



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

Linux-based Mobile Phone Middleware

Application Programming Interface

Circuit-Switched Communication Service

Document: CELF_MPP_CS_D_FR4

WARNING: This is a working draft for review only, it is NOT a published specification of the CE Linux Forum. It is likely that further substantial changes will be made in the course of review and issue resolution. Send comments on this version to:

MppApiComments@tree.celinuxforum.org

Revision History

Revision	Comment	Reviewer	Editor	Date
2.2	Initial	F2F meeting	NEC/Panasonic	05/09/28
2.2.1	Editorial Changes		AK	05/10/03
2.2.1a	NEC comments	NEC	AK	05/11/08
2.2.2	Review comments	Sharp	AK	05/11/20
2.2.2a	Template Change		AK	05/11/21
2.2.3	Appld ref removed Extract Common		AK	05/12/28
2.2.4	Review changes		AK	06/06/08
2.2.5	Clarifications Reformatting	NEC	AK	06/06/15
2.2.6 (FR3)	Minor reformatting, capitalization		Scott Preece	06/07/06
2.2.7 (FR4)	Bug Fixes after FR3 review		AK	11/01/06

25

26	0. INTRODUCTION	15
27	0.1 REFERENCES	15
28	0.1.1 Normative	15
29	0.1.2 Informative.....	15
30	1. PRIMITIVES.....	16
31	1.1 CONSTANTS.....	16
32	1.1.1 Line type (int).....	16
33	1.1.2 Dial Number	16
34	1.1.3 TAF address.....	16
35	1.1.4 Additional Service.....	16
36	1.2 ENUMS	18
37	1.2.1 Voice communication status (CelfMpCsComStatus)	18
38	1.2.2 Forwarding result (CelfMpCsFwResult)	19
39	1.2.3 Forwarding result details (CelfMpCsFwError).....	19
40	1.2.4 Bearer Type (CelfMpCsBtype).....	19
41	1.2.5 Call Reference Status (CelfMpCsCallRefStatus).....	19
42	1.2.6 Call Status (CelfMpCsChan)	20
43	1.2.7 Existence of continuation data (CelfMpCsContData)	20
44	1.2.8 Busy Tone sound flag (CelfMpCsBusyTone).....	20
45	1.2.9 Cause of Calling Line Identity (CLI) not available (CelfMpCsNoCLI).....	20
46	1.2.10 Dial number / Redirect number display indicator (CelfMpCsPrsntInd).....	21
47	1.2.11 Signal information (CelfMpCsSignal)	21
48	1.2.12 Originating Number notification (CelfMpCsNotice)	21
49	1.2.13 Line status (CelfMpCsLineStatus).....	21
50	1.2.14 Normal and emergency originating restriction (CelfMpCsLineRestrictData)	21
51	1.2.15 Receive level (CelfMpCsRSSILevel).....	22
52	1.2.16 Area status information (CelfMpCsLineCvrStatus).....	22
53	1.2.17 RRC mode (CelfMpCsLineRRCMode).....	22
54	1.2.18 Network identification information (CelfMpCsLineNetwork).....	22
55	1.2.19 Service status (CelfMpLineSrvStatus).....	22
56	1.2.20 Restriction status (CelfMpCsLineRestrict)	22
57	1.2.21 Identifying flag (CelfMpCsFlag).....	23
58	1.2.22 Notification Set (CelfMpCsNotifySet).....	23
59	1.2.23 Event Structure Category.....	23
60	1.2.24 Event Structure Subtype.....	23
61	1.2.25 DCF Event Set (CelfMpCsDCFSet).....	23
62	1.2.26 Voice message (CelfMpCsRecMsg).....	23
63	1.2.27 Off Hook Option (CelfMpCsOffHk).....	24
64	1.2.28 64K/AV Communication (CelfMpCsUDComStatus).....	24
65	1.2.29 AV Communication (CelfMpAVComStatus)	24
66	1.2.30 Receive Types (CelfMpCsRcvType)	25
67	1.2.31 Line Monitoring (CelfMpCsMtype)	25
68	1.2.32 Coverage Indicators (CelfMpCsCoverage)	25
69	1.2.33 Incoming Call Selection (CelfMpCallSelect).....	25
70	1.2.34 Optional Registered Number (CelfMpRegNum).....	25
71	1.2.35 Service Data (CelfMpCsSrvData).....	25
72	1.2.36 Reconnection Tone (CelfMpCsReconnectionTone)	25
73	1.2.37 Noise Canceling (CelfMpCsNoiseCancel).....	26
74	1.2.38 Call Quality Alarm (CelfMpCsCallQualAlarm).....	26
75	1.2.39 Connection Priority Setting (CelfMpCsHiPrioCom).....	26
76	1.2.40 Message Sound Settings (CelfMpCsVmSound).....	26
77	1.2.41 Incoming Call Auto Receive (CelfMpCsAutoRcv)	26

Classification: **Circuit-Switched Service**

78	1.3	DATA TYPES AND STRUCTURES	27
79	1.3.1	<i>Circuit switched status notification event structure</i>	27
80	1.3.2	<i>Call duration notification event structure</i>	27
81	1.3.3	<i>Disconnection cause notification event structure</i>	27
82	1.3.4	<i>Disconnection cause information structure</i>	27
83	1.3.5	<i>Forwarding result notification event structure</i>	28
84	1.3.6	<i>Forwarding result structure (CelfMpCsFwResult)</i>	28
85	1.3.7	<i>Off-hook transmission timeout event structure</i>	28
86	1.3.8	<i>Connection Destination Information (CelfMpConnectInfo)</i>	28
87	1.3.9	<i>Connection Request (CelfMpCsConReq)</i>	29
88	1.3.10	<i>Redirection number</i>	29
89	1.3.11	<i>Channel Number Information (CelfMpCsChanNum)</i>	29
90	1.3.12	<i>Channel not in use Flag</i>	29
91	1.3.13	<i>DCF Event Structure</i>	29
92	1.3.14	<i>Line status change notification event structure</i>	30
93	1.3.15	<i>Restriction display information structure (CelfMpCsResChgInf)</i>	30
94	1.3.16	<i>Receive level change notification event structure</i>	30
95	1.3.17	<i>Line Status structure (CelfMpCsAreaRefChgInf)</i>	30
96	1.3.18	<i>Additional service data structure (CelfMpCsAddSrvData)</i>	31
97	1.3.19	<i>Response Message Data Structure (CelfMpCsResponseMsgData)</i>	31
98	1.3.20	<i>Line Status Extension (CelfMpCsLineStatusEx)</i>	32
99	1.3.21	<i>Number of stored messages (CelfMpCsVMNum)</i>	32
100	1.3.22	<i>Date Format Structure (CelfMpCsDate)</i>	32
101	1.3.23	<i>Dial Buffer (CelfMpCsDialBuffer)</i>	32
102	1.3.24	<i>Dial Buffer Length (CelfMpCsDialLen)</i>	32
103	1.3.25	<i>Multi Party Operation (CelfMpCsMop)</i>	32
104	1.3.26	<i>Timer Value (CelfMpCsTimer)</i>	32
105	1.4	EVENTS TYPE.....	33
106	1.4.1	<i>DCF Event Type</i>	33
107	1.4.2	<i>CCP Notification type</i>	33
108	1.4.3	<i>Notification type</i>	35
109	1.4.4	<i>Restriction status</i>	35
110	1.5	STATUS CODES (CELFMPSTATUS).....	36
111	2.	START NOTIFICATION.....	37
112	2.1	SYMBOL: CELF_MP_CS_NOTIFICATION_START.....	37
113	2.1.1	<i>Syntax</i>	37
114	2.1.2	<i>Argument</i>	37
115	2.1.3	<i>Return Value</i>	38
116	2.1.4	<i>Include File</i>	38
117	2.1.5	<i>Functional Description</i>	38
118	3.	STOP NOTIFICATION.....	39
119	3.1	SYMBOL: CELF_MP_CS_NOTIFICATION_STOP.....	39
120	3.1.1	<i>Syntax</i>	39
121	3.1.2	<i>Argument</i>	39
122	3.1.3	<i>Return Value</i>	39
123	3.1.4	<i>Include File</i>	40
124	3.1.5	<i>Functional Description</i>	40
125	4.	GET VOICE COMMUNICATION STATUS	41
126	4.1	SYMBOL: CELF_MP_CS_GET_COM_STATUS.....	41
127	4.1.1	<i>Syntax</i>	41
128	4.1.2	<i>Argument</i>	41
129	4.1.3	<i>Return Value</i>	41

130	4.1.4	<i>Include File</i>	41
131	4.1.5	<i>Functional Description</i>	41
132	5.	GET CONNECTION INFORMATION TO OTHER PARTY	42
133	5.1	SYMBOL: CELF_MP_CS_GET_CON_INFO_REF	42
134	5.1.1	<i>Syntax</i>	42
135	5.1.2	<i>Argument</i>	42
136	5.1.3	<i>Return Value</i>	42
137	5.1.4	<i>Include File</i>	43
138	5.1.5	<i>Functional Description</i>	43
139	6.	GET CALL DURATION	44
140	6.1	SYMBOL: CELF_MP_CS_GET_CALL_DURATION	44
141	6.1.1	<i>Syntax</i>	44
142	6.1.2	<i>Argument</i>	44
143	6.1.3	<i>Return Value</i>	44
144	6.1.4	<i>Include File</i>	44
145	6.1.5	<i>Functional Description</i>	44
146	7.	OFF-HOOK NOTIFICATION	46
147	7.1	SYMBOL: CELF_MP_CS_NOTIFICATION_OFF_HOOK	46
148	7.1.1	<i>Syntax</i>	46
149	7.1.2	<i>Argument</i>	46
150	7.1.3	<i>Return Value</i>	46
151	7.1.4	<i>Include File</i>	47
152	7.1.5	<i>Functional Description</i>	47
153	8.	DISCONNECT	48
154	8.1	SYMBOL: CELF_MP_CS_DISCONNECT	48
155	8.1.1	<i>Syntax</i>	48
156	8.1.2	<i>Argument</i>	48
157	8.1.3	<i>Return Value</i>	48
158	8.1.4	<i>Include File</i>	48
159	8.1.5	<i>Functional Description</i>	48
160	9.	DIAL	50
161	9.1	SYMBOL: CELF_MP_CS_DIAL.....	50
162	9.1.1	<i>Syntax</i>	50
163	9.1.2	<i>Argument</i>	50
164	9.1.3	<i>Return Value</i>	51
165	9.1.4	<i>Include File</i>	51
166	9.1.5	<i>Functional Description</i>	51
167	10.	DIAL COMPLETE	52
168	10.1	SYMBOL: CELF_MP_CS_DIAL_END	52
169	10.1.1	<i>Syntax</i>	52
170	10.1.2	<i>Argument</i>	52
171	10.1.3	<i>Return Value</i>	52
172	10.1.4	<i>Include File</i>	52
173	10.1.5	<i>Functional Description</i>	52
174	11.	RESPONSE TO INCOMING CALL	54
175	11.1	SYMBOL: CELF_MP_CS_CALL_RCV	54
176	11.1.1	<i>Syntax</i>	54
177	11.1.2	<i>Argument</i>	54

