimitives {			
370	type record UpdateKeyRequest {			
371	ListOfSTKMKeys keys			
372	}			
373				
374	type record of SecureStreamingKey ListOfSTKMKeys;			
375				
376	type record SecureStreamingKey {			
377	octetstring masterKey,			
378	octetstring saltKey optional			
379	}			
380				
381	type record UpdateKeyResponse {			
382	UInt returnCode,			
383	charstring reason optional			
384	}			
385				
386	type record ActivateKeyRequest {			
387	integer keyListNumber,			
388	SecureStreamingKey activeKey optional			
389	}			
390				
391	type record ActivateKeyResponse {			
392	UInt returnCode,			
393	charstring reason optional			
394	}			
395				
396	type record StartSecureStreamingRequest {			
397	charstring destinationIPAddress,			
398	Oct4 ssrc			
399	}			
400				
401	type record StartSecureStreamingResponse {			
402	UInt returnCode,			
403	charstring reason optional			
404	}			
405				
406	type record StopSecureStreamingRequest {			
407	}			
408				
409	type record StopSecureStreamingResponse {			
410	UInt returnCode,			
411	charstring reason optional			
412	}			
413	}			
414	} with { // change 26 (WK 6): support of new codec	=	364	} with { // change 26 (WK 6): support of new codec
415	encode "StreamingServerCodec"		365	encode "StreamingServerCodec"
416	} // end streamServerPrimitives		366	} // end streamServerPrimitives
417			367	
418	group fileServerPrimitives {		368	group fileServerPrimitives {
419	/**		369	/**
420	*		370	*
421	* @desc		371	* @desc

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

422	* The message wich triggers the file server to transfer files.	372	* The message wich triggers the file server to transfer files.
423	* @member	373	* @member
424	* transferId The id of the transfer	374	* transferId The id of the transfer
425	* @member	375	* @member
426	* fileList The list of file which should be transfered by the	376	* fileList The list of file which should be transfered by the
427	* file server	377	* file server
428	* @member	378	* @member
429	* sdp Includes IP information, codecs, rates, unicast vs	379	* sdp Includes IP information, codecs, rates, unicast vs
430	* broadcast, etc	380	* broadcast, etc
431	* @member	381	* @member
432	* contentProtection for futher use	382	* contentProtection for futher use
433	* @member	383	* @member
434	* serviceProtection for futher use	384	* serviceProtection for futher use
435	*/	385	*/
436	type record StartFileTransferRequest {	386	type record StartFileTransferRequest {
437	UInt transferId,	387	UInt transferId,
438	charstring FluteSessionIp, // change 20 (WK18): (additional to 11) further	388	charstring FluteSessionIp, // change 20 (WK18): (additional to 11) further
439	IPAddressType FluteSessionIpType, // change 20 (WK18): (additional to 11) further	389	IPAddressType FluteSessionIpType, // change 20 (WK18): (additional to 11) further
440	UInt16 FluteSessionPort, // change 20 (WK18): (additional to 11) further	390	UInt16 FluteSessionPort, // change 20 (WK18): (additional to 11) further
441	UInt32 FluteUpdateID, // CR (WK 34/2008): Flute Update Signalling	391	UInt32 FluteUpdateID, // CR (WK 34/2008): Flute Update Signalling
442	FDT_InstanceType FDT_Instance, // change 20 (WK18): (additional to 11) further	392	FDT_InstanceType FDT_Instance, // change 20 (WK18): (additional to 11) further
443	TransferFileList fileList,	393	TransferFileList fileList,
444	Sdp sdp,	394	Sdp sdp,
445	ContentProtection contentProtection optional,	395	ContentProtection contentProtection optional,
446	ServiceProtection serviceProtection optional // for future use	396	ServiceProtection serviceProtection optional // for future use
447	};	397	};
448		398	
449	// change 20 (WK18): (additional to 11) further structured TTCN-3 types	399	// change 20 (WK18): (additional to 11) further structured TTCN-3 types
450	/**	400	/**
451	*	401	*
452	* @desc	402	* @desc
453	* A list of Transfer files used by the file server	403	* A list of Transfer files used by the file server
454	*/	404	*/
455	type set of TransferFileWithHeaders TransferFileList;	405	type set of TransferFileWithHeaders TransferFileList;
456		406	
457	// change 20 (WK18): (additional to 11) further structured TTCN-3 types	407	// change 20 (WK18): (additional to 11) further structured TTCN-3 types
458	type record TransferFileWithHeaders {	408	type record TransferFileWithHeaders {
459	AnyUri Content_Location,	409	AnyUri Content_Location,
460	charstring Content_Type optional,	410	charstring Content_Type optional,
461	charstring Content_Encoding optional,	411	charstring Content_Encoding optional,
462	TransferFile file optional // if omit use the content-location as file reference	412	TransferFile file optional // if omit use the content-location as file reference
463	}	413	}
464		414	
465	// change 4 (WK18): according to OMA-TS-BCAST_Services 5.3.6.2	415	// change 4 (WK18): according to OMA-TS-BCAST_Services 5.3.6.2
466	/**	416	/**
467	*	417	*
468	* @desc	418	* @desc
469	* This type represents a Transfer file. A transfer file could	419	* This type represents a Transfer file. A transfer file could
470	* be a zip file or an interactivity media	420	* be a zip file or an interactivity media
471	* document.	421	* document.
472	* @member	422	* @member
473	* zipFile	423	* zipFile
474	* @member	424	* @member
475	* interactivityMediaDocument	425	* interactivityMediaDocument
476	*/	426	*/
477	type union TransferFile {	427	type union TransferFile {
478	ZipFile zipFile,	428	ZipFile zipFile,
479	InteractivityMediaDocumentType InteractivityMediaDocument	429	InteractivityMediaDocumentType InteractivityMediaDocument
480	}	430	}
481		431	
482	/**	432	/**
483	*	433	*
484	* @desc	434	* @desc
485	* This type represents a file reference.	435	* This type represents a file reference.
486	*/	436	*/
487	type charstring FileReference;	437	type charstring FileReference;
488		438	
489	/**	439	/**
490	*	440	*
491	* @desc	441	* @desc

Datei: LibBCast_ServicePrimitives_TypesAndValues.ttcn
(Fortsetzung)

492	* This type is a list of file references.	442	* This type is a list of file references.
493	*/	443	*/
494	type set of FileReference FileReferenceList;	444	type set of FileReference FileReferenceList;
495		445	
496	/**	446	/**
497	*	447	*
498	* @desc	448	* @desc
499	* This type represents a zip file. It includes all imformation	449	* This type represents a zip file. It includes all imformation
500	* for building the zip file at the test adaption.	450	* for building the zip file at the test adaption.
501	* @member	451	* @member
502	* name The name of the zip file	452	* name The name of the zip file
503	* @member	453	* @member
504	* fileList The list of file references which should be included	454	* fileList The list of file references which should be included
505	* inside the zip file	455	* inside the zip file
506	*/	456	*/
507	type record ZipFile {	457	type record ZipFile {
508	charstring name,	458	charstring name,
509	FileReferenceList fileList	459	FileReferenceList fileList
510	}	460	}
511		461	
512	/**	462	/**
513	*	463	*
514	* @desc	464	* @desc
515	* This response will be send by the file server to indicate the	465	* This response will be send by the file server to indicate the
516	* status of the file transfer start.	466	* status of the file transfer start.
517	* @member	467	* @member
518	* returnCode 0 - success, 1+ - error code	468	* returnCode 0 - success, 1+ - error code
519	* @member	469	* @member
520	* reason should be set if return code > 0	470	* reason should be set if return code > 0
521	*/	471	*/
522	type record StartFileTransferResponse {	472	type record StartFileTransferResponse {
523	UInt returnCode,	473	UInt returnCode,
524	charstring reason optional	474	charstring reason optional
525	}	475	}
526		476	
527	/**	477	/**
528	*	478	*
529	* @desc	479	* @desc
530	* Send to a file server to stop streaming.	480	* Send to a file server to stop streaming.
531	* @member	481	* @member
532	* transferId The stream identifier	482	* transferId The stream identifier
533	*/	483	*/
534	type record StopFileTransferRequest {	484	type record StopFileTransferRequest {
535	UInt transferId	485	UInt transferId
536	};	486	};
537		487	
538	/**	488	/**
539	*	489	*
540	* @desc	490	* @desc
541	* Send by the SA to indicate that a stop fiel transfer request	491	* Send by the SA to indicate that a stop fiel transfer request
542	* could be send.	492	* could be send.
543	* @member	493	* @member
544	* returnCode 0 - success 1+ - error code	494	* returnCode 0 - success 1+ - error code
545	* @member	495	* @member
546	* reason should be set if return code > 0	496	* reason should be set if return code > 0
547	*/	497	*/
548	type record StopFileTransferResponse {	498	type record StopFileTransferResponse {
549	UInt returnCode,	499	UInt returnCode,
550	charstring reason optional	500	charstring reason optional
551	}	501	}
552		502	
553	} with {	503	} with {
554	encode "FileTransferCodec" // change 27 (WK18): support of new codec	504	encode "FileTransferCodec" // change 27 (WK18): support of new codec
555	} // end fileServerPrimitives	505	} // end fileServerPrimitives
556	} // end module AtsBCast_ConfigPrimitives_TypesAndValues	506	} // end module AtsBCast_ConfigPrimitives_TypesAndValues

Datei: LibCommon_BSM.ttcn

<pre> 1 //change 1 (WK18): for content protection tests 2 module LibCommon_BSM { 3 import from LibCommon_HTTP_TypesAndValues all; 4 import from LibCommon_DataStrings all; 5 import from AtsBCast_ModuleParameters { 6 group GBA; 7 group BSM; 8 } 9 import from LibCommon_GBA_BSF all; 10 import from LibCommon_BSM_TypesAndValues all; 11 import from LibCommon_Time all; </pre>	=	<pre> 1 //change 1 (WK18): for content protection tests 2 module LibCommon_BSM { 3 import from LibCommon_HTTP_TypesAndValues all; 4 import from LibCommon_DataStrings all; 5 import from AtsBCast_ModuleParameters { 6 group GBA; 7 group BSM; 8 } 9 import from LibCommon_GBA_BSF all; 10 import from LibCommon_BSM_TypesAndValues all; 11 import from LibCommon_Time all; </pre>
<pre> 12 // change 1 (WK23): Service Request extensions for content protection tests 13 import from LibCommon_GBA_BSF_TypesAndValues all; 14 // change 1 (WK23): Service Request extensions for content protection tests 15 import from LibCommon_BasicTypesAndValues all; </pre>	+-	
<pre> 16 17 group BSM_Registration { 18 group BSM_Templates { 19 group BSM_HTTP_Templates { 20 group RequestTemplates { 21 template HttpRequest mw_registerAllServicesHttpRequest modifies mw 22 UriParameters := { 23 {Parameter := "requesttype", Value := "register"}, 24 * 25 }, 26 ContentType := { 27 // see TS 26.346 28 Name := "Content-Type", ListOfValue := {{Va 29 }, 30 Authorization := omit, 31 Body := { 32 //OMA Services 5.1.6.7 Registration Procedure 'oma- 33 mbmsSecurityRegister := mw_simpleMbmsSecurityRegistr 34 } 35 } 36 </pre>	=	<pre> 12 13 group BSM_Registration { 14 group BSM_Templates { 15 group BSM_HTTP_Templates { 16 group RequestTemplates { 17 template HttpRequest mw_registerAllServicesHttpRequest modifies mw 18 UriParameters := { 19 {Parameter := "requesttype", Value := "register"}, 20 * 21 }, 22 ContentType := { 23 // see TS 26.346 24 Name := "Content-Type", ListOfValue := {{Va 25 }, 26 Authorization := omit, 27 Body := { 28 //OMA Services 5.1.6.7 Registration Procedure 'oma- 29 mbmsSecurityRegister := mw_simpleMbmsSecurityRegistr 30 } 31 } 32 </pre>
<pre> 37 // change 1 (WK23): Service Request extensions for content protection tests 38 // MBMS User Service Registration extensions 39 template HttpRequest mw_registerAllServicesHttpRequestWithAuthorization modifies mw 40 Authorization := { 41 Value := "Digest", 42 ParameterList := { 43 {Name := "response", Value := ?}, 44 {Name := "username", Value := "" & p_username}, 45 * 46 } </pre>	<>	<pre> 33 template HttpRequest mw_registerAllServicesHttpRequestWithAuthorization modifies mw 34 UriParameters := { 35 {Parameter := "requesttype", Value := "register"}, 36 * 37 }, 38 ContentType := { 39 // see TS 26.346 40 Name := "Content-Type", ListOfValue := {{Va 41 }, 42 Body := { 43 //OMA Services 5.1.6.7 Registration Procedure 'oma- 44 mbmsSecurityRegister := mw_simpleMbmsSecurityRegistr </pre>
<pre> 47 } 48 } 49 50 template HttpRequest mw_deRegisterAllServicesHttpRequest modifies mw 51 UriParameters := { 52 {Parameter := "requesttype", Value := "deregister"}, 53 * 54 }, 55 ContentType := { 56 // see TS 26.346 57 Name := "Content-Type", ListOfValue := {{Va 58 }, 59 Authorization := omit, 60 Body := { 61 mbmsSecurityRegister := ? 62 } 63 } </pre>	=	<pre> 45 } 46 } 47 48 template HttpRequest mw_deRegisterAllServicesHttpRequest modifies mw 49 UriParameters := { 50 {Parameter := "requesttype", Value := "deregister"}, 51 * 52 }, 53 ContentType := { 54 // see TS 26.346 55 Name := "Content-Type", ListOfValue := {{Va 56 }, 57 Authorization := omit, 58 Body := { 59 mbmsSecurityRegister := ? 60 } 61 } </pre>
<pre> 64 // change 1 (WK23): Service Request extensions for content protection tests 65 template HttpRequest mw_smartcartServiceRequestHttpRequest (UInt p_groupId) modifies mw 66 ContentType := { 67 // OMA Services 5.1.2.1 68 Name := "Content-Type", ListOfValue := {{Va 69 </pre>	+-	

Datei: LibCommon_BSM.ttcn (Fortsetzung)

70	},			
71	Authorization := omit,			
72	Body := {			
73	//OMA Services 5.1.5.2.1 Service Request			
74	ServiceRequest := mw_smartcartServiceRequest(p_prot			
75	}			
76	}			
77				
78	// change 1 (WK23): Service Request extensions for content protecti			
79	template HttpRequest mw_anyServiceRequestHttpRequest modifies mw_ar			
80	ContentType := {			
81	// OMA Services 5.1.2.1			
82	Name := "Content-Type", ListOfValue := {{Va			
83	},			
84	Body := {			
85	//OMA Services 5.1.5.2.1 Service Request			
86	ServiceRequest := mw_anyServiceRequest			
87	}			
88	}			
89				
90	// change 1 (WK23): Service Request extensions for content protecti			
91	template HttpRequest mw_smartcartServiceRequestHttpRequestWithAutho			
92	Authorization := {			
93	Value := "Digest",			
94	ParameterList := {			
95	{Name := "response", Value := ?},			
96	{Name := "username", Value := "" & p_user			
97	* }			
98	}			
99	}			
100	}			
101	}	=	62	}
102			63	
103	group ResponseTemplates {		64	group ResponseTemplates {
104	template HttpResponse m_digestBsmResponse(charstring p_nonce) modif<>		65	template HttpResponse m_digestBsmResponse(integer p_statusCode, cha
105	anotherHeaders := {	=	66	anotherHeaders := {
106	{Name := "Server", ListOfValue := {m_defaultBsmServ		67	{Name := "Server", ListOfValue := {m_defaultBsmServ
107	{Name := "WWW-Authenticate", ListOfValue := {m_bsmW		68	{Name := "WWW-Authenticate", ListOfValue := {m_bsmW
108	}		69	}
109	}		70	}
110			71	
111	template HttpResponse m_successBsmResponse(integer p_statusCode, ch		72	template HttpResponse m_successBsmResponse(integer p_statusCode, ch
112	ContentType := {	<>	73	ContentType := {
113	// see TS 26.346		74	// see TS 26.346
114	Name := "Content-Type", ListOfValue := {{Value := "		75	Name := "Content-Type", ListOfValue := {{Va
115	},		76	},
116	anotherHeaders := {		77	anotherHeaders := {
117	{Name := "Server", ListOfValue := {m_defaultBsmServ		78	{Name := "Server", ListOfValue := {m_defaul
118	{Name := "Authentication-Info", ListOfValue := {m_d		79	{Name := "Authentication-Info", ListOfValue
119	},		80	},
120	Body := {		81	Body := {
121	mbmsSecurityRegisterResponse := valueof(p_mbmsSecur		82	mbmsSecurityRegisterResponse := valueof (p_
			83	}
122	}	=	84	}
123	}	+-		
124	// change 1 (WK23): Service Request extensions for content protecti			
125	template HttpResponse m_successBsmResponseCommon(charstring p_conte			
126	ContentType := {			
127	// see TS 26.346			
128	Name := "Content-Type", ListOfValue := {{Value := p			
129	},			
130	anotherHeaders := {			
131	{Name := "Server", ListOfValue := {m_defaultBsmServ			
132	{Name := "Authentication-Info", ListOfValue := {m_d			
133	},			
134	Body := p_httpBody			
135	}			
136	}	=	85	}
137			86	
138	group HttpHeaderTemplates {		87	group HttpHeaderTemplates {
139	template HttpHeaderValue m_defaultBsmServerHeader := {		88	template HttpHeaderValue m_defaultBsmServerHeader := {

Datei: LibCommon_BSM.ttcn (Fortsetzung)

140	Value := "RS ATE BSM Server/0.01",		89	Value := "RS ATE BSM Server/0.01",
141	ParameterList := omit		90	ParameterList := omit
142	}		91	}
143			92	
144	template HttpHeaderValue m_bsmWwwAuthenticate(charstring p_nonce) :		93	template HttpHeaderValue m_bsmWwwAuthenticate(charstring p_nonce) :
145	Value := "Digest",		94	Value := "Digest",
146	ParameterList := {		95	ParameterList := {
147	// see OMA BCAST SvcCntProtection 6.6		96	// see OMA BCAST SvcCntProtection 6.6
148	{Name := "realm", Value := "3GPP-bootstrapping-uicc",		97	{Name := "realm", Value := "3GPP-bootstrapping-uicc",
149	{Name := "nonce", Value := "" & p_nonce &; ""},		98	{Name := "nonce", Value := "" & p_nonce & ""},
150	{Name := "opaque", Value := "" & "1234567890abcde",		99	{Name := "opaque", Value := "" & "1234567890abcde",
151	{Name := "algorithm", Value := "MD5"},		100	{Name := "algorithm", Value := "MD5"},
152	{Name := "qop", Value := "" & "auth-int" & ""}, <>	101		{Name := "qop", Value := "" & "auth-int" & ""}
153	{Name := "stale", Value := "true"} // change 2 (WK2)			
154	}	=	102	}
155	}		103	}
156	}		104	}
157	}		105	}
158			106	
159	group BSM_XML_Templates {		107	group BSM_XML_Templates {
160	group RegisterTemplates {	<>	108	group RegisterTemplates {
161	template mbmsSecurityRegisterType mw_simpleMbmsSecurityRegister(charstring p_serviceId,	=	109	template mbmsSecurityRegisterType mw_simpleMbmsSecurityRegister(charstring p_serviceId,
162	serviceID := {p_serviceId},	<>	110	serviceID := {
			111	p_serviceId
			112	},
163	registrationRequestExtension := *	=	113	registrationRequestExtension := *
164	}		114	}
165			115	
166	template mbmsSecurityRegisterResponseType m_simpleMbmsSecurityRegisterResponse(charstring p_serviceId,		116	template mbmsSecurityRegisterResponseType m_simpleMbmsSecurityRegisterResponse(charstring p_serviceId,
167	ResponseTypes := {		117	ResponseTypes := {
168	{		118	{
169	serviceID := {p_serviceId},	<>	119	serviceID := {p_serviceId},
170	ResponseCode := {"200 OK"}		120	ResponseCode := {"200 OK"},
171	}		121	RegistrationResponseServiceExtension := omit
172	}	=	122	}
173	}	<>	123	},
174				
175	template mbmsSecurityRegisterResponseType m_testMbmsSecurityRegisterResponse(charstring p_serviceId,			
176	ResponseTypes := {			
177	{			
178	serviceID := {p_serviceId},			
179	ResponseCode := {"200 OK"}			
180	}			
181	},			
182	RegistrationResponseExtension := {	124		RegistrationResponseExtension := omit
183	version := 0, // BCAST 1.0			
184	LTKMDelivery := {Types := {{0}}} // 0 = UDP			
185	}			
186	}			
187	}			
188		125		
189	group DeRegisterTemplates {			
190	template mbmsSecurityDeregisterType mw_simpleMbmsSecurityDeregister(charstring p_serviceId,			
191	serviceID := {			
192	p_serviceId			
193	}			
194	}	=	126	}
195	}	<>		
196			127	
197	// change 1 (WK23): Service Request extensions for content protection tests		128	template mbmsSecurityRegisterResponseType m_testMbmsSecurityRegisterResponse(charstring p_serviceId,
198	group ServiceTemplates {		129	ResponseTypes := {
199	template ServiceRequestType mw_smartcardServiceRequest(UInt p_protectionKeyIds,		130	{
200	SmartcardProfileSpecificPart := {		131	serviceID := {p_serviceId},
201	ProtectionKeyIDs := {		132	ResponseCode := {"200 OK"},
202	{p_protectionKeyId},		133	RegistrationResponseServiceExtension := omit
203	*		134	}
204	}		135	},
205	}		136	RegistrationResponseExtension := {
206	}		137	version := 0, // BCAST 1.0
207			138	LTKMDelivery := {Types := {0}} // 0 = UDP
208	// change 1 (WK23): Service Request extensions for content protection tests		139	}

Datei: LibCommon_BSM.ttcn (Fortsetzung)

209	template ServiceRequestType mw_anyServiceRequest := {			
210	requestID := *,			
211	UserID := *,			
212	DeviceID := *,			
213	PurchaseItems := {*},			
214	DrmProfileSpecificPart := *,			
215	SmartcardProfileSpecificPart := *			
216				
217	}	140		}
218				
219	// change 1 (WK23): Service Request extensions for content protection			
220	template ServiceResponseType m_testServiceResponse(UInt p_requestId			
221	requestID := p_requestId			
222	}	141		}
223		142		
224	// change 1 (WK23): Service Request extensions for content protection	143		
225	template ServiceResponseType m_testServiceResponseWithoutRequestID	144		group DeRegisterTemplates {
226	requestID := omit,	145		template mbmsSecurityDeregisterType mw_simpleMbmsSecurityDeRegister(charstring p_servic
227	globalStatusCode := 0,	146		serviceID := {
228	adaptationMode := omit,	147		p_serviceId
229	PurchaseItems := p_purchaseItemList			}
230	}	148		}
231	}	149		}
232	}	=	150	}
233	}		151	}
234			152	
235	group BSM_Functions {		153	group BSM_Functions {
236	// change 1 (WK23): Service Request extensions for content protection tests	<>	154	function MBMS_User_Service_Registration_ServerSimulation(boolean p_isParallelCompor
237	function BSM_ServerSimulation(boolean p_activateUserServiceRegistration, boolean p_	155	155	// BSF generates authentication vector and calculate expected response from
238	boolean p_activateS	156	156	var bitstring v_IK := '0'B, v_CK := '0'B, v_XRES := '0'B;
239) runs on BSMServerComponent {	157	157	var Bit128 v_AUTN := ts_AuthenticationInit(v_IK, v_CK, v_XRES);
240	// variables for HTTP communication	158	158	var charstring vNonce := fx_bitstring2Base64(PX_AuthRAND & v_AUTN);
241	var HttpResponse v_response;	159	159	
		160	160	// MBMS User Service Registration Server is ready
242	var HttpRequest v_request;	=	161	var HttpRequest v_request;
243			162	
244	var octetstring v_rspAuth, v_mrk;	<>		
245	var charstring v_nonce := fx_bitstring2Base64('00000000'B);			
246				
247	// variables used for MBMS Security Register Requests/Responses			
248	var mbmsSecurityRegisterResponseType v_mbmsXml;	163	163	var mbmsSecurityRegisterResponseType v_mbmsXml;
249	// variables used for Service Requests/Responses	164	164	var octetstring rspAuth;
250	var ServiceResponseType v_serviceResponseXml;	165	165	var HttpResponse response;
251		=	166	
252	var UInt v_protectionKeyId;	<>	167	log("MBMS User Service Registration waiting...");
253				
254	if (p_activateServiceRequest) {	168	168	if (not p_isParallelComponent) {
255	v_protectionKeyId := hex2int(protectionKeyId)			
256	}			
257				
258	// MRK = KDF(Ks_ext_NAF, "mbms-mrk")			
259	v_mrk := fx_calculatingKDF4MRK(ksNAF, fx_char2octet("mbms-mrk"));			
260				
261	log("BSM ready...");			
262				
263	if ((not p_UsrActAsParallelComponent) or (not p_SrActAsParallelComponent))			
264	t_wait.start(PX_TWAIT);	=	169	t_wait.start(PX_TWAIT);
265	}		170	}
266	alt {		171	alt {
267	[p_activateUserServiceRegistration] bsm.receive(mw_registerAllServ	<>	172	[] bsm.receive(mw_registerAllServicesHttpRequest) -> value v_reques
268	log("Verdict Info: 2a. The terminal sends the Registration	=	173	log("Verdict Info: 2a. The terminal sends the Registration
269	setverdict(pass);		174	setverdict(pass);
270	bsm.send(m_digestBsmResponse(v_nonce));	<>	175	bsm.send(m_digestBsmResponse(401, vNonce));
271	alt {	=	176	alt {
272	[] bsm.receive(mw_registerAllServicesHttpRequestWith	<>	177	[] bsm.receive(mw_registerAllServicesHttpRequestWith
273	if (match(str2oct(unq(getHeaderParameterVal		178	if (match(str2oct(unq(getHeaderParameterVal
274	log("Verdict Info: 2c. The second E	=	179	log("Verdict Info: 2c. The second E
275	setverdict(pass);		180	setverdict(pass);
276	v_mbmsXml := valueof(m_testMbmsSecu	<>	181	//var mbmsSecurityRegisterResponse
			182	v_mbmsXml := valueof(m_testMbmsSecurityRegisterResponse("bcast-all-serv
277	v_rspAuth := ts_calculateRspauth(v_		183	rspAuth := ts_calculateRspauth(v_re

Datei: LibCommon_BSM.ttcn (Fortsetzung)