178	11.1.3	<i>Return Value</i>	54
179	11.1.4	<i>Include File</i>	54
180	11.1.5	<i>Functional Description</i>	54
181	12.	FORWARD INCOMING CALL	56
182	12.1	SYMBOL: CELF_MP_CS_CALL_FORWARD	56
183	12.1.1	<i>Syntax</i>	56
184		<i>Argument</i>	56
185	12.1.2	56	56
186	12.1.3	<i>Return Value</i>	56
187	12.1.4	<i>Include File</i>	56
188	12.1.5	<i>Functional Description</i>	56
189	13.	FORWARD TO VOICE MAIL SYSTEM	57
190	13.1	SYMBOL: CELF_MP_CS_CALL_FORWARD_VOICE_MSG	57
191	13.1.1	<i>Syntax</i>	57
192		<i>Argument</i>	57
193	13.1.2	57	57
194	13.1.3	<i>Return Value</i>	57
195	13.1.4	<i>Include File</i>	57
196	13.1.5	<i>Functional Description</i>	57
197	14.	CALL HOLD	59
198	14.1	SYMBOL: CELF_MP_CS_CALL_HOLD	59
199	14.1.1	<i>Syntax</i>	59
200		<i>Argument</i>	59
201	14.1.2	59	59
202	14.1.3	<i>Return Value</i>	59
203	14.1.4	<i>Include File</i>	59
204	14.1.5	<i>Functional Description</i>	59
205	15.	CALL REJECT	61
206	15.1	SYMBOL: CELF_MP_CS_CALL_REJECT	61
207	15.1.1	<i>Syntax</i>	61
208		<i>Argument</i>	61
209	15.1.2	61	61
210	15.1.3	<i>Return Value</i>	61
211	15.1.4	<i>Include File</i>	61
212	15.1.5	<i>Functional Description</i>	61
213	16.	MULTI PARTY CALL	63
214	16.1	SYMBOL: CELF_MP_CS_MP_CALL	63
215	16.1.1	<i>Syntax</i>	63
216		<i>Argument</i>	63
217	16.1.2	63	63
218	16.1.3	<i>Return Value</i>	64
219	16.1.4	<i>Include File</i>	64
220	16.1.5	<i>Functional Description</i>	64
221	17.	ON-HOOK ORIGINATING	66
222	17.1	SYMBOL: CELF_MP_CS_ORIGINATING_ON_HOOK	66
223	17.1.1	<i>Syntax</i>	66
224		<i>Argument</i>	66
225	17.1.2	66	66
226	17.1.3	<i>Return Value</i>	66

227	17.1.4	<i>Include File</i>	66
228	17.1.5	<i>Functional Description</i>	67
229	18.	GET CALL REFERENCE	68
230	18.1	SYMBOL: CELF_MP_CS_GET_CALL_REFERENCE.....	68
231	18.1.1	<i>Syntax</i>	68
232	18.1.2	<i>Argument</i>	68
233	18.1.3	<i>Return Value</i>	68
234	18.1.4	<i>Include File</i>	68
235	18.1.5	<i>Functional Description</i>	68
236	19.	START DCF MESSAGE NOTIFICATION	70
237	19.1	SYMBOL: CELF_MP_CS_DCF_NOTIFICATION_START.....	70
238	19.1.1	<i>Syntax</i>	70
239	19.1.2	<i>Argument</i>	70
240	19.1.3	<i>Return Value</i>	71
241	19.1.4	<i>Include File</i>	71
242	19.1.5	<i>Functional Description</i>	71
243	20.	STOP DCF MESSAGE NOTIFICATION	73
244	20.1	SYMBOL: CELF_MP_CS_DCF_NOTIFICATION_STOP.....	73
245	20.1.1	<i>Syntax</i>	73
246	20.1.2	<i>Argument</i>	73
247	20.1.3	<i>Return Value</i>	73
248	20.1.4	<i>Include File</i>	74
249	20.1.5	<i>Functional Description</i>	74
250	21.	VOICE MESSAGE NOTIFICATION	75
251	21.1	SYMBOL: CELF_MP_CS_VOICE_MSG_NOTIFY.....	75
252	21.1.1	<i>Syntax</i>	75
253		<i>Argument</i>	75
254	21.1.2	75	
255	21.1.3	<i>Return Value</i>	75
256	21.1.4	<i>Include File</i>	75
257	21.1.5	<i>Functional Description</i>	75
258	22.	HOLD TONE START	76
259	22.1	SYMBOL: CELF_MP_CS_HOLD_TONE_START.....	76
260	22.1.1	<i>Syntax</i>	76
261		<i>Argument</i>	76
262	22.1.2	76	
263	22.1.3	<i>Return Value</i>	76
264	22.1.4	<i>Include File</i>	76
265	22.1.5	<i>Functional Description</i>	76
266	23.	HOLD TONE STOP	77
267	23.1	SYMBOL: CELF_MP_CS_HOLD_TONE_STOP.....	77
268	23.1.1	<i>Syntax</i>	77
269		<i>Argument</i>	77
270	23.1.2	77	
271	23.1.3	<i>Return Value</i>	77
272	23.1.4	<i>Include File</i>	77
273	23.1.5	<i>Functional Description</i>	77
274	24.	GET 64K / AV COMMUNICATION STATUS	78

275	24.1	SYMBOL: CELF_MP_CS_GET_UD_COM_STAT	78
276	24.1.1	<i>Syntax</i>	78
277		<i>Argument</i>	78
278	24.1.2	78	
279	24.1.3	<i>Return Value</i>	78
280	24.1.4	<i>Include File</i>	78
281	24.1.5	<i>Functional Description</i>	78
282	25.	GET INTERNAL/EXTERNAL AV COMMUNICATION STATUS.....	79
283	25.1	SYMBOL: CELF_MP_CS_GET_AV_COM_STAT	79
284	25.1.1	<i>Syntax</i>	79
285	25.1.2	<i>Argument</i>	79
286	25.1.3	<i>Return Value</i>	79
287	25.1.4	<i>Include File</i>	80
288	25.1.5	<i>Functional Description</i>	80
289	26.	GET COMMUNICATION STATUS.....	81
290	26.1	SYMBOL: CELF_MP_CS_GET_COM_STAT	81
291	26.1.1	<i>Syntax</i>	81
292	26.1.2	<i>Argument</i>	81
293	26.1.3	<i>Return Value</i>	82
294	26.1.4	<i>Include File</i>	82
295	26.1.5	<i>Functional Description</i>	82
296	27.	START LINE STATUS NOTIFICATION.....	83
297	27.1	SYMBOL: CELF_MP_CS_LINE_STATUS_NOTIFICATION_START.....	83
298	27.1.1	<i>Syntax</i>	83
299	27.1.2	<i>Argument</i>	83
300	27.1.3	<i>Return Value</i>	84
301	27.1.4	<i>Include File</i>	84
302	27.1.5	<i>Functional Description</i>	84
303	28.	STOP LINE STATUS NOTIFICATION	85
304	28.1	SYMBOL: CELF_MP_CS_LINE_STATUS_NOTIFICATION_STOP.....	85
305	28.1.1	<i>Syntax</i>	85
306	28.1.2	<i>Argument</i>	85
307	28.1.3	<i>Return Value</i>	85
308	28.1.4	<i>Include File</i>	86
309	28.1.5	<i>Functional Description</i>	86
310	29.	GET RECEPTION LEVEL	87
311	29.1	SYMBOL: CELF_MP_CS_GET_RECEPTION_LEVEL.....	87
312	29.1.1	<i>Syntax</i>	87
313	29.1.2	<i>Argument</i>	87
314	29.1.3	<i>Return Value</i>	87
315	29.1.4	<i>Include File</i>	87
316	29.1.5	<i>Functional Description</i>	87
317	30.	GET LINE STATUS	88
318	30.1	SYMBOL: CELF_MP_CS_GET_LINE_STATUS.....	88
319	30.1.1	<i>Syntax</i>	88
320	30.1.2	<i>Argument</i>	88
321	30.1.3	<i>Return Value</i>	88
322	30.1.4	<i>Include File</i>	88
323	30.1.5	<i>Functional Description</i>	88

324	31. GET COVERAGE STATUS	89
325	31.1 SYMBOL: CELF_MP_CS_GET_COVERAGE_STATUS.....	89
326	31.1.1 Syntax.....	89
327	31.1.2 Argument.....	89
328	31.1.3 Return Value.....	89
329	31.1.4 Include File.....	89
330	31.1.5 Functional Description.....	90
331	32. GET VOICE MAIL INFORMATION	91
332	32.1 SYMBOL: CELF_MP_CS_GET_VM_INFO.....	91
333	32.1.1 Syntax.....	91
334	32.1.2 Argument.....	91
335	32.1.3 Return Value.....	91
336	32.1.4 Include File.....	91
337	32.1.5 Functional Description.....	91
338	33. SET VOICE MAIL INFORMATION	92
339	33.1 SYMBOL: CELF_MP_CS_SET_VM_INFO.....	92
340	33.1.1 Syntax.....	92
341	33.1.2 Argument.....	92
342	33.1.3 Return Value.....	92
343	33.1.4 Include File.....	92
344	33.1.5 Functional Description.....	92
345	34. GET CALL SELECTION	93
346	34.1 SYMBOL: CELF_MP_CS_GET_CALL_SELECT.....	93
347	34.1.1 Syntax.....	93
348	34.1.2 Argument.....	93
349	34.1.3 Return Value.....	93
350	34.1.4 Include File.....	93
351	34.1.5 Functional Description.....	93
352	35. SET CALL SELECTION	94
353	35.1 SYMBOL: CELF_MP_CS_SET_CALL_SELECT.....	94
354	35.1.1 Syntax.....	94
355	35.1.2 Argument.....	94
356	35.1.3 Return Value.....	94
357	35.1.4 Include File.....	94
358	35.1.5 Functional Description.....	94
359	36. SET SERVICE INFORMATION	95
360	36.1 SYMBOL: CELF_MP_CS_SET_SERVICE_INFO.....	95
361	36.1.1 Syntax.....	95
362	36.1.2 Argument.....	95
363	36.1.3 Return Value.....	95
364	36.1.4 Include File.....	95
365	36.1.5 Functional Description.....	95
366	37. GET SERVICE INFORMATION	97
367	37.1 SYMBOL: CELF_MP_CS_GET_SERVICE_INFO.....	97
368	37.1.1 Syntax.....	97
369	37.1.2 Argument.....	97
370	37.1.3 Return Value.....	97
371	37.1.4 Include File.....	97
372	37.1.5 Functional Description.....	97

373	38. DELETE SERVICE INFORMATION.....	99
374	38.1 SYMBOL: CELF_MP_CS_DEL_SERVICE_INFO	99
375	38.1.1 Syntax.....	99
376	38.1.2 Argument.....	99
377	38.1.3 Return Value	99
378	38.1.4 Include File.....	99
379	38.1.5 Functional Description.....	99
380	39. REMOVE SERVICE INFORMATION.....	100
381	39.1 SYMBOL: CELF_MP_CS_REMOVE_ALL_SERVICE_INFO	100
382	39.1.1 Syntax.....	100
383	39.1.2 Argument.....	100
384	39.1.3 Return Value	100
385	39.1.4 Include File.....	100
386	39.1.5 Functional Description.....	100
387	40. SET RESPONSE MESSAGE SETTINGS	101
388	40.1 SYMBOL: CELF_MP_CS_SET_RESP_MSG	101
389	40.1.1 Syntax.....	101
390	40.1.2 Argument.....	101
391	40.1.3 Return Value	101
392	40.1.4 Include File.....	101
393	40.1.5 Functional Description.....	101
394	41. GET RESPONSE MESSAGE SETTINGS	103
395	41.1 SYMBOL: CELF_MP_CS_GET_RESP_MSG.....	103
396	41.1.1 Syntax.....	103
397	41.1.2 Argument.....	103
398	41.1.3 Return Value	103
399	41.1.4 Include File.....	103
400	41.1.5 Functional Description.....	103
401	42. DELETE RESPONSE MESSAGE SETTINGS.....	105
402	42.1 SYMBOL: CELF_MP_CS_DEL_RESP_MSG.....	105
403	42.1.1 Syntax.....	105
404	42.1.2 Argument.....	105
405	42.1.3 Return Value	105
406	42.1.4 Include File.....	105
407	42.1.5 Functional Description.....	105
408	43. REMOVE ALL RESPONSE MESSAGE SETTINGS.....	106
409	43.1 SYMBOL: CELF_MP_CS_REMOVE_ALL_RESP_MSG	106
410	43.1.1 Syntax.....	106
411	43.1.2 Argument.....	106
412	43.1.3 Return Value	106
413	43.1.4 Include File.....	106
414	43.1.5 Functional Description.....	106
415	44. SET RECONNECTION TONE	107
416	44.1 SYMBOL: CELF_MP_CS_SET_RECONNECTION_TONE	107
417	44.1.1 Syntax.....	107
418	44.1.2 Argument.....	107
419	44.1.3 Return Value	107
420	44.1.4 Include File.....	107
421	44.1.5 Functional Description.....	107