278	<code>v_response := m_successBsmResponse()</code>	=	184	<code>response := valueof(m_successBsmRes</code>
279	<code>bsm.send(v_response);</code>		185	<code>bsm.send(response);</code>
280	<code>}</code>		186	<code>}</code>
281	<code>else {</code>		187	<code>else {</code>
282	<code>log("Verdict Info: RES doesn't corr</code>		188	<code>log("Verdict Info: RES doesn't corr</code>
283	<code>setverdict(fail);</code>		189	<code>setverdict(fail);</code>
284	<code>bsm.send(m_basicResponse(400));</code>		190	<code>bsm.send(m_basicResponse(400));</code>
285	<code>}</code>		191	<code>}</code>
286	<code>}</code>		192	<code>}</code>
287	<code>[] bsm.receive(mw_anyHttpRequest) {</code>		193	<code>[] bsm.receive(mw_anyHttpRequest) {</code>
288	<code>log("Verdict Info: The terminal doesn't ser</code>		194	<code>log("Verdict Info: The terminal doesn't ser</code>
289	<code>setverdict(fail);</code>		195	<code>setverdict(fail);</code>
290	<code>bsm.send(m_basicResponse(400));</code>		196	<code>bsm.send(m_basicResponse(400));</code>
291	<code>}</code>		197	<code>}</code>
292	<code>[not p_UsrActAsParallelComponent] t wait.timeout {</code>	<>	198	<code>[not p_isParallelComponent] t wait.timeout {</code>
293	<code>log("Verdict Info: The terminal doesn't ser</code>	=	199	<code>log("Verdict Info: The terminal doesn't ser</code>
294	<code>setverdict(inconc);</code>		200	<code>setverdict(inconc);</code>
295	<code>}</code>		201	<code>}</code>
296	<code>} // inner alt</code>		202	<code>} // inner alt</code>
297			203	
298	<code>if (p_UsrActAsParallelComponent) {</code>	<>	204	<code>if (p_isParallelComponent) {</code>
299	<code>repeat;</code>		205	<code>repeat;</code>
300	<code>}</code>			
301	<code>}</code>			
302	<code>[p_activateServiceRequest] bsm.receive(mw_anyServiceRequestHttpRequ</code>			
303	<code>log("Verdict Info: 2a. The terminal sends the Service Request for t</code>			
304	<code>setverdict(pass);</code>			
305	<code>bsm.send(m_digestBsmResponse(v_nonce));</code>			
306	<code>alt {</code>			
307	<code> [] bsm.receive(mw_anyServiceRequestHttpRequest) -></code>			
308	<code> if (match(str2oct(unq(getHeaderParameterVal</code>			
309	<code> log("Verdict Info: 2c. The second E</code>			
310	<code>setverdict(pass);</code>			
311	<code>var PurchaseItemList requestPurchas</code>			
312	<code>var PurchaseItemResponseList respon</code>			
313	<code>for (var integer i := 0; i < sizeof</code>			
314	<code> responsePurchaseItems[i] :=</code>			
315	<code>}</code>			
316				
317	<code>if (ispresent(v_request.Body.Servic</code>			
318	<code> v_serviceResponseXml := val</code>			
319	<code>}</code>			
320	<code>else {</code>			
321	<code> v_serviceResponseXml := val</code>			
322	<code>}</code>			
323				
324	<code>v_rspAuth := ts_calculateRspauth(v_</code>			
325	<code>v_response := m_successBsmResponse()</code>			
326	<code>bsm.send(v_response);</code>			
327	<code>}</code>			
328	<code>else {</code>			
329	<code>log("Verdict Info: RES doesn't corr</code>			
330	<code>setverdict(fail);</code>			
331	<code>bsm.send(m_basicResponse(400));</code>			
332	<code>}</code>			
333	<code>}</code>			
334	<code>[] bsm.receive(mw_anyHttpRequest) {</code>			
335	<code>log("Verdict Info: The terminal doesn't ser</code>			
336	<code>setverdict(fail);</code>			
337	<code>bsm.send(m_basicResponse(400));</code>			
338	<code>}</code>			
339	<code>[not p_SrActAsParallelComponent] t wait.timeout {</code>			
340	<code>log("Verdict Info: The terminal doesn't ser</code>			
341	<code>setverdict(inconc);</code>			
342	<code>}</code>			
343	<code>} // inner alt</code>		206	<code>}</code>
344				
345	<code>if (p_SrActAsParallelComponent) {</code>			
346	<code>repeat;</code>			
347	<code>}</code>			
348	<code>}</code>	=	207	<code>}</code>

Datei: LibCommon_BSM.ttcn (Fortsetzung)

349	[] bsm.receive(mw_anyHttpRequest) {		208	[] bsm.receive(mw_anyHttpRequest) {
350	log("Verdict Info: The terminal doesn't send a HTTP request		209	log("Verdict Info: The terminal doesn't send a HTTP request
351	setverdict(fail);		210	setverdict(fail);
352	bsm.send(m_basicResponse(400));		211	bsm.send(m_basicResponse(400));
353			212	
354	if (p_UsrActAsParallelComponent or p_SrActAsParallelComponen	<>	213	if (p_isParallelComponent) {
355	repeat;		214	repeat;
356	}	=	215	}
357	}		216	}
358	[(not p_UsrActAsParallelComponent) or (not p_SrActAsParallelComponen	<>	217	[not p_isParallelComponent] t_wait.timeout {
359	log("Verdict Info: The terminal doesn't send a HTTP request		218	log("Verdict Info: The terminal doesn't send a HTTP request
360	setverdict(fail);		219	setverdict(inconc);
361	}	=	220	}
362	} // alt		221	} // alt
363			222	
364	return;		223	return;
365	}		224	}
366			225	
367	// change 2 (WK23): GBA authentication extensions for content protection test	<>	226	function ts_AuthResponseCheckMD5 (in HttpRequest p_httpRequest, in bitstring p_XRES
368	function ts_calculateAuthResponse(in HttpRequest p_httpRequest, in octetstring p_pa		227	var charstring v_Method := p_httpRequest.Method;
369	var ListOfParameter paramList := p_httpRequest.Authorization.ParameterList;	=	228	var ListOfParameter paramList := p_httpRequest.Authorization.ParameterList;
370	var charstring v_Username := getHeaderParameterValue("username", paramList);		229	var charstring v_Username := getHeaderParameterValue("username", paramList);
371	var octetstring v_oUsername := fx_char2octet(unq(v_Username));	+-		
372	var charstring v_Realm := getHeaderParameterValue("realm", paramList);	=	230	var charstring v_Realm := getHeaderParameterValue("realm", paramList);
373	var octetstring v_oRealm := fx_char2octet(unq(v_Realm));	<>	231	var charstring v_DigestUri := getHeaderParameterValue("uri", paramList);
374	var charstring v_Method := p_httpRequest.Method;		232	var charstring v_Nonce := getHeaderParameterValue("nonce", paramList);
			233	var charstring v_NC := getHeaderParameterValue("nc", paramList);
			234	var charstring v_CNonce := getHeaderParameterValue("cnonce", paramList);
			235	var charstring v_Qop := getHeaderParameterValue("qop", paramList);
			236	var charstring v_Dresponse := getHeaderParameterValue("response", paramList
			237	
			238	
			239	
375	var octetstring v_oMETHOD := fx_char2octet(v_Method);	=	240	var octetstring v_oMETHOD := fx_char2octet(v_Method);
376	var charstring v_DigestUri := getHeaderParameterValue("uri", paramList);	<>	241	var octetstring v_oUsername := fx_char2octet(unq(v_Username));
			242	var octetstring v_opasswd := bit2oct(p_XRES);
			243	
			244	
			245	var octetstring v_oRealm := fx_char2octet(unq(v_Realm));
377	var octetstring v_oDigestUri := fx_char2octet(v_DigestUri);	=	246	var octetstring v_oDigestUri := fx_char2octet(v_DigestUri);
378	var charstring v_Nonce := getHeaderParameterValue("nonce", paramList);	+-		
379	var octetstring v_oNonce := fx_char2octet(unq(v_Nonce));	=	247	var octetstring v_oNonce := fx_char2octet(unq(v_Nonce));
380	var charstring v_NC := getHeaderParameterValue("nc", paramList);	+-		
381	var octetstring v_oNC := fx_char2octet(v_NC);	=	248	var octetstring v_oNC := fx_char2octet(v_NC);
382	var charstring v_CNonce := getHeaderParameterValue("cnonce", paramList);	+-		
383	var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));	=	249	var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));
384	var charstring v_Qop := getHeaderParameterValue("qop", paramList);	+-		
385	var octetstring v_oQop := fx_char2octet(v_Qop);	=	250	var octetstring v_oQop := fx_char2octet(v_Qop);
		+-	251	var octetstring v_oDresponse := str2oct(unq(v_Dresponse));
386		=	252	
		<>	253	var octetstring v_Secret := '0;
			254	var octetstring v_HA2 := '0;
			255	var octetstring v_Data := '0;
			256	var octetstring v_RequestDigest := '0;
			257	
387	// RFC 2617 3.2.2.2	=	258	// RFC 2617 3.2.2.2
388	// A1 = unq(username-value) ":" unq(realm-value) ":" passwd		259	// A1 = unq(username-value) ":" unq(realm-value) ":" passwd
389	var octetstring A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & p_pa	<>	260	var octetstring A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & v_o
			261	
			262	
390	// H(A1)	=	263	// H(A1)
391	var octetstring HA1 := fx_md5_hex(A1);	<>	264	var octetstring HA1 := fx_md5_hex (A1);
			265	
392	var charstring cHA1 := oct2str(HA1);	=	266	var charstring cHA1 := oct2str(HA1);
393	var octetstring v_RequestDigest, A2, secret_data, v_Secret, v_HA2, v_Data;	<>	267	
			268	var octetstring A2, secret_data;
394		=	269	
395	v_Secret := fx_charLow2octet(cHA1);		270	v_Secret := fx_charLow2octet(cHA1);
396			271	
397	// RFC 2617 3.2.2.3		272	// RFC 2617 3.2.2.3
398	// A2 = Method ":" digest-uri-value		273	// A2 = Method ":" digest-uri-value

Datei: LibCommon_BSM.ttcn (Fortsetzung)

399	A2 := v_oMETHOD & COLON_HEX & v_oDigestUri;		274	A2 := v_oMETHOD & COLON_HEX & v_oDigestUri;
400			275	
401	v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));		276	v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));
402			277	
403	// RFC 2617 3.2.2.1		278	// RFC 2617 3.2.2.1
404	// unq(nonce-value) ":" nc-value ":" unq(cnonce-value) ":" unq(qop-value) "		279	// unq(nonce-value) ":" nc-value ":" unq(cnonce-value) ":" unq(qop-value) "
405	v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce & COLON_HEX		280	v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce & COLON_HEX
406			281	
407	// now apply KD function described in RFC 2617 section 3.2.1		282	// now apply KD function described in RFC 2617 section 3.2.1
408	// KD(secret, data) = H(concat(secret, ":", data))		283	// KD(secret, data) = H(concat(secret, ":", data))
409	secret_data := v_Secret & COLON_HEX & v_Data;		284	secret_data := v_Secret & COLON_HEX & v_Data;
410			285	
411	v_RequestDigest := fx_md5_hex(secret_data);		286	v_RequestDigest := fx_md5_hex(secret_data);
412			287	
413	log("ts_calculateAuthResponse(): METHOD '" & v_Method & "'");	<>	288	
414	log("ts_calculateAuthResponse(): client username '" & v_Username & "'");		289	log("ts_AuthResponseCheckMD5(): METHOD '" & v_Method & "'");
415	log("ts_calculateAuthResponse(): client realm '" & v_Realm & "'");		290	log("ts_AuthResponseCheckMD5(): client username '" & v_Username & "'");
416	log("ts_calculateAuthResponse(): client URI '" & v_DigestUri & "'");		291	log("ts_AuthResponseCheckMD5(): client realm '" & v_Realm & "'");
417	log("ts_calculateAuthResponse(): client nonce '" & v_Nonce & "'");		292	log("ts_AuthResponseCheckMD5(): client URI '" & v_DigestUri & "'");
418	log("ts_calculateAuthResponse(): client nc '" & v_NC & "'");		293	log("ts_AuthResponseCheckMD5(): client nonce '" & v_Nonce & "'");
419	log("ts_calculateAuthResponse(): client cnonce '" & v_CNonce & "'");		294	log("ts_AuthResponseCheckMD5(): client nc '" & v_NC & "'");
420	log("ts_calculateAuthResponse(): client QoP '" & v_QoP & "'");		295	log("ts_AuthResponseCheckMD5(): client CNonce '" & v_CNonce & "'");
			296	log("ts_AuthResponseCheckMD5(): client QoP '" & v_QoP & "'");
			297	log("ts_AuthResponseCheckMD5(): client response value '" & v_Dresponse & "'");
			298	
			299	log("ts_AuthResponseCheckMD5(): server XRES: " & oct2str(v_opasswd));
			300	
			301	log("ts_AuthResponseCheckMD5(): A1 = unq(username-value) ":" unq(realm-value) ":" passv
			302	log("ts_AuthResponseCheckMD5(): A1: " & oct2str(A1));
			303	
			304	log("ts_AuthResponseCheckMD5(): H(A1): " & oct2str(HA1));
			305	log("ts_AuthResponseCheckMD5(): cH(A1): " & cHA1);
			306	log("ts_AuthResponseCheckMD5(): v_Secret (H(A1)): " & oct2str(v_Secret));
			307	
			308	log("ts_AuthResponseCheckMD5(): A2 = Method ":" digest-uri-value");
			309	log("ts_AuthResponseCheckMD5(): A2: " & oct2str(A2));
			310	
			311	log("ts_AuthResponseCheckMD5(): v_HA2: " & oct2str(v_HA2));
			312	
			313	log("ts_AuthResponseCheckMD5(): data: " & oct2str(v_Data));
			314	
			315	log("ts_AuthResponseCheckMD5(): concat(secret, ":", data));
			316	log("ts_AuthResponseCheckMD5(): secret_data: " & oct2str(secret_data));
			317	
			318	log("ts_AuthResponseCheckMD5(): KD(secret, data) = H(concat(secret, ":", data));
			319	log("ts_AuthResponseCheckMD5(): v_RequestDigest: " & oct2str(v_RequestDigest));
			320	
			321	if(v_oDresponse == v_RequestDigest) {
			322	setverdict(pass);
			323	} else {
			324	log("Verdict Info: HTTP Digest response value is wrong.");
			325	setverdict(fail);
			326	}
421			327	
422	return v_RequestDigest;	=	327	return v_RequestDigest;
423	}		328	}
424			329	
425		<>	330	function ts_calculateRspauth(in HttpRequest p_request, in bitstring p_XRES, in char
426	// change 2 (WK23): GBA authentication extensions for content protection test			
427	function ts_calculateRspauth(in HttpRequest p_request, in octetstring p_password, i			
428	// RFC 2617 and 3310	=	331	// RFC 2617 and 3310
429	var ListOfParameter p_list := p_request.Authorization.ParameterList;		332	var ListOfParameter p_list := p_request.Authorization.ParameterList;
430	var charstring v_NC := getHeaderParameterValue("nc", p_list);		333	var charstring v_NC := getHeaderParameterValue("nc", p_list);
431	var charstring v_CNonce := getHeaderParameterValue("cnonce", p_list);		334	var charstring v_CNonce := getHeaderParameterValue("cnonce", p_list);
432	var charstring v_DigestUri := getHeaderParameterValue("uri", p_list);		335	var charstring v_DigestUri := getHeaderParameterValue("uri", p_list);
433			336	
434	// username is BTID		337	// username is BTID
435	var octetstring v_oUsername := fx_char2octet(btId);	<>	338	var octetstring v_oUsername := fx_char2octet(f_generateBtid());
			339	var octetstring v_opasswd := bit2oct(p_XRES);
			340	// see OMA BCAS1 SvcCntProtection 6.6
			341	var octetstring v_oRealm := fx_char2octet("3GPP-bootstrapping-uicc@" & PX_F
436	// complete URI (e.g. "/bmsc.home1.net/keymanagement?requesttype=register")	=	342	// complete URI (e.g. "/bmsc.home1.net/keymanagement?requesttype=register")

Datei: LibCommon_BSM.ttcn (Fortsetzung)

437	var octetstring v_oDigestUri := fx_char2octet(v_DigestUri);		343	var octetstring v_oDigestUri := fx_char2octet(v_DigestUri);
438	var octetstring v_oNonce := fx_char2octet(p_nonce);		344	var octetstring v_oNonce := fx_char2octet(p_nonce);
439	var octetstring v_oNC := fx_char2octet(v_NC);		345	var octetstring v_oNC := fx_char2octet(v_NC);
440	var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));		346	var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));
441	var octetstring v_oQop := fx_char2octet("auth-int");		347	var octetstring v_oQop := fx_char2octet("auth-int");
442			348	
443	var octetstring A1, v_oRealm, v_Secret, v_HA2, v_Data, v_RequestDigest, h_e<>	349	var octetstring v_Secret := 'O;	
444		350	var octetstring v_HA2 := 'O;	
445	// see OMA BCAST SvcCntProtection 6.6	351	var octetstring v_Data := 'O;	
446	if (PX_GBA == e_gba_me) {	352	var octetstring v_RequestDigest := 'O;	
447	v_oRealm := fx_char2octet("3GPP-bootstrapping@" & PX_BSM_FQDN);			
448	}			
449	else {			
450	v_oRealm := fx_char2octet("3GPP-bootstrapping-uicc@" & PX_BSM_FQDN)			
451	}			
452		=	353	
453	// RFC 2617 3.2.2.2		354	
454	// A1 = unq(username-value) ":" unq(realm-value) ":" passwd		355	// RFC 2617 3.2.2.2
455	A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & p_password;	<>	356	// A1 = unq(username-value) ":" unq(realm-value) ":" passwd
			357	var octetstring A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & v_op
			358	
			359	var octetstring h_entityBody, A2, secret_data;
456		=	360	
457	// H(A1)		361	// H(A1)
458	v_Secret := fx_charLow2octet(oct2str(fx_md5_hex(A1)));		362	v_Secret := fx_charLow2octet(oct2str(fx_md5_hex(A1)));
459			363	
460	// RFC 2617 3.2.3		364	// RFC 2617 3.2.3
461	// A2 = ":" digest-uri-value ":" H(entity-body)		365	// A2 = ":" digest-uri-value ":" H(entity-body)
462	h_entityBody := fx_charLow2octet(oct2str(fx_md5_hex(p_entityBody)));		366	h_entityBody := fx_charLow2octet(oct2str(fx_md5_hex(p_entityBody)));
463	A2 := COLON_HEX & v_oDigestUri & COLON_HEX & h_entityBody;		367	A2 := COLON_HEX & v_oDigestUri & COLON_HEX & h_entityBody;
464			368	
465	v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));		369	v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));
466			370	
467	// RFC 2617 3.2.2.1		371	// RFC 2617 3.2.2.1
468	// unq(nonce-value) ":" nc-value ":" unq(cnonce-value) ":" unq(qop-value) "		372	// unq(nonce-value) ":" nc-value ":" unq(cnonce-value) ":" unq(qop-value) "
469	v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce & COLON_HEX		373	v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce & COLON_HEX
470			374	
471	//now apply KD function described in RFC 2617 section 3.2.1		375	//now apply KD function described in RFC 2617 section 3.2.1
472	//KD(secret, data) = H(concat(secret, ":", data))		376	//KD(secret, data) = H(concat(secret, ":", data))
473	secret_data := v_Secret & COLON_HEX & v_Data;	<>	377	secret_data := v_Secret & COLON_HEX & v_Data;
474		=	378	
475	log("ts_calculateRspauth(): v_NC -> " & v_NC);	<>	379	log("ts_calculateRspauth(): v_NC -> " & v_NC);
476	log("ts_calculateRspauth(): v_CNonce -> " & v_CNonce);		380	log("ts_calculateRspauth(): v_CNonce -> " & v_CNonce);
477	log("ts_calculateRspauth(): v_oUsername -> " & oct2str(v_oUsername));		381	log("ts_calculateRspauth(): v_oUsername -> " & oct2str(v_oUsername));
478			382	log("ts_calculateRspauth(): v_opasswd -> " & oct2str(v_opasswd));
479	log("ts_calculateRspauth(): v_oRealm -> " & oct2str(v_oRealm));		383	log("ts_calculateRspauth(): v_oRealm -> " & oct2str(v_oRealm));
480	log("ts_calculateRspauth(): v_DigestUri -> " & v_DigestUri);		384	log("ts_calculateRspauth(): v_DigestUri -> " & v_DigestUri);
481	log("ts_calculateRspauth(): v_oNonce/p_nonce -> " & p_nonce);		385	log("ts_calculateRspauth(): v_oNonce/p_nonce -> " & p_nonce);
482	log("ts_calculateRspauth(): v_oNC/v_NC -> " & v_NC);		386	log("ts_calculateRspauth(): v_oNC/v_NC -> " & v_NC);
483	log("ts_calculateRspauth(): v_oCNonce -> " & v_CNonce);		387	log("ts_calculateRspauth(): v_oCNonce -> " & v_CNonce);
484	log("ts_calculateRspauth(): v_oQop -> " & oct2str(v_oQop));		388	log("ts_calculateRspauth(): v_oQop -> " & oct2str(v_oQop));
485		=	389	
486	return fx_md5_hex(secret_data);		390	return fx_md5_hex(secret_data);
487		+ -		
488	// change 2 (WK23): GBA authentication/authorization extensions for content protect			
489	function f_bsm_setProtectionKeyId(hexstring p_protectionKeyId) runs on BSMServerCon			
490	protectionKeyId := p_protectionKeyId;			
491	}			
492				
493	// change 2 (WK23): GBA authentication/authorization extensions for content protect			
494	function f_bsm_setBtid(charstring p_btid) runs on BSMServerComponent {			
495	btid := p_btid;			
496	}			
497				
498	// change 2 (WK23): GBA authentication/authorization extensions for content protect			
499	function f_bsm_setProtectionKeysAndValues(octetstring p_ksNAF) runs on BSMServerCon			
500	ksNAF := p_ksNAF;			
501	}			
502	}	=	391	}
503			392	
504	group BSM_Configuration_and_Components {		393	group BSM_Configuration_and_Components {

Datei: LibCommon_BSM.ttcn (Fortsetzung)

505	type component BSMServerComponent {		394	type component BSMServerComponent {
506	port BSMPort bsm;		395	port BSMPort bsm;
507			396	
508	timer t_wait;		397	timer t_wait;
509		+-		
510	var hexstring protectionKeyId; // change 2 (WK23): GBA authentication/autho			
511	var charstring btid; // change 2 (WK23): GBA authentication/authorization e			
512	var octetstring ksNAF // change 2 (WK23): GBA authentication/authorization			
513	}	=	398	}
514			399	
515	type port BSMPort message {		400	type port BSMPort message {
516	in HttpRequest;		401	in HttpRequest;
517	out HttpResponse		402	out HttpResponse
518	}		403	}
519		+-		
520	}	=	404	}
521			405	
522	group externalFunctions {		406	group externalFunctions {
523	external function fx_mbmsXml2Oct(in mbmsSecurityRegisterResponseType p_mbmsXml) ret		407	external function fx_mbmsXml2Oct(in mbmsSecurityRegisterResponseType p_mbmsXml) ret
524	// change 1 (WK23): Service Request extensions for content protection tests	+-		
525	external function fx_serviceXml2Oct(in ServiceResponseType p_serviceXml) return oct			
526	// change 1 (WK23): Service Request extensions for content protection tests			
527	external function fx_calculatingKDF4MRK(in octetstring p_keys, in octetstring p_kdf			
528	}	=	408	}
529	}		409	}
530	}		410	}

Datei: LibCommon_BSM_TypesAndValues.ttcn

1	//change 1 (WK18): for content protection tests	=	1	//change 1 (WK18): for content protection tests
2	module LibCommon_BSM_TypesAndValues {		2	module LibCommon_BSM_TypesAndValues {
3	import from LibCommon_BasicTypesAndValues all;		3	import from LibCommon_BasicTypesAndValues all;
4	// change 1 (WK23): Service Request extensions for content protection tests	+-		
5	import from LibBCast_Common_TypesAndValues all;			
6	import from LibBCast_ServiceGuide_TypesAndValues {	=	4	import from LibBCast_ServiceGuide_TypesAndValues {
7	group commonTypes;		5	group commonTypes;
8	}		6	}
9			7	
10	// according to scheme for Service Protection Registration (see TS 26.346 11.4.1)		8	// according to scheme for Service Protection Registration (see TS 26.346 11.4.1)
11	group ServiceProtectionRegistration_TypeDefinition {		9	group ServiceProtectionRegistration_TypeDefinition {
12	type record mbmsSecurityRegisterType {		10	type record mbmsSecurityRegisterType {
13	serviceIDType serviceID,		11	serviceIDType serviceID,
14	RegistrationRequestExtensionType registrationRequestExtension optional		12	RegistrationRequestExtensionType registrationRequestExtension optional
15	} with {		13	} with {
16	variant "namespacedef nsprefix = '' nsuri = 'urn:3GPP:metadata:2005:MBMS:securityRe		14	variant "namespacedef nsprefix = '' nsuri = 'urn:3GPP:metadata:2005:MBMS:securityRe
17	}		15	}
18			16	
19	type record serviceIDType {		17	type record serviceIDType {
20	XmlElementAnyURIValue text_		18	XmlElementAnyURIValue text_
21	}		19	}
22			20	
23	// according to scheme OMA-TS-BCAST_Services (5.1.6.7.1)		21	// according to scheme OMA-TS-BCAST_Services (5.1.6.7.1)
24	group BcastRegistrationRequestExtension_TypeDefinition {		22	group BcastRegistrationRequestExtension_TypeDefinition {
25	type record RegistrationRequestExtensionType {		23	type record RegistrationRequestExtensionType {
26	UInt version,		24	UInt version,
27	LTKMDeliveryType LTKMDelivery optional		25	LTKMDeliveryType LTKMDelivery optional
28	} with {		26	} with {
29	variant "namespace prefix = 'bcast' namespacedef nsprefix = 'bcast' nsuri =		27	variant "namespace prefix = 'bcast' namespacedef nsprefix = 'bcast' nsuri =
30	}		28	}
31			29	
32	type record LTKMDeliveryType {		30	type record LTKMDeliveryType {
33	TypeList Types		31	TypeList Types
34	} with {		32	} with {
35	variant "namespace prefix = 'bcast'"		33	variant "namespace prefix = 'bcast'"
36	}		34	}
37			35	
38	type set length (1 .. infinity) of TypeType TypeList	<>	36	type set length (1 .. infinity) of XmlElementUnsignedByteValue TypeList
39	with {	=	37	with {
40	variant "namespace prefix = 'bcast'"		38	variant "namespace prefix = 'bcast'"
41	}	<>		
42				
43	// change 1 (WK23): Service Request extensions for content protection tests			
44	// MBMS User Service Registration extensions			
45	type record TypeType {			
46	XmlElementUnsignedByteValue text_			
47	}	=	39	}
48	}		40	}
49	}		41	}
50			42	
51	// according to scheme for Service Protection De-Registration (see TS 26.346 11.5.1)		43	// according to scheme for Service Protection De-Registration (see TS 26.346 11.5.1)
52	group ServiceProtectionDeRegistration_TypeDefinition {		44	group ServiceProtectionDeRegistration_TypeDefinition {
53	type record mbmsSecurityDeregisterType {		45	type record mbmsSecurityDeregisterType {
54	serviceIDType serviceID		46	serviceIDType serviceID
55	} with {		47	} with {
56	variant "namespacedef nsprefix = '' nsuri = 'urn:3GPP:metadata:2005:MBMS:securityDe		48	variant "namespacedef nsprefix = '' nsuri = 'urn:3GPP:metadata:2005:MBMS:securityDe
57	}		49	}
58	}		50	}
59			51	
60	// according to scheme for Service Protection Registration Response (see TS 26.346 11.7.1)		52	// according to scheme for Service Protection Registration Response (see TS 26.346 11.7.1)
61	// used as Service Protection Registration AND De-Registration Response Format		53	// used as Service Protection Registration AND De-Registration Response Format
62	group ServiceProtectionRegistrationResponse_TypeDefinition {		54	group ServiceProtectionRegistrationResponse_TypeDefinition {
63	type record mbmsSecurityRegisterResponseType {		55	type record mbmsSecurityRegisterResponseType {
64	ResponseTypeList ResponseTypes,		56	ResponseTypeList ResponseTypes,
65	RegistrationResponseExtensionType RegistrationResponseExtension optional		57	RegistrationResponseExtensionType RegistrationResponseExtension optional
66	} with {		58	} with {
67	variant "namespacedef nsprefix = '' nsuri = 'urn:3GPP:metadata:2005:MBMS:securityRe		59	variant "namespacedef nsprefix = '' nsuri = 'urn:3GPP:metadata:2005:MBMS:securityRe
68	}		60	}
69			61	
70	type set of ResponseType ResponseTypeList;		62	type set of ResponseType ResponseTypeList;
71			63	

Datei: LibCommon_BSM_TypesAndValues.ttcn (Fortsetzung)