422	45. GET RECONNECTION TONE	108
423	45.1 SYMBOL: CELF_MP_CS_GET_RECONNECTION_TONE	108
424	45.1.1 <i>Syntax</i>	108
425	45.1.1 <i>Argument</i>	108
426	45.1.2 <i>108</i>	
427	45.1.3 <i>Return Value</i>	108
428	45.1.4 <i>Include File</i>	108
429	45.1.5 <i>Functional Description</i>	108
430	46. GET NOISE CANCEL	109
431	46.1 SYMBOL: CELF_MP_CS_GET_NOISE_CANCEL	109
432	46.1.1 <i>Syntax</i>	109
433	46.1.2 <i>Argument</i>	109
434	46.1.3 <i>Return Value</i>	109
435	46.1.4 <i>Include File</i>	109
436	46.1.5 <i>Functional Description</i>	109
437	47. SET NOISE CANCEL	110
438	47.1 SYMBOL: CELF_MP_CS_SET_NOISE_CANCEL.....	110
439	47.1.1 <i>Syntax</i>	110
440	47.1.2 <i>Argument</i>	110
441	47.1.3 <i>Return Value</i>	110
442	47.1.4 <i>Include File</i>	110
443	47.1.5 <i>Functional Description</i>	110
444	48. GET CALL QUALITY ALARM.....	111
445	48.1 SYMBOL: CELF_MP_CS_GET_CALL_QUALITY_ALARM	111
446	48.1.1 <i>Syntax</i>	111
447	48.1.2 <i>Argument</i>	111
448	48.1.3 <i>Return Value</i>	111
449	48.1.4 <i>Include File</i>	111
450	48.1.5 <i>Functional Description</i>	111
451	49. SET CALL QUALITY ALARM.....	112
452	49.1 SYMBOL: CELF_MP_CS_SET_CALL_QUALITY_ALARM.....	112
453	49.1.1 <i>Syntax</i>	112
454	49.1.2 <i>Argument</i>	112
455	49.1.3 <i>Return Value</i>	112
456	49.1.4 <i>Include File</i>	112
457	49.1.5 <i>Functional Description</i>	112
458	50. GET NOISE CANCEL PERMIT.....	113
459	50.1 SYMBOL: CELF_MP_CS_GET_NOISE_CANCEL_PERMIT	113
460	50.1.1 <i>Syntax</i>	113
461	50.1.2 <i>Argument</i>	113
462	50.1.3 <i>Return Value</i>	113
463	50.1.4 <i>Include File</i>	113
464	50.1.5 <i>Functional Description</i>	113
465	51. GET HIGH PRIORITY COMMUNICATION MODE.....	114
466	51.1 SYMBOL: CELF_MP_CS_GET_HI_PRIO_COM	114
467	51.1.1 <i>Syntax</i>	114
468	51.1.2 <i>Argument</i>	114
469	51.1.3 <i>Return Value</i>	114
470	51.1.4 <i>Include File</i>	114

471	51.1.5	<i>Functional Description</i>	114
472	52.	SET HIGH PRIORITY COMMUNICATION MODE	115
473	52.1	SYMBOL: CELF_MP_CS_SET_HI_PRIO_COM.....	115
474	52.1.1	<i>Syntax</i>	115
475	52.1.2	<i>Argument</i>	115
476	52.1.3	<i>Return Value</i>	115
477	52.1.4	<i>Include File</i>	115
478	52.1.5	<i>Functional Description</i>	115
479	53.	GET PHONE ANSWERING SOUND ACTIVATION	116
480	53.1	SYMBOL: CELF_MP_CS_GET_VM_SOUND_STATUS.....	116
481	53.1.1	<i>Syntax</i>	116
482	53.1.2	<i>Argument</i>	116
483	53.1.3	<i>Return Value</i>	116
484	53.1.4	<i>Include File</i>	116
485	53.1.5	<i>Functional Description</i>	116
486	54.	SET PHONE ANSWERING SOUND ACTIVATION	117
487	54.1	SYMBOL: CELF_MP_CS_SET_VM_SOUND_STATUS.....	117
488	54.1.1	<i>Syntax</i>	117
489	54.1.2	<i>Argument</i>	117
490	54.1.3	<i>Return Value</i>	117
491	54.1.4	<i>Include File</i>	117
492	54.1.5	<i>Functional Description</i>	117
493	55.	GET AUTOMATIC RECEIVE STATUS	118
494	55.1	SYMBOL: CELF_MP_CS_GET_AUTO_RCV_STATUS.....	118
495	55.1.1	<i>Syntax</i>	118
496	55.1.2	<i>Argument</i>	118
497	55.1.3	<i>Return Value</i>	118
498	55.1.4	<i>Include File</i>	118
499	55.1.5	<i>Functional Description</i>	118
500	56.	SET AUTOMATIC RECEIVE STATUS	119
501	56.1	SYMBOL: CELF_MP_CS_SET_AUTO_RCV_STATUS.....	119
502	56.1.1	<i>Syntax</i>	119
503	56.1.2	<i>Argument</i>	119
504	56.1.3	<i>Return Value</i>	119
505	56.1.4	<i>Include File</i>	119
506	56.1.5	<i>Functional Description</i>	119
507	57.	GET AUTOMATIC TIMER	120
508	57.1	SYMBOL: CELF_MP_CS_GET_AUTO_TIMER.....	120
509	57.1.1	<i>Syntax</i>	120
510	57.1.2	<i>Argument</i>	120
511	57.1.3	<i>Return Value</i>	120
512	57.1.4	<i>Include File</i>	120
513	57.1.5	<i>Functional Description</i>	120
514	58.	SET AUTOMATIC TIMER	121
515	58.1	SYMBOL: CELF_MP_CS_SET_AUTO_TIMER.....	121
516	58.1.1	<i>Syntax</i>	121
517	58.1.2	<i>Argument</i>	121
518	58.1.3	<i>Return Value</i>	121

519	58.1.4	<i>Include File</i>	121
520	58.1.5	<i>Functional Description</i>	121
521	59.	GET RESET DATE	122
522	59.1	SYMBOL: CELF_MP_CS_GET_RESET_DATE.....	122
523	59.1.1	<i>Syntax</i>	122
524	59.1.2	<i>Argument</i>	122
525	59.1.3	<i>Return Value</i>	122
526	59.1.4	<i>Include File</i>	122
527	59.1.5	<i>Functional Description</i>	122
528	60.	SET RESET DATE	123
529	60.1	SYMBOL: CELF_MP_CS_SET_RESET_DATE	123
530	60.1.1	<i>Syntax</i>	123
531	60.1.2	<i>Argument</i>	123
532	60.1.3	<i>Return Value</i>	123
533	60.1.4	<i>Include File</i>	123
534	60.1.5	<i>Functional Description</i>	123
535	61.	GET CALL SILENT TIME	124
536	61.1	SYMBOL: CELF_MP_CS_GET_CALL_SILENT_TIME.....	124
537	61.1.1	<i>Syntax</i>	124
538	61.1.2	<i>Argument</i>	124
539	61.1.3	<i>Return Value</i>	124
540	61.1.4	<i>Include File</i>	124
541	61.1.5	<i>Functional Description</i>	124
542	62.	SET CALL SILENT TIME	125
543	62.1	SYMBOL: CELF_MP_CS_SET_CALL_SILENT_TIME.....	125
544	62.1.1	<i>Syntax</i>	125
545	62.1.2	<i>Argument</i>	125
546	62.1.3	<i>Return Value</i>	125
547	62.1.4	<i>Include File</i>	125
548	62.1.5	<i>Functional Description</i>	125
549	63.	GET CALL RECORDED	126
550	63.1	SYMBOL: CELF_MP_CS_GET_CALL_RECORDED.....	126
551	63.1.1	<i>Syntax</i>	126
552	63.1.2	<i>Argument</i>	126
553	63.1.3	<i>Return Value</i>	126
554	63.1.4	<i>Include File</i>	126
555	63.1.5	<i>Functional Description</i>	126
556	64.	SET CALL RECORDED	127
557	64.1	SYMBOL: CELF_MP_CS_SET_CALL_RECORDED	127
558	64.1.1	<i>Syntax</i>	127
559	64.1.2	<i>Argument</i>	127
560	64.1.3	<i>Return Value</i>	127
561	64.1.4	<i>Include File</i>	127
562	64.1.5	<i>Functional Description</i>	127
563	65.	SET RADIO	128
564	65.1	SYMBOL: CELF_MP_CS_SET_RADIO	128
565	65.1.1	<i>Syntax</i>	128
566	65.1.2	<i>Argument</i>	128

567	65.1.3	<i>Return Value</i>	128
568	65.1.4	<i>Include File</i>	128
569	65.1.5	<i>Functional Description</i>	128
570	66.	GET RADIO STATUS	129
571	66.1	SYMBOL: CELF_MP_CS_GET_RADIO	129
572	66.1.1	<i>Syntax</i>	129
573	66.1.2	<i>Argument</i>	129
574	66.1.3	<i>Return Value</i>	129
575	66.1.4	<i>Include File</i>	129
576	66.1.5	<i>Functional Description</i>	129
577			

DRAFT

578 **0. Introduction**

579 Circuit-Switched Communication Service (CS Service) has the function of the call control, the call
580 state management, the tone control and the log processing.

581 Circuit-Switched Communication Service includes

582 Voice communication service,

583 Video communication service, and

584 Digital data communication service.

585

586 **0.1 References**

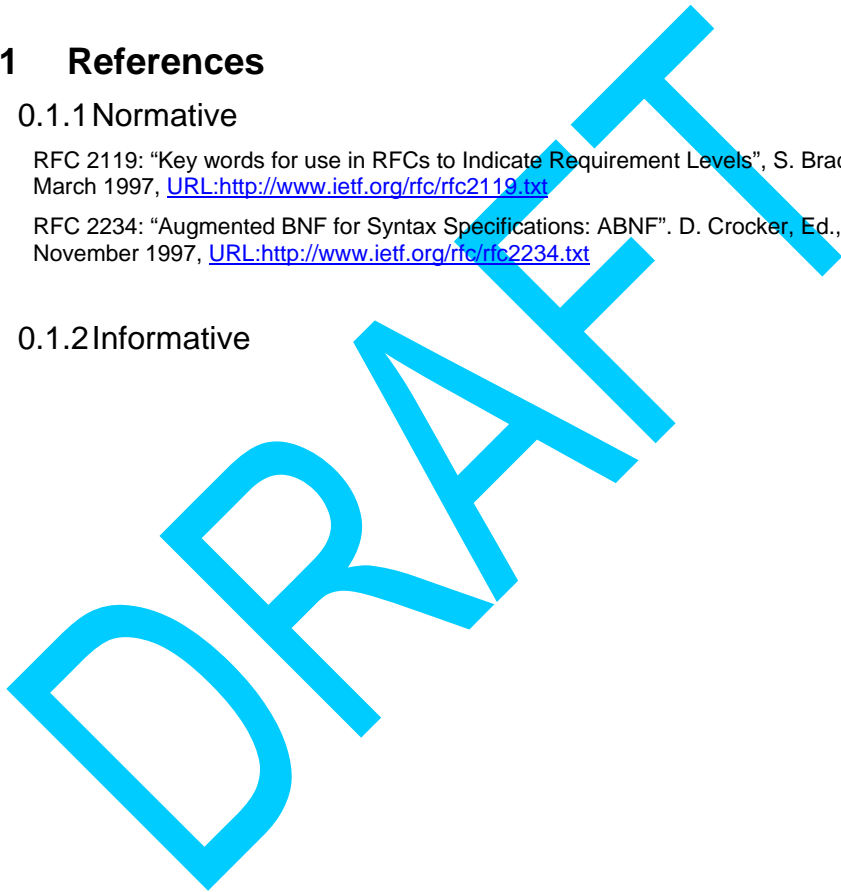
587 **0.1.1 Normative**

588 RFC 2119: "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner,
589 March 1997, [URL:http://www.ietf.org/rfc/rfc2119.txt](http://www.ietf.org/rfc/rfc2119.txt)

590 RFC 2234: "Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell.
591 November 1997, [URL:http://www.ietf.org/rfc/rfc2234.txt](http://www.ietf.org/rfc/rfc2234.txt)

592

593 **0.1.2 Informative**



594 **1. Primitives**

595 This section contains the definitions of the data types and constants used in the interfaces of this
596 service.

597 **1.1 Constants**

598 **1.1.1 Line type (int)**

599	CELF_MP_CS_LINE_CDMA1	CDMA1
600	CELF_MP_CS_LINE_UMTS	UMTS
601	CELF_MP_CS_LINE_FOMA	FOMA
602	CELF_MP_CS_LINE_GSM	GSM
603	CELF_MP_CS_LINE_WLAN	WLAN
604	CELF_MP_CS_LINE_BLUET	BLUET

Comment [AK1]: Add other types and change to enum

Comment [AK2]: An API to set/change the Line would be nice

605 The constants denote several different forms of wireless connectivity. Currently the first 3
606 are supported by this specification.