72	type record ResponseType {		64	type record ResponseType {
73	serviceIDType serviceID,		65	serviceIDType serviceID,
74	ResponseCodeType ResponseCode,		66	ResponseCodeType ResponseCode,
75	RegistrationResponseServiceExtensionType RegistrationResponseServiceExtension optional		67	RegistrationResponseServiceExtensionType RegistrationResponseServiceExtension optional
76	}		68	}
77			69	
78	type record ResponseCodeType {		70	type record ResponseCodeType {
79	XmlElementStringValue text_		71	XmlElementStringValue text_
80	}		72	}
81			73	
82	// according to scheme OMA-TS-BCAST_Services (5.1.6.7.2 and 5.1.6.7.3)		74	// according to scheme OMA-TS-BCAST_Services (5.1.6.7.2 and 5.1.6.7.3)
83	group BcastRegistrationResponseExtension_TypeDefinition {		75	group BcastRegistrationResponseExtension_TypeDefinition {
84	type record RegistrationResponseExtensionType {		76	type record RegistrationResponseExtensionType {
85	UInt version,		77	UInt version,
86	LTKMDeliveryType LTKMDelivery optional		78	LTKMDeliveryType LTKMDelivery optional
87	} with {		79	} with {
88	variant "namespace prefix = 'bcast'"		80	variant "namespace prefix = 'bcast'"
89	}		81	}
90			82	
91	type record RegistrationResponseServiceExtensionType {		83	type record RegistrationResponseServiceExtensionType {
92	UInt version,		84	UInt version,
93	LTKMLList LTKMs optional,		85	LTKMLList LTKMs optional,
94	SubscriptionWindowType SubscriptionWindow optional		86	SubscriptionWindowType SubscriptionWindow optional
95	} with {		87	} with {
96	variant "namespace prefix = 'bcast'"		88	variant "namespace prefix = 'bcast'"
97	}		89	}
98			90	
99	type set length (1 .. infinity) of LTKMType LTKMLList	<>	91	type set length (1 .. infinity) of XmlElementStringValue LTKMLList
100	with {	=	92	with {
101	variant "namespace prefix = 'bcast'"		93	variant "namespace prefix = 'bcast'"
102	}	+-		
103				
104	// change 1 (WK23): Service Request extensions for content protection tests			
105	// MBMS User Service Registration extensions			
106	type record LTKMType {			
107	XmlElementStringValue text_			
108	}	=	94	}
109			95	
110	type record SubscriptionWindowType {		96	type record SubscriptionWindowType {
111	UInt startTime,		97	UInt startTime,
112	UInt endTime optional		98	UInt endTime optional
113	} with {		99	} with {
114	variant "namespace prefix = 'bcast'"		100	variant "namespace prefix = 'bcast'"
115	}		101	}
116	}		102	}
117	}		103	}
118		<>		
119	// change 1 (WK23): Service Request extensions for content protection tests			
120	// according to scheme for Service Request (see OMA-TS-BaCST_Services 5.1.5.2.1)			
121	group ServiceRequest_TypeDefinition {			
122	type record ServiceRequestType {			
123	UInt requestID optional,			
124	UserIDType UserID optional, // for smartcard is omitted			
125	DeviceIDType DeviceID optional,			
126	ServiceEncryptionProtocolList ServiceEncryptionProtocols optional,			
127	PurchaseItemList PurchaseItems,			
128	DrmProfileSpecificPartType DrmProfileSpecificPart optional,			
129	SmartcardProfileSpecificPartType SmartcardProfileSpecificPart optional			
130	}			
131				
132	type set length (1 .. infinity) of PurchaseItemType PurchaseItemList;			
133				
134	type set length (1 .. infinity) of ServiceEncryptionProtocolType ServiceEncryptionProtocols;			
135				
136	type record ServiceEncryptionProtocolType {			
137	XmlElementStringValue text_			
138	}			
139				
140	type record PurchaseItemType {			
141	AnyUri globalIDRef,			
142	PurchaseDataReferenceType PurchaseDataReference optional,			

Datei: LibCommon_BSM_TypesAndValues.ttcn (Fortsetzung)

143	UserConsentAnswerType UserConsentAnswer optional,	
144	ServiceType Service optional	
145	}	
146		
147	type record PurchaseDataReferenceType {	
148	AnyUri idRef,	
149	PriceType Price optional	
150	}	
151		
152	type record PriceType {	
153	charstring currency optional,	
154	XmlElementStringValue text_	
155	}	
156		
157	type record UserConsentAnswerType {	
158	AnyUri id,	
159	XmlElementStringValue text_ optional // boolean > 'true' 'false' '0' '1'	
160	}	
161		
162	type record ServiceType {	
163	AnyUri globalIDRef,	
164	charstring notification // boolean > 'true' 'false' '0' '1'	
165	}	
166		
167	type record SmartcardProfileSpecificPartType {	
168	charstring timestampMin optional,	
169	charstring timestampMax optional,	
170	ProtectionKeyIDList ProtectionKeyIDs	
171	}	
172		
173	type set length (1 .. infinity) of ProtectionKeyIDType ProtectionKeyIDList;	
174		
175	type record ProtectionKeyIDType {	
176	XmlElementUnsignedIntValue text_	
177	}	
178		
179	type record DeviceIDType {	
180	UInt8 type_	
181	}	
182		
183	type record DrmProfileSpecificPartType {	
184	AnyUri rightsIssuerURI optional,	
185	BroadcastModeType BroadcastMode optional	
186	}	
187		
188	type record BroadcastModeType {	
189	XmlElementStringValue text_	
190	}	
191		
192	type record UserIDType {	
193	UInt8 type_	
194	}	
195	}	
196		
197	// change 1 (WK23): Service Request extensions for content protection tests	
198	// according to scheme for Service Response (see OMA-TS-BaCST_Services 5.1.5.2.2)	
199	group ServiceResponse_TypeDefinition {	
200	type record ServiceResponseType {	
201	UInt requestID optional,	
202	UInt8 globalStatusCode optional,	
203	charstring adaptationMode optional,	
204	PurchaseItemResponseList PurchaseItems	
205	// DrmProfileSpecificPartType DrmProfileSpecificPart // not applicable to smartcard	
206	}	
207		
208	type set length (1 .. infinity) of PurchaseItemWithStatusAndTimesType PurchaseItemResponseList;	
209		
210	type record PurchaseItemWithStatusAndTimesType {	
211	AnyUri globalIDRef,	
212	UInt8 itemwiseStatusCode optional,	
213	SubscriptionWindowType SubscriptionWindow optional	

Datei: LibCommon_BSM_TypesAndValues.ttcn (Fortsetzung)

214	}			
215	}			
216	}	=	104	}

Datei: LibCommon_GBA_BSF.ttcn

1	//change 1 (WK18): for content protection tests	=	1	//change 1 (WK18): for content protection tests
2	module LibCommon_GBA_BSF {		2	module LibCommon_GBA_BSF {
3	import from LibCommon_HTTP_TypesAndValues all;		3	import from LibCommon_HTTP_TypesAndValues all;
4	import from LibCommon_DataStrings all;		4	import from LibCommon_DataStrings all;
5	import from AtsBCast_ModuleParameters {		5	import from AtsBCast_ModuleParameters {
6	group GBA;		6	group GBA;
7	group BSM; // change 2 (WK23): GBA authentication/authorization extensions for content protection	+-		
8	}	=	7	}
9	import from LibCommon_GBA_BSF_TypesAndValues all;		8	import from LibCommon_GBA_BSF_TypesAndValues all;
10	import from LibCommon_Time all;	<>	9	import from LibCommon_Time all;
11			10	
12	group BSF {	=	11	group BSF {
13	group BSF_Templates {		12	group BSF_Templates {
14	group HTTPTemplates {		13	group HTTPTemplates {
15	group RequestTemplates {	<>	14	group RequestTemplates {
16	template HttpRequest mw_anyHttpRequest := {	=	15	template HttpRequest mw_anyHttpRequest := {
17	Method := ?,		16	Method := ?,
18	Uri := ?,		17	Uri := ?,
19	Version := ?,		18	Version := ?,
20	UriParameters := *,		19	UriParameters := *,
21	Host := ?,		20	Host := ?,
22	ContentType := *,		21	ContentType := *,
23	Authorization := *,		22	Authorization := *,
24	anotherHeaders := *,		23	anotherHeaders := *,
25	Body := *		24	Body := *
26	}		25	}
27		<>	26	
28	template HttpRequest mw_initialHttpRequest modifies mw_anyHttpRequest := {	=	27	template HttpRequest mw_initialHttpRequest modifies mw_anyHttpRequest := {
29	Authorization := {	<>	28	Authorization := {
30	Value := "Digest",		29	Value := "Digest",
31	ParameterList := {	=	30	ParameterList := {
32	{Name := "response", Value := ?},		31	{Name := "response", Value := ?},
33	{Name := "username", Value := "" & PX_IMSI},		32	{Name := "username", Value := "" & PX_IMSI},
34	*		33	*
35	}	<>	34	}
36	}	=	35	}
37	}		36	}
38	}		37	}
39		<>	38	
40	group ResponseTemplates {	=	39	group ResponseTemplates {
41	template HttpResponse m_basicResponse(integer p_statusCode) := {		40	template HttpResponse m_basicResponse(integer p_statusCode) := {
42	Version := "HTTP/1.1",		41	Version := "HTTP/1.1",
43	Code := p_statusCode,		42	Code := p_statusCode,
44	ContentType := omit,		43	ContentType := omit,
45	anotherHeaders := {	<>	44	anotherHeaders := {
46	{Name := "Server", ListOfValue := {m_defaultBsfServer}},	=	45	{Name := "Server", ListOfValue := {m_defaultBsfServer}}
47	,		46	,
48	,		47	,
49	Body := omit		48	Body := omit
50	}		49	}
51		<>	50	
52	template HttpResponse m_digestBsfResponse(charstring p_nonce) modifies mw_anyHttpRequest := {		51	template HttpResponse m_digestBsfResponse(integer p_statusCode, charstring p_nonce) modifies mw_anyHttpRequest := {
53	anotherHeaders := {		52	anotherHeaders := {
54	{Name := "Server", ListOfValue := {m_defaultBsfServer}},	=	53	{Name := "Server", ListOfValue := {m_defaultBsfServer}}
55	{Name := "WWW-Authenticate", ListOfValue := {m_defaultBsfWWWAuthenticate}},		54	{Name := "WWW-Authenticate", ListOfValue := {m_defaultBsfWWWAuthenticate}}
56	}		55	}
57	}		56	}
58		<>	57	
59	template HttpResponse m_successBsfResponse(charstring p_response, contentType p_contentType) := {		58	template HttpResponse m_successBsfResponse(integer p_statusCode, charstring p_response, contentType p_contentType) := {
60	ContentType := {	=	59	ContentType := {
61	Name := "Content-Type", ListOfValue := {{Value := "application/xml"},		60	Name := "Content-Type", ListOfValue := {{Value := "application/xml"},
62	},		61	},
63	anotherHeaders := {		62	anotherHeaders := {
64	{Name := "Server", ListOfValue := {m_defaultBsfServer}},		63	{Name := "Server", ListOfValue := {m_defaultBsfServer}}
65	{Name := "Expires", ListOfValue := {{Value := "no-cache"},		64	{Name := "Expires", ListOfValue := {{Value := "no-cache"},
66	{Name := "Authentication-Info", ListOfValue := {m_defaultBsfAuthenticationInfo}},		65	{Name := "Authentication-Info", ListOfValue := {m_defaultBsfAuthenticationInfo}}
67	},		66	},
68	Body := {		67	Body := {
69	BootstrappingInfo := p_bsFXML		68	BootstrappingInfo := p_bsFXML
70	}		69	}
71	}		70	}

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

72	}		71	}
73		<>	72	
74	group HttpHeaderTemplates {	=	73	group HttpHeaderTemplates {
75	template HttpHeaderValue m_defaultBsfServerHeader := {		74	template HttpHeaderValue m_defaultBsfServerHeader := {
76	Value := "RS ATE BSF Server/0.02",		75	Value := "RS ATE BSF Server/0.02",
77	ParameterList := omit		76	ParameterList := omit
78	}		77	}
79		<>	78	
80	template HttpHeaderValue m_defaultWwwAuthenticate(charstring p_nonce	=	79	template HttpHeaderValue m_defaultWwwAuthenticate(charstring p_nonce
81	Value := "Digest",		80	Value := "Digest",
82	ParameterList := {		81	ParameterList := {
83	{Name := "realm", Value := "" & PX_BSF_FQDN & " " & "realm",		82	{Name := "realm", Value := "" & PX_BSF_FQDN & " " & "realm",
84	{Name := "nonce", Value := "" & p_nonce & ""},		83	{Name := "nonce", Value := "" & p_nonce & ""},
85	{Name := "algorithm", Value := "AKAv1-MD5"},		84	{Name := "algorithm", Value := "AKAv1-MD5"},
86	{Name := "qop", Value := "" & "auth-int" & ""}		85	{Name := "qop", Value := "" & "auth-int" & ""}
87	}		86	}
88	}		87	}
89		<>	88	
90	template HttpHeaderValue m_defaultAuthInfoHeaderRspauth(charstring	=	89	template HttpHeaderValue m_defaultAuthInfoHeaderRspauth(charstring
91	Value := "rspauth",		90	Value := "rspauth",
92	ParameterList := {{ Name := p_response, Value := on		91	ParameterList := {{ Name := p_response, Value := on
93	}		92	}
94	}		93	}
95		<>	94	
96	template HttpHeaderValue m_defaultAuthInfoHeaderQop := {	=	95	template HttpHeaderValue m_defaultAuthInfoHeaderQop := {
97	Value := "qop",		96	Value := "qop",
98	ParameterList := {{ Name := "auth-int", Value := on		97	ParameterList := {{ Name := "auth-int", Value := on
99	}		98	}
100	}		99	}
101	}		100	}
102	}		101	}
103		<>	102	
104	group BSFXMLTemplate {	=	103	group BSFXMLTemplate {
105	template BootstrappingInfoType m_bsfXml(charstring p_btid, charstring p_lifeTime		104	template BootstrappingInfoType m_bsfXml(charstring p_btid, charstring p_lifeTime
106	btid := { text_ := p_btid },		105	btid := { text_ := p_btid },
107	lifeTime := { text_ := p_lifeTime }		106	lifeTime := { text_ := p_lifeTime }
108	}		107	}
109	}		108	}
110	}		109	}
111		<>	110	
112	group BSF_Functions {	=	111	group BSF_Functions {
113	/* change 2 (WK23): GBA authentication/authorization extensions for content protect	<>	112	function BSF_ServerSimulation(boolean p_isParallelComponent) runs on BSFServerComp
114	/**			
115	* @desc Initialisation of BSF depending values.			
116	* @param p_ks_ext_NAF Returns Ks_ext_NAF value. Used from BSM server.			
117	*/			
118	function f_bsf_init(out octetstring p_ks_ext_NAF, out octetstring p_ks_int_NAF) run			
119				
120	// BSF generates authentication vector and calculate expected response from	=	113	// BSF generates authentication vector and calculate expected response from
121	var bitstring v_IK := '0'B, v_CK := '0'B;	<>	114	var bitstring v_IK := '0'B, v_CK := '0'B, v_XRES := '0'B;
			115	var Bit128 v_AUTN := ts_AuthenticationInit(v_IK, v_CK, v_XRES);
			116	var charstring vNonce := fx_bitstring2Base64(PX_AuthRAND & v_AUTN);
122	var Bit128 v_AUTN;		117	
			118	//BSF generate B-TID -> TS 33.220 4.5.2
			119	var charstring btid := f_generateBtid();
123		=	120	
124	// BSF generate key material -> TS 33.220 4.5.2		121	// BSF generate key material -> TS 33.220 4.5.2
125	var octetstring v_UaSecProtocolId := '0100000001'0;	<>	122	var bitstring Ks := v_CK & v_IK;
126	var octetstring v_NAF_Id := fx_char2octet(PX_BSM_FQDN) & v_UaSecProtocolId;		123	// log("new Ks: " & bit2str(Ks));
127			124	
128	if (PX_GBA == e_gba_me) {		125	var charstring sameTime;
129	v_AUTN := ts_AuthenticationInit(v_IK, v_CK, xres);		126	var BootstrappingInfoType v_bsfXml;
130	}		127	var octetstring rspAuth;
131	else {		128	var HttpResponse response;
132	v_AUTN := ts_AuthenticationInitForGbaU(v_IK, v_CK, xres);		129	// BSF generate Ks_(ext)_NAF = KDF(Ks, "gba-me/u", RAND, IMPI, NAF_Id)
133	}		130	// hint: NAF_Id can computed if the used cipher suite is known!!!
134			131	// -> 1st step PSK-TLS handshake between peer and H-SLP in order to gather
135	// nonce for HTTP Digest		132	// -> assumption: key generation is according to TS 33.220 4.5.2
136	nonce := fx_bitstring2Base64(PX_AuthRAND & v_AUTN);		133	//var octetstring UaSecProtocolId := generateUaSecurityProtocolId(mapChiper
137	//p_nonce := nonce;		134	// log("new UaSecProtocolId: " & oct2str(UaSecProtocolId));
138			135	//var octetstring NAF_Id := fx_char2octet(PX_H_SLP_FQDN) & UaSecProtocolId