607

608 **1.1.2 Dial Number**

609 Dial number length of the other party
610 This data is valid when this mobile phone originates a call.

611 **Definition:**

612	CELF_MP_CS_DIAL_MAX	45
-----	---------------------	----

613

614 **1.1.3 TAF address**

615 TAF address is the connection ID in TAF (Terminal Adaptation Function).
616 This is external to the CELF MPP specification.
617 See the following documents for further details: 3GPP TS 27.001 / 3GPP TS 24.002

618

619 **1.1.4 Additional Service**

Comment [AK3]: Supplementary to Additional change!

620 Service Info Name Length

621 **Definition:**

622	CELF_SRVINFO_TITLE	21
-----	--------------------	----

623

624 Dial data length for accessing the service

625 **Definition:**

626	CELF_SRVINFO_DATA	40
-----	-------------------	----

627

628 Response Message Name Length

629 **Definition:**
630 CELF_RESMSG_TITLE 21
631
632 Response data length for accessing the service
633 **Definition:**
634 CELF_RESMSG_DATA 40

DRAFT

635 **1.2 Enums**

636 **1.2.1 Voice communication status (CelfMpCsComStatus)**

637 In this operation mode, following multiple calls can be handled by a mobile phone,
638 simultaneously. Each state of handling calls is:

- 639 • Holding a receiving call
- 640 • Talk over the phone
- 641 • Receive another incoming call

642 The number of handling states is three.

643 This is called the multiple calls.

644

645 In case that one call is AV call, the mobile phone handles this call only.

646

647 **1.2.1.1 Condition: only one call**

- 648 CELF_MP_CS_COM_STATUS_WAIT : Standby
- 649 CELF_MP_CS_COM_STATUS_IN : Incoming call
- 650 CELF_MP_CS_COM_STATUS_OUT : Outgoing call
- 651 CELF_MP_CS_COM_STATUS_CALL_ALERT : Calling
- 652 CELF_MP_CS_COM_STATUS_TALK : In conversation
- 653 CELF_MP_CS_COM_STATUS_HOLD : In hold

654 This status is

655 (a) that incoming call was received, and

656 (b) that this incoming call cannot transit to conversation status because of the mobile
657 phone.

658 CELF_MP_CS_COM_STATUS_DISCONNECT : Disconnecting

659

660 **1.2.1.2 Condition: two call**

661 One call is in conversation, and another call is in some status.

- 662 CELF_MP_CS_COM_STATUS_2ND_IN : In conversation and incoming
- 663 CELF_MP_CS_COM_STATUS_2ND_OUT : In conversation and outgoing
- 664 CELF_MP_CS_COM_STATUS_2ND_TALK : In conversation and calling
- 665 CELF_MP_CS_COM_STATUS_2ND_HOLD : In conversation and hold
- 666 CELF_MP_CS_COM_STATUS_2ND_DISCONNECT : In conversation and
667 disconnecting

668 - three call One call is in conversation, another call is in hold, and 3rd call is in
669 incoming.

670 CELF_MP_CS_COM_STATUS_2ND_HOLD_IN : In conversation, hold, and incoming

Comment [AK4]: Change to generic plus call type info

671 **1.2.1.3 Condition: only one AV call**
672 CELF_MP_CS_COM_STATUS_AV_IN : Incoming AV call
673 CELF_MP_CS_COM_STATUS_AV_OUT : Outgoing AV call
674 CELF_MP_CS_COM_STATUS_AV_CALL_ALERT : Calling AV call
675 CELF_MP_CS_COM_STATUS_AV_TALK : In conversation
676 CELF_MP_CS_COM_STATUS_AV_HOLD : In hold
677 CELF_MP_CS_COM_STATUS_AV_DISCONNECT : Disconnecting an AV call

678 Other combinations of voice communication calls are not defined.

679

680 **1.2.2 Forwarding result (CelfMpCsFwResult)**

681 CELF_MP_CS_OK : Successful forwarding
682 CELF_MP_CS_ERR : Forwarding failure

683

684 **1.2.3 Forwarding result details (CelfMpCsFwError)**

685 Set only at forwarding failure.

686 CELF_MP_CS_FW_ERROR_NO_JOIN : Service is not contracted.
687 CELF_MP_CS_FW_ERROR_NO_SETDATA : the forwarded destination is not
688 registered.
689 CELF_MP_CS_FW_ERROR_ETC : Others

690

691 **1.2.4 Bearer Type (CelfMpCsBtype)**

692 CELF_MP_CS_BTYPE_NONE : None (unfixed)
693 CELF_MP_CS_BTYPE_ANY : Not Specified
694 CELF_MP_CS_BTYPE_VOICE : Voice
695 CELF_MP_CS_BTYPE_UD32 : 32K communication
696 CELF_MP_CS_BTYPE_UD64 : 64K communication
697 CELF_MP_CS_BTYPE_AV32 : 32K communication
698 CELF_MP_CS_BTYPE_AV64 : 64K communication

699

700 **1.2.5 Call Reference Status (CelfMpCsCallRefStatus)**

701 CELF_MP_CS_USED : Valid "CN_No" – Connection Number entry.
702 CELF_MP_CS_UNUSED : "CN_No" – Connection Number is not used.

703 In some cases the Call reference status is unused, indicated by CELF_MP_CS_UNUSED.
704 If so, there is no connection between this mobile phone and other party and all data is void.

705

706 **1.2.6 Call Status (CelfMpCsChan)**

707 Call status for this mobile phone

- 708 CELF_MP_CS_CHAN_NULL : Vacant
- 709 CELF_MP_CS_CHAN_OFFHK : Off-hook
- 710 CELF_MP_CS_CHAN_OUT : Outgoing call
- 711 CELF_MP_CS_CHAN_CALL_ALERT : Calling
- 712 CELF_MP_CS_CHAN_IN : Incoming call
- 713 CELF_MP_CS_CHAN_REQ_IN : Response (conversation)
- 714 (The status of responding mobile phone is in conversation.)
- 715 CELF_MP_CS_CHAN_TALK : In conversation
- 716 CELF_MP_CS_CHAN_REQ_HOLD : Response (hold)
- 717 (The status of responding mobile phone is in hold.)
- 718 CELF_MP_CS_CHAN_HOLD : Hold response
- 719 CELF_MP_CS_CHAN_2ND_HOLD : In conversation and hold
- 720 CELF_MP_CS_CHAN_DISCONNECT : Disconnecting

721

722 **1.2.7 Existence of continuation data (CelfMpCsContData)**

723 CELF_MP_CS_ON : valid below data

724 CELF_MP_CS_OFF : non valid below data

725 The following data entries in 1.3.8, from "Calling_Dial" to "cause", is valid data if the call
726 status is incoming or conversation and incoming call.

727

728 **1.2.8 Busy Tone sound flag (CelfMpCsBusyTone)**

729 Whether Busy Tone (engaged tone) sounds for this phone, or not

730 CELF_MP_CS_BUSY_TONE_ON : Busy Tone sounding.

731 CELF_MP_CS_BUSY_TONE_OFF : Busy Tone not sounding.

732

733 **1.2.9 Cause of Calling Line Identity (CLI) not available**
734 **(CelfMpCsNoCLI)**

735 The reason why the dial number of the other party is not notified.

736 The dial number of the other party is in "Calling dial" or "Called dial" status.

737 CELF_MP_CS_NOCLI_NOSRV : service is not supported.

738 CELF_MP_CS_NOCLI_BLOCKED : user blocked the display.

739 CELF_MP_CS_NOCLI_CONFLICT : service conflicts.

740 CELF_MP_CS_NOCLI_PUBLICPHONE : origination is from a public phone.

741 This data is valid, when the unsigned char num_presentation_indicator is filled.

742 1.2.10 Dial number / Redirect number display indicator
743 (CelfMpCsPrsntInd)

744 Whether dial number / redirection number of the other party can be displayed, or not.

745 CELF_MP_CS_PRSNT_IND_ALLOWED : Displayable

746 CELF_MP_CS_PRSNT_IND_RESTRICTED : Impossible to display

747 CELF_MP_CS_PRSNT_IND_NOT_AVAILABLE : Displayable number does not exist.

748

749 1.2.11 Signal information (CelfMpCsSignal)

750 The type of tone of this phone

751 CELF_MP_CS_SIGNAL_DIAL_TONE_ON : Dial tone on

752 CELF_MP_CS_SIGNAL_RINGBACK_TONE_ON : Ring back tone on

753 CELF_MP_CS_SIGNAL_INTERCEPT_TONE_ON : Intercept tone on

754 CELF_MP_CS_SIGNAL_NW_CONGESTION_TONE_ON : Network congestion tone on

755 CELF_MP_CS_SIGNAL_BUSY_TONE_ON : Busy tone on

756 CELF_MP_CS_SIGNAL_CONFIRM_TONE_ON : Confirm tone on

757 CELF_MP_CS_SIGNAL_ANSWER_TONE_ON : Answer tone on

758 CELF_MP_CS_SIGNAL_CALLWAITING_TONE_ON : Call waiting tone on

759 CELF_MP_CS_SIGNAL_OFFHK_WARNING_TONE_ON : Off-hook warning tone on

760 CELF_MP_CS_SIGNAL_TONES_OFF : Tones off

761 CELF_MP_CS_SIGNAL_ALERTING_OFF : Alerting off

762 CELF_MP_CS_SIGNAL_NOTSET : Signal information is not set.

763

764 1.2.12 Originating Number notification (CelfMpCsNotice)

765 Whether the originating dial number is notified or not.

766 CELF_MP_CS_NOTICE_ON : Notified

767 CELF_MP_CS_NOTICE_OFF : Not notified

768 CELF_MP_CS_NOTICE_NOSET : Not set

769

770 1.2.13 Line status (CelfMpCsLineStatus)

771 CELF_MP_CS_LINE_STATUS_OUT : Out-of-communication area

772 CELF_MP_CS_LINE_STATUS_IN : Within-communication area

773

774 1.2.14 Normal and emergency originating restriction
775 (CelfMpCsLineRestrictData)

776 CELF_MP_CS_LINE_RESTRICT_DATA_ON : With originating restriction

777 CELF_MP_CS_LINE_RESTRICT_DATA_OFF : Without originating restriction

778

779 1.2.15 Receive level (CelfMpCsRSSILevel)

780 CELF_MP_CS_RSSI_LEVEL_NONE : No reception

781 CELF_MP_CS_RSSI_LEVEL_LOW : Receive level (Lowest)

782 CELF_MP_CS_RSSI_LEVEL_MEDIUM1 : Receive level

783 CELF_MP_CS_RSSI_LEVEL_MEDIUM2 : Receive level

784 CELF_MP_CS_RSSI_LEVEL_HIGH : Receive level (Highest)

785

786 1.2.16 Area status information (CelfMpCsLineCvrStatus)

787 CELF_MP_CS_LINE_CVR_STATUS_IN : Inside the area

788 CELF_MP_CS_LINE_CVR_STATUS_OUT : Outside the area

789

790 1.2.17 RRC mode (CelfMpCsLineRRCMode)

791 CELF_MP_CS_LINE_RRC_MODE_IDLE : idle-mode

792 CELF_MP_CS_LINE_RRC_MODE_UTRAN : utran-connected-mode

793

794 1.2.18 Network identification information (CelfMpCsLineNetwork)

795 CELF_MP_CS_LINE_NETWORK_HOME : home network

796 CELF_MP_CS_LINE_NETWORK_VISIT : visiting network (roamed)

797 CELF_MP_CS_LINE_NO_INFORMATION : No network sinformation

798

799 1.2.19 Service status (CelfMpLineSrvStatus)

800 CELF_MP_LINE_SRV_STATUS_CS : CS is in service.

801 CELF_MP_LINE_SRV_STATUS_PS : PS is in service.

802 CELF_MP_LINE_SRV_STATUS_CSPS : CS and PS are in service.

803 CELF_MP_LINE_NO_INFORMATION : No information

804 CS is the Circuit-Switched Communication Service, and

805 PS is the Packet-Switched Communication Service.

806

807 1.2.20 Restriction status (CelfMpCsLineRestrict)

808 CELF_MP_CS_LINE_RESTRICT_ON : In traffic restriction

809 CELF_MP_CS_LINE_RESTRICT_OFF : Out of traffic restriction

810

811 1.2.21 Identifying flag (CelfMpCsFlag)