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

139	//now we can generate Ks_ext(int)_NAF = KDF(Ks, "gba-me/u", RAND, IMPI, NAF)		136	//var octetstring impi := fx_char2octet(PX_IMPI);
140	ks_ext_NAF := fx_calculatingKDF4NAF(bit2oct(v_CK & v_IK), fx_char2octet("gba-me/u", RAND, IMPI, NAF))			
141	p_ks_ext_NAF := ks_ext_NAF;			
142	ks_int_NAF := fx_calculatingKDF4NAF(bit2oct(v_CK & v_IK), fx_char2octet("gba-me/u", RAND, IMPI, NAF))			
143	p_ks_int_NAF := ks_int_NAF;			
144	}		137	
145			138	// BSF is ready
146	// change 2 (WK23): GBA authentication/authorization extensions for content protection			
147	function BSF_ServerSimulation(boolean p_isParallelComponent) runs on BSFServerComponent			
148	var HttpResponse v_response;			
149	var HttpRequest v_request;	=	139	var HttpRequest v_request;
150		<>	140	log("BSF waiting...");
151	var BootstrappingInfoType v_bsfxml;			
152	var octetstring v_rspauth;			
153				
154	log("BSF ready...");			
155	if (not p_isParallelComponent) {		141	if (not p_isParallelComponent) {
156	t_wait.start(PX_TWAIT);		142	t_wait.start(PX_TWAIT);
157	}		143	}
158	alt {	=	144	alt {
159	[] bsf.receive(mw_initialHttpRequest) -> value v_request {		145	[] bsf.receive(mw_initialHttpRequest) -> value v_request {
160	log("Verdict Info: 4a. The terminal sends a POST request with digest response");		146	log("Verdict Info: 4a. The terminal sends a POST request with digest response");
161	setverdict(pass);		147	setverdict(pass);
162	bsf.send(m_digestBsfResponse(nonce));	<>	148	bsf.send(m_digestBsfResponse(401,vNonce));
163	alt {	=	149	alt {
164	[] bsf.receive(mw_initialHttpRequest) -> value v_request {	<>	150	[] bsf.receive(mw_initialHttpRequest) -> value v_request {
165	if (match(str2oct(unq(getHeaderParameterValue("digest-response"), v_request)))		151	if (match(str2oct(unq(getHeaderParameterValue("digest-response"), v_request)))
166	log("Verdict Info: 4c. RES corresponds to the request");	=	152	log("Verdict Info: 4c. RES corresponds to the request");
167	setverdict(pass);	=	153	setverdict(pass);
168	v_bsfxml := valueof(m_bsfxml(btId, v_request));	<>	154	sameTime := fx_date2RFC1123(2); //
169	v_rspauth := ts_calculateRspauth(v_request, v_bsfxml);		155	v_bsfxml := valueof(m_bsfxml(btId, v_request));
170	v_response := m_successBsfResponse(v_request, v_rspauth);		156	rspauth := ts_calculateRspauth(v_request, v_bsfxml);
171	bsf.send(v_response);		157	response := valueof(m_successBsfResponse(v_request, v_rspauth));
172	}	=	158	bsf.send(response);
173	else {		159	}
174	log("Verdict Info: RES doesn't correspond to the request");		160	else {
175	setverdict(fail);		161	log("Verdict Info: RES doesn't correspond to the request");
176	bsf.send(m_basicResponse(400));		162	setverdict(fail);
177	}		163	bsf.send(m_basicResponse(400));
178	}		164	}
179	}		165	}
180	if (p_isParallelComponent) {		166	}
181	repeat;		167	if (p_isParallelComponent) {
182	}		168	repeat;
183	}		169	}
184	[] bsf.receive {		170	}
185	log("Verdict Info: The terminal doesn't send a POST request with digest response");		171	[] bsf.receive {
186	setverdict(fail);		172	log("Verdict Info: The terminal doesn't send a POST request with digest response");
187	bsf.send(m_basicResponse(400));		173	setverdict(fail);
188	if (p_isParallelComponent) {	<>	174	bsf.send(m_basicResponse(400));
189	repeat;		175	if (p_isParallelComponent) {
190	}	=	176	repeat;
191	}		177	}
192	[not p_isParallelComponent] t_wait.timeout {	<>	178	}
193	log("Verdict Info: The terminal doesn't send a HTTP request with digest response");		179	[not p_isParallelComponent] t_wait.timeout {
194	setverdict(inconc);		180	log("Verdict Info: The terminal doesn't send a HTTP request for BSF.");
195	}		181	setverdict(inconc);
196	} // alt		182	}
197	return;	=	183	} // alt
198	}		184	return;
199			185	}
200	function ts_AuthResponseCheckAKAv1MD5(in HttpRequest p_httpRequest, in bitstring p_digestResponse, in bitstring p_XRES)	<>	186	
201	// input is the comma separated parameter list: digestResponse from the AuthenticationInit	=	187	function ts_AuthResponseCheckAKAv1MD5 (in HttpRequest p_httpRequest, in bitstring p_digestResponse, in bitstring p_XRES)
202	// and the value of XRES calculated in ts_AuthenticationInit		188	// input is the comma separated parameter list: digestResponse from the AuthenticationInit
203	// This function calculates the digest response (v_RequestDigest) for p_XRES	<>	189	// and the value of XRES calculated in ts_AuthenticationInit
204	// RFC 2617 and compares it with the response received from the UE (v_DigestResponse)	=	190	// This function calculates the digest response (v_RequestDigest) for p_XRES
205		<>	191	// RFC 2617 and compares it with the response received from the UE (v_DigestResponse)
206	var charstring v_Method := p_httpRequest.Method;	=	192	
207	var ListOfParameter paramList := p_httpRequest.Authorization.ParameterList;		193	var charstring v_Method := p_httpRequest.Method;
208	var charstring v_Username := getHeaderParameterValue("username", paramList);		194	var ListOfParameter paramList := p_httpRequest.Authorization.ParameterList;
			195	var charstring v_Username := getHeaderParameterValue("username", paramList);

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

209	var charstring v_Realm := getHeaderParameterValue("realm", paramList);		196	var charstring v_Realm := getHeaderParameterValue("realm", paramList);
210	var charstring v_DigestUri := getHeaderParameterValue("uri", paramList);		197	var charstring v_DigestUri := getHeaderParameterValue("uri", paramList);
211	var charstring v_Nonce := getHeaderParameterValue("nonce", paramList);		198	var charstring v_Nonce := getHeaderParameterValue("nonce", paramList);
212	var charstring v_NC := getHeaderParameterValue("nc", paramList);		199	var charstring v_NC := getHeaderParameterValue("nc", paramList);
213	var charstring v_CNonce := getHeaderParameterValue("cnonce", paramList);		200	var charstring v_CNonce := getHeaderParameterValue("cnonce", paramList);
214	var charstring v_Qop := getHeaderParameterValue("qop", paramList);		201	var charstring v_Qop := getHeaderParameterValue("qop", paramList);
215	var charstring v_Dresponse := getHeaderParameterValue("response", paramList);		202	var charstring v_Dresponse := getHeaderParameterValue("response", paramList);
		<>	203	
			204	
216		=	205	
217	var octetstring v_oMETHOD := fx_char2octet(v_Method);		206	var octetstring v_oMETHOD := fx_char2octet(v_Method);
218	var octetstring v_oUsername := fx_char2octet(unq(v_Username));		207	var octetstring v_oUsername := fx_char2octet(unq(v_Username));
219	var octetstring v_opasswd := bit2oct(p_XRES);		208	var octetstring v_opasswd := bit2oct(p_XRES);
220			209	
221	var octetstring v_oRealm := fx_char2octet(unq(v_Realm));		210	var octetstring v_oRealm := fx_char2octet(unq(v_Realm));
222	var octetstring v_oDigestUri := fx_char2octet(v_DigestUri);		211	var octetstring v_oDigestUri := fx_char2octet(v_DigestUri);
223	var octetstring v_oNonce := fx_char2octet(unq(v_Nonce));		212	var octetstring v_oNonce := fx_char2octet(unq(v_Nonce));
224	var octetstring v_oNC := fx_char2octet(v_NC);		213	var octetstring v_oNC := fx_char2octet(v_NC);
225	var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));		214	var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));
226	var octetstring v_oQop := fx_char2octet(v_Qop);		215	var octetstring v_oQop := fx_char2octet(v_Qop);
227	var octetstring v_oDresponse := str2oct(unq(v_Dresponse));		216	var octetstring v_oDresponse := str2oct(unq(v_Dresponse));
228		<>	217	
229	var octetstring v_Secret := '0;	=	218	var octetstring v_Secret := '0;
230	var octetstring v_HA2 := '0;		219	var octetstring v_HA2 := '0;
231	var octetstring v_Data := '0;		220	var octetstring v_Data := '0;
232	var octetstring v_RequestDigest := '0;		221	var octetstring v_RequestDigest := '0;
233			222	
234	// RFC 2617 3.2.2.2		223	// RFC 2617 3.2.2.2
235	// A1 = unq(username-value) ":" unq(realm-value) ":" passwd		224	// A1 = unq(username-value) ":" unq(realm-value) ":" passwd
		<>	225	var octetstring A1 :=
236	var octetstring A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & v_oQop		226	v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & v_opasswd;
237			227	
238	var octetstring A2, secret_data;		228	var octetstring A2, secret_data;
239		=	229	
240	// H(A1)		230	// H(A1)
241	v_Secret := fx_charLow2octet(oct2str(fx_md5_hex(A1)));		231	v_Secret := fx_charLow2octet(oct2str(fx_md5_hex(A1)));
242			232	
243	// RFC 2617 3.2.2.3		233	// RFC 2617 3.2.2.3
244	// A2 = Method ":" digest-uri-value		234	// A2 = Method ":" digest-uri-value
245	A2 := v_oMETHOD & COLON_HEX & v_oDigestUri;		235	A2 := v_oMETHOD & COLON_HEX & v_oDigestUri;
246			236	
247	v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));		237	v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));
248			238	
249	// RFC 2617 3.2.2.1		239	// RFC 2617 3.2.2.1
250	// unq(nonce-value)		240	// unq(nonce-value)
251	// ":" nc-value		241	// ":" nc-value
252	// ":" unq(cnonce-value)		242	// ":" unq(cnonce-value)
253	// ":" unq(qop-value)		243	// ":" unq(qop-value)
254	// ":" H(A2)		244	// ":" H(A2)
255	v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce		245	v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce
256			246	
257	//now apply KD function described in RFC 2617 section 3.2.1		247	//now apply KD function described in RFC 2617 section 3.2.1
258	//KD(secret, data) = H(concat(secret, ":", data))		248	//KD(secret, data) = H(concat(secret, ":", data))
259	secret_data := v_Secret & COLON_HEX & v_Data;		249	secret_data := v_Secret & COLON_HEX & v_Data;
260	v_RequestDigest := fx_md5_hex(secret_data);		250	v_RequestDigest := fx_md5_hex(secret_data);
261		<>	251	
262	log("ts_AuthResponseCheckAKAv1MD5(): METHOD '" & v_Method & "'");		252	log("ts_AuthResponseCheckAKAv1MD5(): METHOD '" & v_Method & "'");
263	log("ts_AuthResponseCheckAKAv1MD5(): client username '" & v_Username & "'");		253	log("ts_AuthResponseCheckAKAv1MD5(): client username '" & v_Username & "'");
264	log("ts_AuthResponseCheckAKAv1MD5(): client realm '" & v_Realm & "'");		254	log("ts_AuthResponseCheckAKAv1MD5(): client realm '" & v_Realm & "'");
265	log("ts_AuthResponseCheckAKAv1MD5(): client URI '" & v_DigestUri & "'");		255	log("ts_AuthResponseCheckAKAv1MD5(): client URI '" & v_DigestUri & "'");
266	log("ts_AuthResponseCheckAKAv1MD5(): client nonce '" & v_Nonce & "'");		256	log("ts_AuthResponseCheckAKAv1MD5(): client nonce '" & v_Nonce & "'");
267	log("ts_AuthResponseCheckAKAv1MD5(): client nc '" & v_NC & "'");		257	log("ts_AuthResponseCheckAKAv1MD5(): client nc '" & v_NC & "'");
268	log("ts_AuthResponseCheckAKAv1MD5(): client CNonce '" & v_CNonce & "'");		258	log("ts_AuthResponseCheckAKAv1MD5(): client CNonce '" & v_CNonce & "'");
269	log("ts_AuthResponseCheckAKAv1MD5(): client QoP '" & v_Qop & "'");		259	log("ts_AuthResponseCheckAKAv1MD5(): client QoP '" & v_Qop & "'");
270	log("ts_AuthResponseCheckAKAv1MD5(): client response value '" & v_Dresponse		260	log("ts_AuthResponseCheckAKAv1MD5(): client response value '" & v_Dresponse & "'");
271			261	
272	log("ts_AuthResponseCheckAKAv1MD5(): server XRES: " & oct2str(v_opasswd));		262	log("ts_AuthResponseCheckAKAv1MD5(): server XRES: " & oct2str(v_opasswd));
273			263	
274	log("ts_AuthResponseCheckAKAv1MD5(): calculated response value "" & oct2st		264	log("ts_AuthResponseCheckAKAv1MD5(): calculated response value "" & oct2st
275		=	265	
276	// if (v_RequestDigest == v_oDresponse) {		266	// if (v_RequestDigest == v_oDresponse) {

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

277	//	return (true)		267	//	return (true)
278	//	} else {		268	//	} else {
279	//	return (false)		269	//	return (false)
280	//	}		270	//	}
281		return v_RequestDigest;		271		return v_RequestDigest;
282		}		272		}
283				273		
284		function ts_calculateRspauth(in ListOfParameter p_list, in bitstring p_XRES, in charstring p_nonce)		274		function ts_calculateRspauth(in ListOfParameter p_list, in bitstring p_XRES, in charstring p_nonce)
285		// RFC 2617 and 3310		275		// RFC 2617 and 3310
286		var charstring v_NC := getHeaderParameterValue("nc", p_list);		276		var charstring v_NC := getHeaderParameterValue("nc", p_list);
287		var charstring v_CNonce := getHeaderParameterValue("cnonce", p_list);		277		var charstring v_CNonce := getHeaderParameterValue("cnonce", p_list);
288		var charstring v_DigestUri := getHeaderParameterValue("uri", p_list);		278		var charstring v_DigestUri := getHeaderParameterValue("uri", p_list);
289			<>	279		
290		var octetstring v_oUsername := fx_char2octet(PX_IMPI);	=	280		var octetstring v_oUsername := fx_char2octet(PX_IMPI);
291		var octetstring v_opasswd := bit2oct(p_XRES);		281		var octetstring v_opasswd := bit2oct(p_XRES);
292		var octetstring v_oRealm := fx_char2octet(PX_BSF_FQDN);		282		var octetstring v_oRealm := fx_char2octet(PX_BSF_FQDN);
293		var octetstring v_oDigestUri := fx_char2octet("/");		283		var octetstring v_oDigestUri := fx_char2octet("/");
294		var octetstring v_oNonce := fx_char2octet(p_nonce);		284		var octetstring v_oNonce := fx_char2octet(p_nonce);
295		var octetstring v_oNC := fx_char2octet(v_NC);		285		var octetstring v_oNC := fx_char2octet(v_NC);
296		var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));		286		var octetstring v_oCNonce := fx_char2octet(unq(v_CNonce));
297		var octetstring v_oQop := fx_char2octet("auth-int");		287		var octetstring v_oQop := fx_char2octet("auth-int");
298			<>	288		
				289		
				290		
299		var octetstring v_Secret := 'O;	=	291		var octetstring v_Secret := 'O;
300		var octetstring v_HA2 := 'O;		292		var octetstring v_HA2 := 'O;
301		var octetstring v_Data := 'O;		293		var octetstring v_Data := 'O;
302		var octetstring v_RequestDigest := 'O;		294		var octetstring v_RequestDigest := 'O;
303				295		
304		// RFC 2617 3.2.2.2		296		// RFC 2617 3.2.2.2
305		// A1 = unq(username-value) ":" unq(realm-value) ":" passwd		297		// A1 = unq(username-value) ":" unq(realm-value) ":" passwd
306		var octetstring A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & v_opasswd;		298		var octetstring A1 := v_oUsername & COLON_HEX & v_oRealm & COLON_HEX & v_opasswd;
307			<>	299		
308		var octetstring h_entityBody, A2, secret_data;		300		var octetstring h_entityBody, A2, secret_data;
309			=	301		
310		// H(A1)		302		// H(A1)
311		v_Secret := fx_charLow2octet(oct2str(fx_md5_hex(A1)));		303		v_Secret := fx_charLow2octet(oct2str(fx_md5_hex(A1)));
312				304		
313		// RFC 2617 3.2.3		305		// RFC 2617 3.2.3
314		// A2 = ":" digest-uri-value ":" H(entity-body)		306		// A2 = ":" digest-uri-value ":" H(entity-body)
315		h_entityBody := fx_charLow2octet(oct2str(fx_md5_hex(p_entityBody)));		307		h_entityBody := fx_charLow2octet(oct2str(fx_md5_hex(p_entityBody)));
316		A2 := COLON_HEX & v_oDigestUri & COLON_HEX & h_entityBody;		308		A2 := COLON_HEX & v_oDigestUri & COLON_HEX & h_entityBody;
317				309		
318		v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));		310		v_HA2 := fx_charLow2octet(oct2str(fx_md5_hex(A2)));
319				311		
320		// RFC 2617 3.2.2.1		312		// RFC 2617 3.2.2.1
321		// unq(nonce-value) ":" nc-value ":" unq(cnonce-value) ":" unq(qop-value)		313		// unq(nonce-value) ":" nc-value ":" unq(cnonce-value) ":" unq(qop-value)
322		v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce & COLON_HEX & v_oQop);		314		v_Data := (v_oNonce & COLON_HEX & v_oNC & COLON_HEX & v_oCNonce & COLON_HEX & v_oQop);
323				315		
324		//now apply KD function described in RFC 2617 section 3.2.1		316		//now apply KD function described in RFC 2617 section 3.2.1
325		//KD(secret, data) = H(concat(secret, ":", data))		317		//KD(secret, data) = H(concat(secret, ":", data))
326		secret_data := v_Secret & COLON_HEX & v_Data;		318		secret_data := v_Secret & COLON_HEX & v_Data;
327			<>	319		
328		log("ts_calculateRspauth(): v_NC -> " & v_NC);		320		log("ts_calculateRspauth(): v_NC -> " & v_NC);
329		log("ts_calculateRspauth(): v_CNonce -> " & v_CNonce);		321		log("ts_calculateRspauth(): v_CNonce -> " & v_CNonce);
330		log("ts_calculateRspauth(): v_oUsername/ixit_IMPI_USIM -> " & PX_IMPI);		322		log("ts_calculateRspauth(): v_oUsername/ixit_IMPI_USIM -> " & PX_IMPI);
331		log("ts_calculateRspauth(): v_opasswd -> " & oct2str(v_opasswd));		323		log("ts_calculateRspauth(): v_opasswd -> " & oct2str(v_opasswd));
332		log("ts_calculateRspauth(): v_oRealm/ixit_BSF_FQDN -> " & PX_BSF_FQDN);		324		log("ts_calculateRspauth(): v_oRealm/ixit_BSF_FQDN -> " & PX_BSF_FQDN);
333		log("ts_calculateRspauth(): v_DigestUri -> " & v_DigestUri);		325		log("ts_calculateRspauth(): v_DigestUri -> " & v_DigestUri);
334		log("ts_calculateRspauth(): v_oNonce/p_nonce -> " & p_nonce);		326		log("ts_calculateRspauth(): v_oNonce/p_nonce -> " & p_nonce);
335		log("ts_calculateRspauth(): v_oNC/v_NC -> " & v_NC);		327		log("ts_calculateRspauth(): v_oNC/v_NC -> " & v_NC);
336		log("ts_calculateRspauth(): v_oCNonce -> " & v_CNonce);		328		log("ts_calculateRspauth(): v_oCNonce -> " & v_CNonce);
337		log("ts_calculateRspauth(): v_oQop -> " & oct2str(v_oQop));		329		log("ts_calculateRspauth(): v_oQop -> " & oct2str(v_oQop));
338				330		
339		return fx_md5_hex(secret_data);	=	331		return fx_md5_hex(secret_data);
340		}		332		}
341				333		
342		function unq(in charstring p_string) return charstring {		334		function unq(in charstring p_string) return charstring {
343		var charstring retVal;	<>	335		var charstring retVal;
344		log("unq(): in -> " & p_string);	=	336		log("unq(): in -> " & p_string);
345			<>	337		

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

346	if ((substr(p_string, 0, 1) == "") and (substr(p_string, lengthof(p_string) - 1, lengthof(p_string) - 2) == ""))	=	338	if ((substr(p_string, 0, 1) == "") and (substr(p_string, lengthof(p_string) - 1, lengthof(p_string) - 2) == ""))
347	retVal := substr(p_string, 1, lengthof(p_string) - 2);		339	retVal := substr(p_string, 1, lengthof(p_string) - 2);
348	}		340	}
349	else {		341	else {
350	retVal := p_string;		342	retVal := p_string;
351	}		343	}
352		<>	344	
353	log("unq(): out -> " & retVal);	=	345	log("unq(): out -> " & retVal);
354	return retVal;		346	return retVal;
355	}		347	}
356		<>	348	
357	function getHeaderParameterValue(in charstring p_headerParameterValue, in ListOfParameters p_list) returns integer	=	349	function getHeaderParameterValue(in charstring p_headerParameterValue, in ListOfParameters p_list) returns integer
358	var integer i;		350	var integer i;
359		<>	351	
360	for (i := 0; i < sizeof(p_list); i:= i+1){	=	352	for (i := 0; i < sizeof(p_list); i:= i+1){
361	if (p_list[i].Name == p_headerParameterValue) {		353	if (p_list[i].Name == p_headerParameterValue) {
362	return (p_list[i].Value);		354	return (p_list[i].Value);
363	}		355	}
364	}		356	}
365		<>	357	
366	return "";	=	358	return "";
367	}		359	}
368		<>	360	
369	function ts_AuthenticationInit(out bitstring v_IKey, out bitstring v_CKey, out bitstring v_XRES) returns bitstring		361	function ts_AuthenticationInit(out bitstring v_IKey, out bitstring v_CKey, out bitstring v_XRES) returns bitstring
			362	
			363	
			364	
370	// Calculation is done according to 34.108, clause 8.1.2 and 33.102, clause 8.1.2	=	365	// Calculation is done according to 34.108, clause 8.1.2 and 33.102, clause 8.1.2
371	var Bit128 v_XDOut, v_AUTN;		366	var Bit128 v_XDOut, v_AUTN;
372	var bitstring v_AUTN_2;		367	var bitstring v_AUTN_2;
373	var Bit64 v_CDOut, v_XDOut_Half, v_MAC;		368	var Bit64 v_CDOut, v_XDOut_Half, v_MAC;
374	var Bit48 v_AK;		369	var Bit48 v_AK;
375	var bitstring v_AUTN_1;		370	var bitstring v_AUTN_1;
376			371	
377	v_XDOut := PX_AuthRAND xor4b PX_AuthK;		372	v_XDOut := PX_AuthRAND xor4b PX_AuthK;
378			373	
379	v_CDOut := PX_AuthSQN & PX_AuthAMF;		374	v_CDOut := PX_AuthSQN & PX_AuthAMF;
380			375	
381	v_XDOut_Half := substr (v_XDOut, 0, 64);		376	v_XDOut_Half := substr (v_XDOut, 0, 64);
382			377	
383	v_AK := substr (v_XDOut, 24, 48);		378	v_AK := substr (v_XDOut, 24, 48);
384			379	
385	v_AUTN_1 := PX_AuthSQN xor4b v_AK;		380	v_AUTN_1 := PX_AuthSQN xor4b v_AK;
386			381	
387	v_MAC := v_XDOut_Half xor4b v_CDOut;		382	v_MAC := v_XDOut_Half xor4b v_CDOut;
388			383	
389	v_AUTN_2 := PX_AuthAMF & v_MAC;		384	v_AUTN_2 := PX_AuthAMF & v_MAC;
390			385	
391	v_AUTN := v_AUTN_1 & v_AUTN_2;		386	v_AUTN := v_AUTN_1 & v_AUTN_2;
392			387	
393	// v_IKey := 128 bits of v_XDOut, wrapped, starting from offset 16		388	// v_IKey := 128 bits of v_XDOut, wrapped, starting from offset 16
394	v_IKey :=		389	v_IKey :=
395	substr (v_XDOut, 16, (128 - 16)) & substr (v_XDOut, 0, 16);		390	substr (v_XDOut, 16, (128 - 16)) & substr (v_XDOut, 0, 16);
396			391	
397	// v_CKey := 128 bits of v_XDOut, wrapped, starting from offset 8		392	// v_CKey := 128 bits of v_XDOut, wrapped, starting from offset 8
398	v_CKey :=		393	v_CKey :=
399	substr (v_XDOut, 8, (128 - 8)) & substr (v_XDOut, 0, 8);		394	substr (v_XDOut, 8, (128 - 8)) & substr (v_XDOut, 0, 8);
400			395	
401	v_XRES := substr (v_XDOut, 0, (PX_AuthN + 1));		396	v_XRES := substr (v_XDOut, 0, (PX_AuthN + 1));
402			397	
403	return (v_AUTN);		398	return (v_AUTN);
404	}		399	}
405		<>	400	
406	// change 2 (WK23): GBA authentication/authorization extensions for content protection			
407	function ts_AuthenticationInitForGbaU(out bitstring v_IKey, out bitstring v_CKey, out bitstring v_XRES) returns bitstring			
408	// Calculation is done according to 34.108, clause 8.1.2 and 33.102, clause 8.1.2			
409	var Bit128 v_XDOut, v_AUTN;			
410	var Bit64 v_CDOut, v_XDOut_Half, v_MAC, v_MAC_;			
411	var Bit48 v_AK;			
412				
413	v_XDOut := PX_AuthRAND xor4b PX_AuthK;			