- 812 CELF_MP_CS_NO_FLAG : no Flag
- 813 CELF_MP_CS_OPT_FLAG : Network Service Option
- 814 CELF_MP_CS_USSD_FLAG : USSD

815

816 1.2.22 Notification Set (CelfMpCsNotifySet)

- 817 CELF_MP_CS_CLASS_COM_STATUS : Voice communication status notification
- 818 CELF_MP_CS_CLASS_TLK_TIME : Call duration notification
- 819 CELF_MP_CS_CLASS_DISC_CAUSE : Disconnection cause notification
- 820 CELF_MP_CS_CLASS_FW_RESULT : Call forwarding result notification
- 821 CELF_MP_CS_CLASS_OFFHK_TO : Off-hook originating timeout notification

822

823 1.2.23 Event Structure Category

- 824 VoiceNotify

825

826 1.2.24 Event Structure Subtype

- 827 ConnInfo
- 828 TelCallTime
- 829 DiscCause
- 830 FW_Result
- 831 OffHk_Trn
- 832 DCF_Event_type
- 833 AreaInfo
- 834 RssiLevel

835

836 1.2.25 DCF Event Set (CelfMpCsDCFSet)

- 837 CELF_MP_CS_DCF_DISP : Display-related message
- 838 CELF_MP_CS_DCF_HISTORY : History-related message
- 839 CELF_MP_CS_DCF_TONE1 : Tone 1-related message
- 840 CELF_MP_CS_DCF_TONE2 : Tone 2-related message
- 841 CELF_MP_CS_DCF_ETC : Other messages
- 842 CELF_MP_CS_CLASS_ALL : All notified

843

844 1.2.26 Voice message (CelfMpCsRecMsg)

- 845 CELF_MP_CS_REC_MSG_START : Start of a voice message

846 CELF_MP_CS_REC_MSG_STOP : Stop of a voice message
 847

1.2.27 Off Hook Option (CelfMpCsOffHk)

849 CELF_MP_CS_OFFHK_AUTO : Automatic transmission
 850 CELF_MP_CS_OFFHK_MANUAL : Manual transmission
 851

Comment [AK5]: Add to section 1

1.2.28 64K/AV Communication (CelfMpCsUDComStatus)

853 CELF_MP_CS_UD_STOP : Under stop
 854 CELF_MP_CS_UD_TALK : Under communication
 855 CELF_MP_CS_UD_IN : Under incoming
 856 CELF_MP_CS_UD_OUT : Under outgoing
 857 CELF_MP_CS_UD_DISCONNECT : Under disconnection
 858 CELF_MP_CS_UD_CALL_ALERT : Under calling
 859 CELF_MP_CS_UD_HOLD : Under hold
 860 CELF_MP_CS_UD_ERR : Error in UD communication
 861

1.2.29 AV Communication (CelfMpAVComStatus)

863 The INTERNAL reference denotes the originating event utilizing the terminal. The
 864 EXTERNAL reference denotes the usage of an outside of the terminal originating device.

865 CELF_MP_CS_AV_INTERNAL_STOP : Under stop
 866 CELF_MP_CS_AV_INTERNAL_TALK : Under communication
 867 CELF_MP_CS_AV_INTERNAL_IN : Under incoming
 868 CELF_MP_CS_AV_INTERNAL_OUT : Under outgoing
 869 CELF_MP_CS_AV_INTERNAL_DISCONNECT : Under disconnection
 870 CELF_MP_CS_AV_INTERNAL_CALL_ALERT : Under calling
 871 CELF_MP_CS_UD_INTERNAL_HOLD : Under hold
 872 CELF_MP_CS_AV_EXTERNAL_STOP : Under stop
 873 CELF_MP_CS_AV_EXTERNAL_TALK : Under communication
 874 CELF_MP_CS_AV_EXTERNAL_IN : Under incoming
 875 CELF_MP_CS_AV_EXTERNAL_OUT : Under outgoing
 876 CELF_MP_CS_AV_EXTERNAL_DISCONNECT : Under disconnection
 877 CELF_MP_CS_AV_EXTERNAL_CALLING_ALERT : Under calling
 878 CELF_MP_CS_UD_EXTERNAL_HOLD : Under hold
 879 CELF_MP_CS_UD_ERR : Error in UD communication
 880

881 **1.2.30 Receive Types (CelfMpCsRcvType)**

- 882 CELF_MP_CS_RCV_TYPE_COMPETE_TRN : Communication Conflict
- 883 CELF_MP_CS_RCV_TYPE_RSV_RETURN : Reestablish held call
- 884 CELF_MP_CS_RCV_TYPE_CALL_BACK : Network Call Back
- 885 CELF_MP_CS_RCV_TYPE_NORMAL : Normal
- 886 CELF_MP_CS_RCV_TYPE_NONE : No Incoming Call

887

888 **1.2.31 Line Monitoring (CelfMpCsMtype)**

- 889 CELF_MP_CS_MONITOR_LINE_STATUS : Line status change notification
- 890 CELF_MP_CS_MONITOR_RESTRICT : Restriction status change notification
- 891 CELF_MP_CS_MONITOR_RSSI : Receive level change notification
- 892 CELF_MP_CS_MONITOR_ALL : All notified

893

894 **1.2.32 Coverage Indicators (CelfMpCsCoverage)**

- 895 CELF_MP_CS_LINE_STATUS_IN : Within-communication area
- 896 CELF_MP_CS_LINE_STATUS_OUT : Out-of-communication area

897

898 **1.2.33 Incoming Call Selection (CelfMpCallSelect)**

- 899 CELF_MP_CS_INCOMING_VOICE_ANSWERING : Forward to the phone-answering
900 message
- 901 CELF_MP_CS_INCOMING_FORWARD : Forward
- 902 CELF_MP_CS_INCOMING_REJECT : Reject (disconnect)
- 903 CELF_MP_CS_INCOMING_NORMAL : Receipt of an incoming call (normal
904 incoming)

905

906 **1.2.34 Optional Registered Number (CelfMpRegNum)**

907 Within a network, optional numbers can be registered to specific services. It is reflected by
908 this index into the list of these numbers.

- 909 int CelfMpRegNum : Registration number

910

911 **1.2.35 Service Data (CelfMpCsSrvData)**

- 912 char* CelfMpCsSrvData : Pointer to additional service data

913

914 **1.2.36 Reconnection Tone (CelfMpCsReconnectionTone)**

- 915 CELF_MP_CS_RECONN_TONE_OFF : Tone OFF
- 916 CELF_MP_CS_RECONN_TONE_LOW : Tone ON low tone

917 CELF_MP_CS_RECONN_TONE_HI : Tone ON high tone
918

919 1.2.37 Noise Canceling (CelfMpCsNoiseCancel)

920 CELF_MP_CS_ON : Noise canceller ON
921 CELF_MP_CS_OFF : Noise canceller OFF
922

923 1.2.38 Call Quality Alarm (CelfMpCsCallQualAlarm)

924
925 CELF_MP_CS_CALL_QUALITY_ALM_OFF : Quality alarm OFF
926 CELF_MP_CS_CALL_QUALITY_ALM_LOW : Quality alarm ON low tone
927 CELF_MP_CS_CALL_QUALITY_ALM_HI : Quality alarm ON high tone
928

929 1.2.39 Connection Priority Setting (CelfMpCsHiPrioCom)

930 CELF_MP_CS_COMPRI_NONE : No setting
931 CELF_MP_CS_COMPRI_VOICE : Voice
932 CELF_MP_CS_COMPRI_PACKET : Packet
933

934 1.2.40 Message Sound Settings (CelfMpCsVmSound)

935 CELF_MP_CS_ON : Message sound ON
936 CELF_MP_CS_OFF : Message sound OFF
937

938 1.2.41 Incoming Call Auto Receive (CelfMpCsAutoRcv)

939 CELF_MP_CS_ON : Automatic incoming call ON
940 CELF_MP_CS_OFF : Automatic incoming call OFF

941 1.3 Data Types and Structures

942 1.3.1 Circuit switched status notification event structure

943 In this sub-section, the associated data structure is CelfMpEvent with the following values:

944 category = VoiceNotify;

945 subtype = ConnInfo;

946 The value of field "info" is from enum CelfMpCsComStatus.

947 The field "data" carries:

```
948 CelfMpCsResChgInf    res_chg_inf;    // to be used in the case of:
949                                     // Restriction display information structure
```

951 1.3.2 Call duration notification event structure

952 In this sub-section, the associated data structure is CelfMpEvent with the following values:

953 category = VoiceNotify;

954 subtype = TelCallTime;

955 The value of field "info" is Call duration (seconds).

956 The field "data" is unused.

957

958 1.3.3 Disconnection cause notification event structure

959 In this sub-section, the associated data structure is CelfMpEvent with the following values:

960 category = VoiceNotify;

961 subtype = DiscCause;

962 The value of field "info" is the call reference.

963 The field "data" carries:

```
964 CelfMpCsDiscCause cme; // Disconnection cause information structure
```

965

966 1.3.4 Disconnection cause information structure

967 The error codes and cause information in this section directly correspond to 3GPP TS
968 24.008.

```
969 typedef struct {
```

```
970     unsigned char e_code; // Result code flag
```

```
971     unsigned char code; // Result code
```

```
972     unsigned char e_cause; // Error reason flag
```

```
973     unsigned char cause; // Error reason
```

```
974 } CelfMpCsDiscCause;
```

975

976 1.3.5 Forwarding result notification event structure

977 In this sub-section, the associated data structure is CelfMpEvent with the following values:

978 category = VoiceNotify;

979 subtype = FW_Result

980 The value of field "info" is the call reference.

981 The value of "subinfo" carries the forwarding result.

982 The field "data" carries:

983 CelfMpCsFwResult fw_result; // Forwarding result structure

984

985 1.3.6 Forwarding result structure (CelfMpCsFwResult)

986 typedef struct {

987 int cause ; // forwarding result details

988 } CelfMpCsFwResult;

989

990 1.3.7 Off-hook transmission timeout event structure

991 In this sub-section, the associated data structure is CelfMpEvent with the following values:

992 category = VoiceNotify;

993 subtype = OffHk_Trn

994 The value of field "info" is the call reference.

995 The field "data" is unused.

996

997 1.3.8 Connection Destination Information (CelfMpConnectInfo)

998 typedef struct {

999 CelfMpCallRef CN_No; // Call reference

1000 CelfMpCsCallRefStatus CN_status;

1001 int continue_flag;

1002 unsigned char Calling_Dial [CELF_MP_CS_DIAL_MAX+1];

1003 unsigned char Called_Dial [CELF_MP_CS_DIAL_MAX+1];

1004 CelfMpCsBusyTone BusyToneSound_inf;

1005 CelfMpCsBtype bc_type;

1006 unsigned char[10] taf_address;

1007 unsigned char cause_of_NoCLI;

1008 unsigned char num_presentation_indicator;

1009 unsigned char redirectnum [CELF_MP_CS_DIAL_MAX+1];

1010 unsigned char redirect_presentation_indicator;

Comment [AK6]: Size does not fit the bit field

Classification: *Circuit-Switched Service*

```

1011     unsigned char        signal;
1012     CelfMpCsDiscCause    cause; // Disconnection cause information structure
1013     } CelfMpCsConnectInf
1014

```

1.3.9 Connection Request (CelfMpCsConReq)

```

1016     typedef struct {
1017     CelfMpCsBtype        type;
1018     unsigned char *      dial_buf;
1019     int                  dial_len;
1020     CelfMpCsNotice      notice;
1021     unsigned char *      subaddr_buf;
1022     int                  subaddr_len;
1023     } CelfConReq
1024

```

1.3.10 Redirection number

```

1026     Destination number of call transfer.
1027     redirectnum [CELF_MP_CS_DIAL_MAX+1]
1028

```

1.3.11 Channel Number Information (CelfMpCsChanNum)

1030 CelfMpCsChanNum is used to hold call reference information.
 1031 If a channel is not used, CELF_MP_CS_CHAN_NOUSE is set as the call reference.

```

1033     typedef struct {
1034     int Slot_0           // Call reference information 00
1035     int Slot_1           // Call reference information 01
1036     int Slot_2           // Call reference information 02
1037     } CelfMpCsChanNum
1038

```

1.3.12 Channel not in use Flag

```

1040     int CELF_MP_CS_CHAN_NOUSE      : usually holds the call reference if channels are
1041     not used.
1042

```

1.3.13 DCF Event Structure

1044 In this sub-section, the associated data structure is CelfMpEvent with the following values:
 1045 category = VoiceNotify;
 1046 subtype = DCF_Event_type;

Classification: *Circuit-Switched Service*

1047 The value of field "info" is the notification type.
1048 The value of field "subinfo" is the bearer type
1049 The field "data" carries:
1050 DCF message structure corresponding to report types.
1051

1.3.14 Line status change notification event structure

1053 In this sub-section, the associated data structure is CelfMpEvent with the following values:
1054 category = VoiceNotify;
1055 subtype = ArealInfo;
1056 The value of field "info" is the line status.
1057 The value of field "subinfo" is the line type.
1058 The field "data" is unused.
1059

Comment [AK7]: CHECK
THE SPELLING

1.3.15 Restriction display information structure (CelfMpCsResChgInf)

```
1062 typedef struct {  
1063     CelfMpCsLineRestrict    NcRestriction; // Normal originating restriction  
1064     CelfMpLineSrvStatus     ServiceStatus; // Service status  
1065     CelfMpCsLineRestrict    EcRestriction; // Emergency originating restriction  
1066 } CelfMpCsResChgInf;
```

1.3.16 Receive level change notification event structure

1068 In this sub-section, the associated data structure is CelfMpEvent with the following values:
1069 category = VoiceNotify;
1070 subtype = RssiLevel;
1071 The value of field "info" is the receive level.
1072 The value of field "subinfo" is the line type.
1073 The field "data" is unused.
1074
1075

1.3.17 Line Status structure (CelfMpCsAreaRefChgInf)

```
1077 typedef struct {  
1078     CelfMpCsLineStatus      LineStatus ;           // Line status  
1079     CelfMpCsCoverage        CoverageStatus ;      // Service status  
1080     CelfMpCsLineRRCMODE     RRC_mode ;           // RRC mode  
1081     CelfMpCsLineNetwork     Network ;             // Network identification  
1082     information
```

Classification: *Circuit-Switched Service*

```

1083     CelfMpCsLineCvrStatus      ServiceStatus_AREA ; // Area status information
1084     // This information is set up when a cellular phone receives "AreaStatus-ind" from the
1085     // network.
1086     // It shows effective service in the present area.
1087     // Either "CS", "PS", "CS &PS" or "No data" is set.
1088     CelfMpCsLineRestrict      RestrictStatus ; // Restriction status
1089     CelfMpCsLineRestrict      NcRestriction ; // Normal originating restriction
1090     CelfMpLineSrvStatus       ServiceStatus_RES ; // Service status restriction
1091     // This information is set up when a cellular phone receives "Permission-ind" from the
1092     // network side.
1093     // It shows the service regulated in the present area.
1094     // Either "CS", "PS" or "CS & PS" is set.
1095     CelfMpCsLineRestrict      EcRestriction ; // Emergency originating
1096     restriction
1097     } CelfMpCsAreaRefChgInf ;
1098

```

1.3.18 Additional service data structure (CelfMpCsAddSrvData)

```

1100     typedef struct {
1101         CelfMpCsFlag   flag ;
1102         char    title[CELF_SRVINFO_TITLE]; // Additional service name
1103                                           // CELF_SRVINFO_TITLE=21
1104         char    send_no[CELF_SRVINFO_DATA]; // Dial data for accessing the service
1105                                           // CELF_SRVINFO_DATA=40
1106     } CelfMpCsAddSrvData;
1107

```

1.3.19 Response Message Data Structure (CelfMpCsResponseMsgData)

The additional response message information is the service name and Dial data, which is response message to send the network.

```

1112
1113     typedef struct {
1114         unsigned char   title[CELF_RESMSG_TITLE] ; // Service name
1115         unsigned char   res_msg[CELF_RESMSG_DATA]; // Dial data
1116     } CelfMpCsResponseMsgData;
1117

```

1118 1.3.20 Line Status Extension (CelfMpCsLineStatusEx)
1119 unsigned char* CelfMpCsLineStatusEx; // data for additional line status
1120 information

1121

1122 1.3.21 Number of stored messages (CelfMpCsVMNum)

1123 int CelfMpCsVMNum

1124

1125 1.3.22 Date Format Structure (CelfMpCsDate)

1126 typedef struct {

1127 unsigned char Month

1128 unsigned char Day

1129 unsigned char Hour

1130 unsigned char Minute

1131 } CelfMpCsDate

1132

1133 1.3.23 Dial Buffer (CelfMpCsDialBuffer)

1134 char* CelfMpCsDialBuffer

1135

1136 1.3.24 Dial Buffer Length (CelfMpCsDialLen)

1137 int CelfMpCsDialLen

1138

1139 1.3.25 Multi Party Operation (CelfMpCsMop)

1140 CELF_MP_CS_MOP_RSV_DISC: // Disconnect the hold call

1141 CELF_MP_CS_MOP_DISC_AND_RSP: // Response after disconnection

1142 CELF_MP_CS_MOP_RSV_AND_RSP: // Response after hold (including operation for
1143 switching a call)

1144 CELF_MP_CS_MOP_CR_DISC: // Disconnect call specified by the call reference

1145

1146 1.3.26 Timer Value (CelfMpCsTimer)

1147 int CelfMpCsTimer // value 1 .. 120 seconds

1148

1149 **1.4 Events Type**

1150 **1.4.1 DCF Event Type**

- 1151 CELF_MP_CS_DCF_DISP : Display-related message
- 1152 CELF_MP_CS_DCF_HISTORY : History-related message
- 1153 CELF_MP_CS_DCF_TONE1 : Tone 1-related message
- 1154 CELF_MP_CS_DCF_TONE2 : Tone 2-related message
- 1155 CELF_MP_CS_DCF_ETC : Other messages
- 1156 CELF_MP_CS_CLASS_ALL : All notified

1157

1158 **1.4.2 CCP Notification type**

1159 **1.4.2.1 CELF_MP_CS_CCP_CALLING_START_REQ**

1160 **Description:** Notification of starting display during CCP outgoing

1161

1162 **1.4.2.2 CELF_MP_CS_CCP_CALLED_START_IND**

1163 **Description:** Notification of starting display during CCP incoming

1164

1165 **1.4.2.3 CELF_MP_CS_CCP_CALLING_ALERT_IND**

1166 **Description:** Notification of starting display during CCP calling

1167

1168 **1.4.2.4 CELF_MP_CS_CCP_CONNECT_START_RSP**

1169 **Description:** Notification of starting display during CCP connection

1170

1171 **1.4.2.5 CELF_MP_CS_CCP_CONNECT_START_IND**

1172 **Description:** Notification of starting display during CCP communication

1173

1174 **1.4.2.6 CELF_MP_CS_CCP_RELEASE_IND**

1175 **Description:** Notification of ending CCP display

1176

1177 **1.4.2.7 CELF_MP_CS_CCP_DISCONNECT_REQ**

1178 **Description:** Notification of starting CCP disconnection (on a mobile device) display

1179

1180 **1.4.2.8 CELF_MP_CS_CCP_DISCONNECT_START_IND**

1181 **Description:** Notification of starting CCP disconnection (on a network) display

1182

1183 **1.4.2.9 CELF_MP_CS_CCP_CALLING_REJ_IND**

1184 **Description:** Notification of rejecting CCP outgoing

1185

1186 **1.4.2.10 CELF_MP_CS_CCP_HOLD_CNF**

1187 **Description:** Notification of CCP hold

1188

1189 **1.4.2.11 CELF_MP_CS_CCP_RETRIEVE_CNF**

1190 **Description:** Notification of releasing CCP hold

1191

1192 **1.4.2.12 CELF_MP_CS_CCP_CALLING_SETUP_REQ**

1193 **Description:** Notification of registering CCP outgoing call history

1194

1195 **1.4.2.13 CELF_MP_CS_CCP_CALLED_REJ_REQ**

1196 **Description:** Notification of registering CCP absence incoming call history

1197

1198 **1.4.2.14 CELF_MP_CS_CCP_CALLED_SETUP_RSP**

1199 **Description:** Notification of registering CCP incoming call history

1200

1201 **1.4.2.15 CELF_MP_CS_CCP_RGT_START**

1202 **Description:** Notification of CCP RGT start

1203

1204 **1.4.2.16 CELF_MP_CS_CCP_RGT_STOP**

1205 **Description:** Notification of CCP RGT stop

1206

1207 **1.4.2.17 CELF_MP_CS_CCP_HRGT_START**

1208 **Description:** Start notification of incoming of a CCP hold call

1209

1210 **1.4.2.18 CELF_MP_CS_CCP_HRGT_STOP**

1211 **Description:** Stop notification of incoming of a CCP hold call

1212

1213 **1.4.2.19 CELF_MP_CS_CCP_DST_START**

1214 **Description:** Notification of CCP DST start

1215

1216 **1.4.2.20 CELF_MP_CS_CCP_DST_STOP**

1217 **Description:** Notification of CCP DST stop

1218

1.4.2.21 CELF_MP_CS_CCP_RBT_START

1219

Description: Notification of CCP RBT start

1220

1221

1.4.2.22 CELF_MP_CS_CCP_RBT_STOP

1222

Description: Notification of CCP RBT stop

1223

1224

1.4.2.23 CELF_MP_CS_CCP_BT_START

1225

Description: Notification of CCP BT start

1226

1227

1.4.2.24 CELF_MP_CS_CCP_CWT_START

1228

Description: Notification of CCP CWT start

1229

1230

1.4.2.25 CELF_MP_CS_CCP_CWT_STOP

1231

Description: Notification of CCP CWT stop

1232

1233

1.4.2.26 CELF_MP_CS_CCP_REJECT_ASK

1234

Description: Inquiry report of rejecting a CCP CS incoming call

1235

1236

1.4.3 Notification type

1237

CELF_MP_CS_RSMP_REST_START : Restriction display start notification

1238

CELF_MP_CS_RSMP_REST_STOP : Restriction display stop notification

1239

1240

1.4.4 Restriction status

1241

The 0th bit is used for PS restriction status, and the 1st bit is used for CS restriction status.

1242

(Bit ON means "restricted." Bit OFF means "unrestricted.")

1243

CELF_MP_BIT_RESTINF_CS : CS restriction information

1244

CELF_MP_BIT_RESTINF_PS : PS restriction information

1245

The 2nd bit is used for PS emergency restriction status, and the 3rd bit is used for CS emergency restriction status.

1246

1247

CELF_MP_BIT_ECRESTINF_CS : Emergency CS restriction information

1248

CELF_MP_BIT_ECRESTINF_PS : Emergency PS restriction information

1249

1250 **1.5 Status Codes (CelfMpStatus)**

1251	CELF_MP_STATUS_OK	: successful completion
1252	CELF_MP_STATUS_APP_ID_ERR	: Application ID is not valid
1253	CELF_MP_STATUS_CALL_NO_ERR	: Call number is not valid
1254	CELF_MP_STATUS_EVENT_SET_ERR	: Notification event set is not valid
1255	CELF_MP_STATUS_COM_TYPE_ERR	: Communication type is not valid
1256	CELF_MP_STATUS_MON_TYPE_ERR	: Monitor type is not valid
1257	CELF_MP_STATUS_ERR	: Other unsuccessful completion
1258	CELF_MP_CS_ONHOOK_DENY	: On-hook originating is impossible.
1259	CELF_MP_CS_ONHOOK_STATUS_ERR	: Error due to communication conflict
1260	CELF_MP_CS_ONHOOK_OB_CR	: Excess of the maximum number of calls
1261		
1262		

DRAFT

1263

2. Start Notification

1264

2.1 Symbol: `celf_mp_cs_notification_start`

1265

2.1.1 Syntax

```

1266 CelfMpStatus celf_mp_cs_notification_start (
1267     CelfMpAppID          app_id,
1268     CelfMpCsNotifySet    event_set,
1269     CelfMpCallback       callback_func);

```

1270

1271

2.1.2 Argument

1272 **Name:** `app_id`1273 **Type:** `CelfMpAppId`1274 **I/O:** `I`1275 **Description:**

1276 Application identifier.

1277

1278 **Name:** `event_set`1279 **Type:** `CelfMpCsNotifySet`1280 **I/O:** `I`1281 **Description:**

1282 Notification event set. Events that are classified as belonging to one of the
 1283 `CelfMpCsNotifySet` class **may** be registered to have a callback function called when
 1284 the event occurs for the application identified by `app_id`. Classes of events are enabled by
 1285 setting the corresponding bit in `event_set`:

1286

1287 The event classes are defined as follows:

1288 `CELF_MP_CS_CLASS_COM_STATUS:` Voice communication status notification1289 `CELF_MP_CS_CLASS_TLK_TIME:` Call duration notification1290 `CELF_MP_CS_CLASS_DISC_CAUSE:` Disconnection cause notification1291 `CELF_MP_CS_CLASS_FW_RESULT:` Call forwarding result notification1292 `CELF_MP_CS_CLASS_OFFHK_TO:` Off-hook originating timeout notification

1293

1294 A callback **may** be registered for all classes of events using special event class

1295 `CELF_MP_CS_CLASS_ALL`, however to reduce overhead it is recommended that only the
 1296 needed event classes **should** be registered.