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

414					
415	v_CDOut := PX_AuthSQN & PX_AuthAMF;				
416					
417	v_XDOut_Half := substr (v_XDOut, 0, 64);				
418					
419	v_AK := substr (v_XDOut, 24, 48);				
420					
421	v_MAC := v_XDOut_Half xor4b v_CDOut;				
422					
423	// v_IKey := 128 bits of v_XDOut, wrapped, starting from offset 16				
424	v_IKey :=				
425	substr (v_XDOut, 16, (128 - 16)) & substr (v_XDOut, 0, 16);				
426					
427	// v_CKey := 128 bits of v_XDOut, wrapped, starting from offset 8				
428	v_CKey :=				
429	substr (v_XDOut, 8, (128 - 8)) & substr (v_XDOut, 0, 8);				
430					
431	v_MAC_ := v_MAC xor4b substr(oct2bit((fx_sha_1_hex(bit2oct(v_IKey)))), 0, 64);				
432					
433	v_AUTN_ := (PX_AuthSQN xor4b v_AK) & PX_AuthAMF & v_MAC_;				
434					
435	// length of v_XRES is depending on PX_AuthN				
436	v_XRES := substr (v_XDOut, 0, (PX_AuthN + 1));				
437	// flipping the last significant bit				
438	v_XRES := v_XRES xor4b int2bit(1, PX_AuthN + 1);				
439					
440	return (v_AUTN_);				
441	}				
442					
443					
444	/**	=	401	/**	
445	*	<>	402	*	
446	* @desc BSF generate B-TID (Bootstrapping Transaction Identifier) -> TS 33.220 4.5.2		403	* @desc BSF generate B-TID -> TS 33.220 4.5.2	
447	* @return B-TID as charstring	=	404	* @return B-TID as charstring	
448	*/		405	*/	
449	function f_bsf_generateBtid() return charstring {	<>	406	function f_generateBtid() return charstring {	
450	return fx_bitstring2Base64(PX_AuthRAND) & "@" & PX_BSF_FQDN;	=	407	return fx_bitstring2Base64(PX_AuthRAND) & "@" & PX_BSF_FQDN;	
451	}		408	}	
452	// change 2 (WK23): GBA authentication/authorization extensions for content protection	<>	409		
453	/**		410	/**	
			411	//	
			412	//	* @desc The Ua security protocol identifier is a string of five octets.
			413	//	* The first octet denotes the organization which specifies the Ua security
			414	//	* protocol. The four remaining octets denote a specific security protocol
			415	//	* within the responsibility of the organization.
			416	//	* The following values for organizations are assigned:
			417	//	* "0x01" 3GPP
			418	//	* "0x02" 3GPP2
			419	//	* "0x03" Open Mobile Alliance
			420	//	* "0x04" GSMA
			421	//	* (0x01,0x00,0x01,yy,zz) Ua security protocol for "Shared key-based UE
			422	//	* authentication with certificatebased NAF authentication", according to TS 33.222
			423	//	* , or "Shared key-based mutual authentication between UE and NAF", according to
			424	//	* TS 33.222. Here, "yy,zz" is the protection mechanism CipherSuite code according
			425	//	* to the defined values for TLS CipherSuites in TLS V1.0 and PSK Ciphersuites for
			426	//	*
454	*		427	//	* @param p_cipherSuiteNumber
455	* @desc Sets a new B-TID for BSF.		428	//	* @return
456	* @param p_btid		429	//	* @verdict
457	* @verdict		430	//	*/
458	*/		431	//	function generateUaSecurityProtocolId(in octetstring p_cipherSuiteNumber) return octetstring {
459	function f_bsf_setBtid(charstring p_btid) runs on BSFServerComponent {		432	//	return PX_OrganizationOctet & '0001'O & p_cipherSuiteNumber;
			433		}
			434	//	function mapChiperSuiteToNumber(in CipherSuites p_ciphersuite) return octetstring {
			435	//	var octetstring csNumber := '0000'O;
			436	//	if (p_ciphersuite == TLS_RSA_WITH_NULL_SHA) {
			437	//	csNumber := '0002'O;
			438	//	}
			439	//	else if (p_ciphersuite == TLS_RSA_WITH_3DES_EDE_CBC_SHA) {
460	btid := p_btid;		440	//	csNumber := '000a'O;
			441	//	}

Datei: LibCommon_GBA_BSF.ttcn (Fortsetzung)

461			442	//	else if (p_ciphersuite == TLS_RSA_WITH_AES_128_CBC_SHA) {
462			443	//	csNumber := '002f'O;
			444	//	}
			445	//	else if (p_ciphersuite == TLS_PSK_WITH_3DES_EDE_CBC_SHA) {
			446	//	csNumber := '008b'O;
			447	//	}
			448	//	else if (p_ciphersuite == TLS_PSK_WITH_AES_128_CBC_SHA) {
			449	//	csNumber := '008c'O;
			450	//	}
			451	//	else {
			452	//	log("Verdict Info: unknown cipher suite");
			453	//	setverdict(inconc);
			454	//	}
			455	//	return csNumber;
			456	//	}
			457		
463	const octetstring COLON_HEX := '3A'O;	=	458		const octetstring COLON_HEX := '3A'O;
464	}		459		}
465		<>	460		
466	group BSF_Configuration_and_Components {	=	461	group BSF_Configuration_and_Components {	
467	type component BSFServerComponent {		462	type component BSFServerComponent {	
468	port BSFPort bsf;		463	port BSFPort bsf;	
469		<>	464		
470	timer t_wait;	=	465	timer t_wait;	
471		+-			
472	//				
473	change 2 (WK23): GBA authentication/authorization extensions for content protection				
474	var charstring btid;				
475	var charstring nonce;				
476	var octetstring ks_ext_NAF, ks_int_NAF;				
477	var bitstring xres;				
478	}	=	466	}	
479		<>	467		
480	type port BSFPort message {	=	468	type port BSFPort message {	
481	in HttpRequest;		469	in HttpRequest;	
482	out HttpResponse		470	out HttpResponse	
483	}		471	}	
484	}		472	}	
485		<>	473		
486	group externalFunctions {	=	474	group externalFunctions {	
487	external function fx_char2octet(in charstring p_string) return octetstring;		475	group externalFunctions {	
488	external function fx_charLow2octet(charstring p_string) return octetstring; //transformes c		476	external function fx_char2octet(in charstring p_string) return octetstring;	
489	//function to calculate the MD5 Message-Digest Algorithm according to RFC 1321	<>	477	external function fx_charLow2octet(charstring p_string) return octetstring; //transformes c	
490	external function fx_md5_hex(octetstring p_data) return octetstring;	=	478	//function to calculate the MD5 Message-Digest Algorithm according to RFC 1321	
491	external function fx_bitstring2Base64(in bitstring p_bitstring) return charstring;		479	external function fx_md5_hex(octetstring p_data) return octetstring;	
492	/**		480	external function fx_bitstring2Base64(in bitstring p_bitstring) return charstring;	
493	* @desc Converts the system time in the RFC 1123 format (e.g. 'Thu, 08 Jan 2004 10:23:17 C		481	/**	
494	* @param p_additionalDays system time + 'p_additionalDays' => return value		482	* @desc Converts the system time in the RFC 1123 format (e.g. 'Thu, 08 Jan 2004 10:23:17 C	
495	* @return date in RFC 1123 format		483	* @param p_additionalDays system time + 'p_additionalDays' => return value	
496	* @verdict	<>	484	* @return date in RFC 1123 format	
497	*/	=	485	* @verdict	
498	external function fx_date2RFC1123(in integer p_additionalDays) return charstring;		486	*/	
499	/**		487	external function fx_date2RFC1123(in integer p_additionalDays) return charstring;	
500	* @desc Converts the system time in the UTC format (e.g. '2007-07-24T13.20:00Z').		488	/**	
501	* @param p_additionalDays system time + 'p_additionalDays' => return value		489	* @desc Converts the system time in the UTC format (e.g. '2007-07-24T13.20:00Z').	
502	* @return date in UTC format		490	* @param p_additionalDays system time + 'p_additionalDays' => return value	
503	* @verdict	<>	491	* @return date in UTC format	
504	*/	=	492	* @verdict	
505	external function fx_date2UTC(in integer p_additionalDays) return charstring;		493	*/	
506	external function fx_bsfxml2Oct(in BootstrappingInfoType p_bsfxml) return octetstring;		494	external function fx_date2UTC(in integer p_additionalDays) return charstring;	
507	// change 2 (WK23): GBA authentication/authorization extensions for content protection test	+-	495	external function fx_bsfxml2Oct(in BootstrappingInfoType p_bsfxml) return octetstring;	
508	external function fx_sha_1_hex(octetstring p_data) return octetstring;				
509	// change 2 (WK23): GBA authentication/authorization extensions for content protection test				
510	external function fx_calculatingKDF4NAF(in octetstring p_keys, in octetstring p_kdfType,				
511	in				
512	in				
513	}	=	496	}	
514	}		497	}	

Datei: LibCommon_GBA_BSF_TypesAndValues.ttcn

1 //change 1 (WK18): for content protection tests 2 module LibCommon_GBA_BSF_TypesAndValues { 3 import from LibBCast_ServiceGuide_TypesAndValues { 4 group commonTypes 5 } 6 7 group BSFXMLMessage { 8 type record BootstrappingInfoType { 9 btidType btid, 10 lifetimeType lifeTime 11 } with {variant "namespacedef nsprefix = '' nsuri = 'uri:3gpp-gba'"} 12 13 type record btidType { 14 XmlElementStringValue text_ 15 } 16 17 type record lifetimeType { 18 XmlElementStringValue text_ 19 } 20 }	=	1 //change 1 (WK18): for content protection tests 2 module LibCommon_GBA_BSF_TypesAndValues { 3 import from LibBCast_ServiceGuide_TypesAndValues { 4 group commonTypes 5 } 6 7 group BSFXMLMessage { 8 type record BootstrappingInfoType { 9 btidType btid, 10 lifetimeType lifeTime 11 } with {variant "namespacedef nsprefix = '' nsuri = 'uri:3gpp-gba'"} 12 13 type record btidType { 14 XmlElementStringValue text_ 15 } 16 17 type record lifetimeType { 18 XmlElementStringValue text_ 19 } 20 }
21 22 // change 2 (WK23): GBA authentication/authorization extensions for content protection tests 23 group GBA { 24 type enumerated GBAType { 25 e_gba_u, 26 e_gba_me 27 } 28 }	+ -	
29 }	=	21 }

Datei: LibCommon_HTTP_TypesAndValues.ttcn

1 //change 1 (WK18): for content protection tests 2 module LibCommon_HTTP_TypesAndValues { 3 import from LibCommon_GBA_BSF_TypesAndValues all; 4 import from LibCommon_BSM_TypesAndValues all; 5 6 group HttpMessages { 7 type record HttpRequest { 8 charstring Method ("GET", "POST"), 9 charstring Uri, 10 charstring Version, 11 UriParameterList UriParameters optional, 12 HTTPHeaderValue Host, 13 HTTPHeader ContentType optional, 14 HTTPHeaderValue Authorization optional, 15 set of HttpHeader anotherHeaders optional, 16 HttpBody Body optional 17 } 18 19 type record HttpResponse { 20 charstring Version, 21 integer Code, 22 HTTPHeader ContentType optional, 23 set of HttpHeader anotherHeaders optional, 24 HttpBody Body optional 25 } 26 27 type union HttpBody { 28 octetstring values, 29 BootstrappingInfoType BootstrappingInfo, 30 mbmsSecurityRegisterType mbmsSecurityRegister, 31 mbmsSecurityRegisterResponseType mbmsSecurityRegisterResponse, 32 // change 1 (WK23): Service Request extensions for content protection tests 33 ServiceRequestType ServiceRequest, 34 // change 1 (WK23): Service Request extensions for content protection tests 35 ServiceResponseType ServiceResponse 36 } 37 38 group HttpHeaders { 39 type record HttpHeader { 40 charstring Name, 41 set length (1..infinity) of HttpHeaderValue ListOfValue 42 } 43 44 type record HttpHeaderValue { 45 charstring Value optional, 46 ListOfParameter ParameterList optional 47 } 48 49 type set of HttpHeaderValueParameter ListOfParameter; 50 51 type record HttpHeaderValueParameter { 52 charstring Name, 53 charstring Value optional 54 } 55 } 56 57 group HttpUriParameter { 58 type set of UriParameter UriParameterList; 59 60 type record UriParameter { 61 charstring Parameter, 62 charstring Value optional 63 } 64 } 65 } with { 66 encode "ExtHttpCodec" 67 } 68 }	=	1 //change 1 (WK18): for content protection tests 2 module LibCommon_HTTP_TypesAndValues { 3 import from LibCommon_GBA_BSF_TypesAndValues all; 4 import from LibCommon_BSM_TypesAndValues all; 5 6 group HttpMessages { 7 type record HttpRequest { 8 charstring Method ("GET", "POST"), 9 charstring Uri, 10 charstring Version, 11 UriParameterList UriParameters optional, 12 HTTPHeaderValue Host, 13 HTTPHeader ContentType optional, 14 HTTPHeaderValue Authorization optional, 15 set of HttpHeader anotherHeaders optional, 16 HttpBody Body optional 17 } 18 19 type record HttpResponse { 20 charstring Version, 21 integer Code, 22 HTTPHeader ContentType optional, 23 set of HttpHeader anotherHeaders optional, 24 HttpBody Body optional 25 } 26 27 type union HttpBody { 28 octetstring values, 29 BootstrappingInfoType BootstrappingInfo, 30 mbmsSecurityRegisterType mbmsSecurityRegister, 31 mbmsSecurityRegisterResponseType mbmsSecurityRegisterResponse 32 } 33 34 group HttpHeaders { 35 type record HttpHeader { 36 charstring Name, 37 set length (1..infinity) of HttpHeaderValue ListOfValue 38 } 39 40 type record HttpHeaderValue { 41 charstring Value optional, 42 ListOfParameter ParameterList optional 43 } 44 45 type set of HttpHeaderValueParameter ListOfParameter; 46 47 type record HttpHeaderValueParameter { 48 charstring Name, 49 charstring Value optional 50 } 51 } 52 53 group HttpUriParameter { 54 type set of UriParameter UriParameterList; 55 56 type record UriParameter { 57 charstring Parameter, 58 charstring Value optional 59 } 60 } 61 } with { 62 encode "ExtHttpCodec" 63 } 64 }
31 32 33 34 35	<>	31