1297

1298 **Name:** `callback_func`1299 **Type:** `CelfMpCallback`

1300 I/O: I

1301 **Description:**

1302 The callback function, which **shall** be called when an event occurs from one of the classes
1303 in `event_set`.

1304

1305 2.1.3 Return Value

1306 **Type:** `CelfMpStatus`

1307 **Description:**

1308 `celf_mp_cs_notification_start()` **shall** return one of the following values:

1309 `CELF_MP_STATUS_OK:` successful completion

1310 `CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid

1311 `CELF_MP_STATUS_EVENT_SET_ERR:` Notification event set is not valid

1312 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1313

1314 2.1.4 Include File

1315 `/usr/include/celf/mp_cs.h`

1316

1317 2.1.5 Functional Description

1318 This is a synchronous function.

1319 This function is used to start notification callbacks for events related to circuit-switched
1320 communication.

1321 Events from a registered class **shall** cause the registered callback function to be called
1322 when the event occurs for the application identified by `app_id`. If a class of events does not
1323 have a registered callback function, no callback **shall** occur for those events.

1324

1325 The event structure `CelfMpEvent` **must** be used and the value subtype **shall be set to**
1326 **ConnInfo**. See section 1.3.1.

1327

1328

1329 3. Stop Notification

1330 3.1 Symbol: `celf_mp_cs_notification_stop`

1331 3.1.1 Syntax

```
1332 CelfMpStatus celf_mp_cs_notification_stop (  
1333     CelfMpAppld      app_id,  
1334     CelfMpCsNotifySet event_set);  
1335
```

1336 3.1.2 Argument

1337 **Name:** `app_id`

1338 **Type:** `CelfMpAppld`

1339 **I/O:** |

1340 **Description:**

1341 Application identifier.

1342

1343 **Name:** `event_set`

1344 **Type:** `CelfMpCsNotifySet`

1345 **I/O:** |

1346 **Description:**

1347 Notification event set. Events that are classified as belonging to one of the
1348 `CelfMpCsNotifySet` class **may** be registered to have a callback function called when
1349 the event occurs for the application identified by `app_id`. Classes of events are enabled by
1350 setting the corresponding bit in `event_set`.

1351

1352 The event classes are defined as follows:

1353 `CELF_MP_CS_CLASS_COM_STATUS:` Voice communication status notification

1354 `CELF_MP_CS_CLASS_TLK_TIME:` Call duration notification

1355 `CELF_MP_CS_CLASS_DISC_CAUSE:` Disconnection cause notification

1356 `CELF_MP_CS_CLASS_FW_RESULT:` Call forwarding result notification

1357 `CELF_MP_CS_CLASS_OFFHK_TO:` Off-hook originating timeout notification

1358

1359 3.1.3 Return Value

1360 **Type:** `CelfMpStatus`

1361 **Description:**

1362 `celf_mp_cs_notification_stop()` **shall** return one of the following values:

1363 `CELF_MP_STATUS_OK:` successful completion

1364 `CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1365 CELF_MP_STATUS_EVENT_SET_ERR: Notification event set is not valid
1366 CELF_MP_STATUS_ERR: Other unsuccessful completion.
1367

1368 3.1.4 Include File

1369 /usr/include/celf/mp_cs.h

1370

1371 3.1.5 Functional Description

1372 This function stops all voice communication related event reporting.

1373 If any voice communication function is called after the notification has been stopped, the
1374 call **shall** report CELF_MP_STATUS_ERR.

1375 For notification events, see "Start notification".

1376 Note: For further information about the event structure consult section 1.3 in this document.

1377

1378

1379

DRAFT

1380 **4. Get Voice Communication Status**

1381 **4.1 Symbol: celf_mp_cs_get_com_status**

1382 **4.1.1 Syntax**

```
1383 CelfMpStatus celf_mp_cs_get_com_status (  
1384     CelfMpAppId app_id);  
1385
```

1386 **4.1.2 Argument**

1387 **Name:** app_id

1388 **Type:** CelfMpAppId

1389 **I/O:** I

1390 **Description:**

1391 Application identifier.
1392

1393 **4.1.3 Return Value**

1394 **Type:** CelfMpStatus

1395 **Description:**

1396 `celf_mp_cs_get_com_status()` shall return one of the values defined in section 0.1.
1397

1398 **4.1.4 Include File**

1399 `/usr/include/celf/mp_cs.h`
1400

1401 **4.1.5 Functional Description**

1402 This function gets the current voice communication status.

1403 Without the monitoring the voice communication, it is possible to get the status of voice
1404 communication.
1405
1406
1407

1408 **5. Get Connection Information to Other Party**

1409 **5.1 Symbol: celf_mp_cs_get_con_info_ref**

1410 **5.1.1 Syntax**

```
1411 CelfMpStatus celf_mp_cs_get_con_info_ref (  
1412     CelfMpAppId      app_id,  
1413     CelfMpCallRef    call_no,  
1414     CelfMpConnectInfo connect_inf_p);  
1415
```

1416 **5.1.2 Argument**

1417 **Name:** app_id

1418 **Type:** CelfMpAppId

1419 **I/O:** I

1420 **Description:**

1421 Application identifier.

1422

1423 **Name:** call_no

1424 **Type:** CelfMpCallRef

1425 **I/O:** I

1426 **Description:**

1427 Call reference (0 to 255).

1428

1429 **Name:** connect_inf_p

1430 **Type:** CelfMpConnectInfo

1431 **I/O:** I

1432 **Description:**

1433 Pointer to the connection destination information. See section 0.1 for details.

1434

1435 **5.1.3 Return Value**

1436 **Type:** CelfMpStatus

1437 **Description:**

1438 celf_mp_cs_get_con_info_ref() **shall** return one of the values defined:

1439 CELF_MP_STATUS_OK: successful completion

1440 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid.

1441 CELF_MP_STATUS_CALL_NO_ERR: Call number is not valid

1442 CELF_MP_STATUS_ERR: Other unsuccessful completion.

1443

1444 5.1.4 Include File

1445 /usr/include/celf/mp_cs.h

1446

1447 5.1.5 Functional Description

1448 This function refers to the connection information to other party specified call reference

1449 Without the monitoring the voice communication, it is possible to get the connection
1450 information

1451

1452 In the following cases, the CelfMpStatus is set CELF_MP_CS_ERR.

1453 1. The call specified by call reference does not exist.

1454 2. Other parameter Error.

1455

DRAFT

1456

6. Get Call Duration

1457

6.1 Symbol: `celf_mp_cs_get_call_duration`

1458

6.1.1 Syntax

1459

```
CelfMpStatus celf_mp_cs_get_call_duration(
```

1460

```
    CelfMpAppId app_id,
```

1461

```
    CelfMpTime time);
```

1462

1463

6.1.2 Argument

1464

Name: `app_id`

1465

Type: `CelfMpAppId`

1466

I/O: `I`

1467

Description:

1468

Application identifier.

1469

1470

Name: `time`

1471

Type: `CelfMpTime`

1472

I/O: `O`

1473

Description:

1474

`time` returns the current call duration in seconds.

1475

1476

6.1.3 Return Value

1477

Type: `CelfMpStatus`

1478

Description:

1479

`celf_mp_cs_get_con_info_ref()` **shall** return one of the values defined:

1480

`CELF_MP_STATUS_OK:` successful completion

1481

`CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1482

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1483

1484

6.1.4 Include File

1485

`/usr/include/celf/mp_cs.h`

1486

1487

6.1.5 Functional Description

1488

This function gets the call duration on the current call.

1489

The call duration is counted by the voice communication service.

1490 When no call exists, the function returns zero.
1491

DRAFT

1492

7. Off-Hook Notification

1493

7.1 Symbol: `celf_mp_cs_notification_off_hook`

1494

7.1.1 Syntax

1495

```
CelfMpStatus celf_mp_cs_notification_off_hook (
```

1496

```
    CelfMpAppld      app_id,
```

1497

```
    CelfMpCsBtype    com_type,
```

1498

```
    CelfMpCsOffHk    option);
```

1499

1500

7.1.2 Argument

1501

Name: `app_id`

1502

Type: `CelfMpAppld`

1503

I/O: `I`

1504

Description:

1505

Application identifier.

1506

1507

Name: `com_type`

1508

Type: `CelfMpCsBtype`

1509

I/O: `I`

1510

Description:

1511

Communication type as defined in section 1.2.4

1512

1513

Name: `option`

1514

Type: `CelfMpCsOffHk`

1515

I/O: `I`

1516

Description:

1517

One the following options **shall** be set:

1518

`CELf_MP_CS_OFFHK_AUTO` Automatic transmission

1519

`CELf_MP_CS_OFFHK_MANUAL` Manual transmission

1520

1521

7.1.3 Return Value

1522

Type: `CelfMpStatus`

1523

Description:

1524

`celf_mp_cs_notification_off_hook()` **shall** return one of the values defined:

1525

`CELf_MP_STATUS_OK`: successful completion

1526

`CELf_MP_STATUS_APP_ID_ERR`: Application ID is not valid.

Comment [AK8]: Add to section 1

1527 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid
1528 CELF_MP_STATUS_ERR: Other unsuccessful completion.
1529

1530 7.1.4 Include File

1531 /usr/include/celf/mp_cs.h

1532

1533 7.1.5 Functional Description

1534 This function receives the request of off-hook.

1535

1536 The term "off-hook" refers to the user first presses the "dial" button, then enters the number
1537 to dial.

1538

1539 By this function,

1540 (1) When the mobile phone is in the wait (standby) status, the dial tone (DT) sounds and it
1541 is possible to input dial number, or

1542 (2) When the input of dial number is completed, the mobile phone starts the originating.

1543 Because the function is an immediate return function, to confirm the complete result,
1544 including the negotiation with the network, `celf_mp_cs_notification_status()`
1545 shall be used to obtain the communication status.

1546

1547 When a mobile phone is moved to low voltage mode, a low voltage notification is sent.

1548 During low voltage, when the communication status is other than the under standby, this
1549 Off-hook is disabled.

1550

1551 If an incoming call arrives during off-hook, this Off-hook is cancelled.

1552 In case of using the subaddress, it should be use the function "On-hook originating".

1553 **8. Disconnect**

1554 **8.1 Symbol: celf_mp_cs_disconnect**

1555 **8.1.1 Syntax**

```
1556 CelfMpStatus celf_mp_cs_disconnect (  
1557     CelfMpAppId    app_id,  
1558     CelfMpCsBtype  com_type);
```

1559

1560 **8.1.2 Argument**

1561 **Name:** app_id

1562 **Type:** CelfMpAppId

1563 **I/O:** I

1564 **Description:**

1565 Application identifier.

1566

1567 **Name:** com_type

1568 **Type:** CelfMpCsBtype

1569 **I/O:** I

1570 **Description:**

1571 Communication type as defined in section 1.2.4.

1572

1573 **8.1.3 Return Value**

1574 **Type:** CelfMpStatus

1575 **Description:**

1576 `celf_mp_cs_disconnect()` shall return one of the values defined:

1577 CELF_MP_STATUS_OK: successful completion

1578 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid

1579 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1580 CELF_MP_STATUS_ERR: Other unsuccessful completion.

1581

1582 **8.1.4 Include File**

1583 `/usr/include/celf/mp_cs.h`

1584

1585 **8.1.5 Functional Description**

1586 This function receives the request to disconnect the call.

1587

1588 Because the function is an immediate return function, to confirm the complete result,
1589 including the negotiation with the network, it should be issued
1590 `celf_mp_cs_notification_status()` to obtain the communication status.

1591

1592 An incoming call cannot be disconnected by this function. (Use "Reject incoming call")

1593

1594 If multiple calls exist, all calls are disconnected.

1595

DRAFT

1596 9. Dial

1597 9.1 Symbol: `celf_mp_cs_dial`

1598 9.1.1 Syntax

```
1599 CelfMpStatus celf_mp_cs_dial (  
1600     CelfMpAppId      app_id,  
1601     CelfMpCsBtype    com_type,  
1602     CelfMpCsDialBuffer dial_buf,  
1603     CelfMpCsDialLen  dial_len);  
1604
```

1605 9.1.2 Argument

1606 **Name:** `app_id`

1607 **Type:** `CelfMpAppId`

1608 **I/O:** |

1609 **Description:**

1610 Application identifier.

1611

1612 **Name:** `com_type`

1613 **Type:** `CelfMpCsBtype`

1614 **I/O:** |

1615 **Description:**

1616 Communication type as defined in section 1.2.4.

1617

1618 **Name:** `dial_buf`

1619 **Type:** `CelfMpCsDialBuffer`

1620 **I/O:** |

1621 **Description:**

1622 Dial data buffer address

1623

1624 **Name:** `dial_len`

1625 **Type:** `CelfMpCsDialLen`

1626 **I/O:** |

1627 **Description:**

1628 Dial data length

1629

1630 **9.1.3 Return Value**

1631 **Type:** CelfMpStatus

1632 **Description:**

1633 `celf_mp_cs_dial()` shall return one of the values defined:

- 1634 CELF_MP_STATUS_OK: successful completion
- 1635 CELF_MP_STATUS_DIAL_ERR: Number in dial buffer or dial buffer incorrect
- 1636 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid.
- 1637 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid
- 1638 CELF_MP_STATUS_ERR: Other unsuccessful completion.

1639

1640 **9.1.4 Include File**

1641 `/usr/include/celf/mp_cs.h`

1642

1643 **9.1.5 Functional Description**

1644 This function receives the sequence of dial number.

1645

1646 Because the function is an immediate return function, to confirm the complete result,
1647 including the negotiation with the network, it should be issued
1648 "`celf_mp_cs_notification_status()`" to obtain the communication status.

1649

1650 The dial data stores the following ASCII codes.

- 1651 1 : 0 x 31 2 : 0 x 32 3 : 0 x 33
- 1652 4 : 0 x 34 5 : 0 x 35 6 : 0 x 36
- 1653 7 : 0 x 37 8 : 0 x 38 9 : 0 x 39
- 1654 * : 0 x 2a 0 : 0 x 30 # : 0 x 23

1655

1656 Under this off-hook status, the mobile phone starts an outgoing call with "Dial" and
1657 "Complete dial".

1658 Five seconds later from the last digit has been entered, the outgoing process starts
1659 automatically, when automatic transmission is specified in "Off-hook".

1660 When "Off-hook" is called, the mobile phone is in off-hook status.

1661

1662 Under this on-hook status, DTMF is sent, if the status is (a) the conversation or (b) the
1663 conversation and hold.

1664 **10.Dial Complete**

1665 **10.1 Symbol: celf_mp_cs_dial_end**

1666 **10.1.1 Syntax**

```
1667 CelfMpStatus celf_mp_cs_dial_end (  
1668     CelfMpAppId      app_id,  
1669     CelfMpCsBtype    com_type);
```

1670

1671 **10.1.2 Argument**

1672 **Name:** app_id

1673 **Type:** CelfMpAppId

1674 **I/O:** I

1675 **Description:**

1676 Application identifier.

1677

1678 **Name:** com_type

1679 **Type:** CelfMpCsBtype

1680 **I/O:** I

1681 **Description:**

1682 Communication type as defined in section 1.2.4.

1683

1684 **10.1.3 Return Value**

1685 **Type:** CelfMpStatus

1686 **Description:**

1687 `celf_mp_cs_dial_end()` shall return one of the values defined:

1688 CELF_MP_STATUS_OK: successful completion

1689 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid.

1690 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1691 CELF_MP_STATUS_ERR: Other unsuccessful completion.

1692

1693 **10.1.4 Include File**

1694 `/usr/include/celf/mp_cs.h`

1695

1696 **10.1.5 Functional Description**

1697 This function receives the request to end the dial entry.

1698

1699

Because this is an asynchronous function the service will return the result through a notification.

1700

1701

`celf_mp_cs_notification_status()` shall be used to obtain the communication status.

1702

1703

Under off-hook status, the mobile phone starts outgoing operation by calling this function with dial number, which was given by preceding function calls "Dial".

1704

1705

1706

Under on-hook status, the calling this function is disabled, a `CELF_MP_STATUS_ERR` **shall** be returned.

1707

1708

DRAFT

1709 11. Response to Incoming Call

1710 11.1 Symbol: `celf_mp_cs_call_rcv`

1711 11.1.1 Syntax

```
1712 CelfMpStatus celf_mp_cs_call_rcv (  
1713     CelfMpAppld      app_id,  
1714     CelfMpCsBtype    com_type);  
1715
```

1716 11.1.2 Argument

1717 **Name:** `app_id`

1718 **Type:** `CelfMpAppld`

1719 **I/O:** `I`

1720 **Description:**

1721 Application identifier.

1722

1723 **Name:** `com_type`

1724 **Type:** `CelfMpCsBtype`

1725 **I/O:** `I`

1726 **Description:**

1727 Communication type as defined in section 1.2.4.

1728

1729 11.1.3 Return Value

1730 **Type:** `CelfMpStatus`

1731 **Description:**

1732 `celf_mp_cs_call_rcv()` shall return one of the values defined:

1733 `CELF_MP_STATUS_OK:` successful completion

1734 `CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid.

1735 `CELF_MP_STATUS_COM_TYPE_ERR:` Communication type is not valid

1736 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

1737

1738 11.1.4 Include File

1739 `/usr/include/celf/mp_cs.h`

1740

1741 11.1.5 Functional Description

1742 This function receives the request to process an incoming call.

1743

1744 Because the function is an immediate return function, to confirm the complete result,
1745 including the negotiation with the network, it should be issued
1746 `celf_mp_cs_notification_status()` to obtain the communication status.

1747

1748 One of the following operations is performed depending on the mobile phone status.

1749 Incoming : Responds to the incoming call.

1750 In hold : Responds to the response hold call

1751 Others : Disabled

1752

1753 If the mobile phone is in low voltage mode, this function is disabled.

1754

1755 To respond to the incoming call in the status, "in conversation and incomings", use "Reject
1756 incoming call".

DRAFT

1757 **12.Forward Incoming Call**

1758 **12.1 Symbol: celf_mp_cs_call_forward**

1759 12.1.1 Syntax

1760 CelfMpStatus celf_mp_cs_call_forward (
1761 CelfMpCsBtype com_type);
1762

1763 12.1.2 Argument

1764 **Name:** com_type

1765 **Type:** CelfMpCsBtype

1766 **I/O:** I

1767 **Description:**

1768 Communication type as defined in section 1.2.4.
1769

1770 12.1.3 Return Value

1771 **Type:** CelfMpStatus

1772 **Description:**

1773 celf_mp_cs_call_forward() **shall** return one of the values defined:

1774 CELF_MP_STATUS_OK: successful completion

1775 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1776 CELF_MP_STATUS_ERR: Other unsuccessful completion.
1777

1778 12.1.4 Include File

1779 /usr/include/celf/mp_cs.h
1780

1781 12.1.5 Functional Description

1782 This function receives the request to forward an incoming call.
1783

1784 Because the function is an immediate return function, to confirm the complete result,
1785 including the negotiation with the network, a celf_mp_cs_notification_status()
1786 should be issued to obtain the communication status.
1787

1788 The incoming call is forwarded when the communication status is (a) under the incoming,
1789 (b) under conversation and incoming, or (c) under hold and incoming.
1790

1791 If the forwarding fails, incoming call is continued between other party and this phone.

1792 **13.Forward to Voice Mail System**

1793 **13.1 Symbol: celf_mp_cs_call_forward_voice_msg**

1794 **13.1.1 Syntax**

1795 CelfMpStatus celf_mp_cs_call_forward_voice_msg (
1796 CelfMpCsBtype com_type);
1797

1798 **13.1.2 Argument**

1799 **Name:** com_type

1800 **Type:** CelfMpCsBtype

1801 **I/O:** I

1802 **Description:**

1803 Communication type as defined in section 1.2.4.
1804

1805 **13.1.3 Return Value**

1806 **Type:** CelfMpStatus

1807 **Description:**

1808 celf_mp_cs_call_forward_voice_msg() **shall** return one of the values defined:

1809 CELF_MP_STATUS_OK: successful completion

1810 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1811 CELF_MP_STATUS_ERR: Other unsuccessful completion.
1812

1813 **13.1.4 Include File**

1814 /usr/include/celf/mp_cs.h
1815

1816 **13.1.5 Functional Description**

1817 This function receives the request to forward a call to a voice mail system.
1818

1819 Because the function is an immediate return function, to confirm the complete result,
1820 including the negotiation with the network, it should be issued
1821 celf_mp_cs_notification_status() to obtain the communication status.
1822

1823 The incoming call is forwarded to phone-answering message when the communication
1824 status is (a) under the incoming, (b) under conversation and incoming, or (c) under hold
1825 and incoming.
1826

1827

If the forwarding fails, incoming call is continued between other party and this phone.

DRAFT

1828 **14.Call Hold**

1829 **14.1 Symbol: celf_mp_cs_call_hold**

1830 **14.1.1 Syntax**

1831 CelfMpStatus celf_mp_cs_call_hold(
1832 CelfMpCsBtype com_type);

1834 **14.1.2 Argument**

1835 **Name:** com_type

1836 **Type:** CelfMpCsBtype

1837 **I/O:** I

1838 **Description:**

1839 Communication type as defined in section 1.2.4.

1841 **14.1.3 Return Value**

1842 **Type:** CelfMpStatus

1843 **Description:**

1844 celf_mp_cs_call_hold() shall return one of the values defined:

1845 CELF_MP_STATUS_OK: successful completion

1846 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1847 CELF_MP_STATUS_ERR: Other unsuccessful completion.

1849 **14.1.4 Include File**

1850 /usr/include/celf/mp_cs.h

1852 **14.1.5 Functional Description**

1853 This function receives the requests response hold.

1854
1855 Because the function is an immediate return function, to confirm the complete result,
1856 including the negotiation with the network, it should be issued
1857 celf_mp_cs_notification_status() to obtain the communication status.

1858
1859 This response hold is performed for an incoming call, only when the communication status
1860 is under incoming.

1861

1862
1863

To release response hold (move to the under conversation status) call "Response to an incoming call".

DRAFT

1864 **15.Call Reject**

1865 **15.1 Symbol: celf_mp_cs_call_reject**

1866 **15.1.1 Syntax**

```
1867 CelfMpStatus celf_mp_cs_call_reject (  
1868 CelfMpCsBtype com_type);  
1869
```

1870 **15.1.2 Argument**

1871 **Name:** com_type

1872 **Type:** CelfMpCsBtype

1873 **I/O:** I

1874 **Description:**

1875 Communication type as defined in section 1.2.4.
1876

1877 **15.1.3 Return Value**

1878 **Type:** CelfMpStatus

1879 **Description:**

1880 celf_mp_cs_call_reject() shall return one of the values defined:

1881 CELF_MP_STATUS_OK: successful completion

1882 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1883 CELF_MP_STATUS_ERR: Other unsuccessful completion.
1884

1885 **15.1.4 Include File**

1886 /usr/include/celf/mp_cs.h
1887

1888 **15.1.5 Functional Description**

1889 This function receives the request to reject an incoming call.
1890

1891 Because the function is an immediate return function, to confirm the complete result,
1892 including the negotiation with the network, it should be issued
1893 celf_mp_cs_notification_start() to obtain the communication status.
1894

1895 The operation for each communication status is as follows:

1896 incoming: Rejects an incoming call

1897 In conversation and incoming: Rejects an incoming call

1898 In hold and incoming: Rejects an incoming call

1899 In conversation, hold, and incoming: Rejects an incoming call

DRAFT

1900 **16.Multi Party Call**

1901 **16.1 Symbol: celf_mp_cs_mp_call**

1902 **16.1.1 Syntax**

```
1903 CelfMpStatus celf_mp_cs_mp_call (  
1904     CelfMpCsBtype      com_type,  
1905     CelfMpCsMop        mode,  
1906     CelfMpCallRef      call_reference);  
1907
```

1908 **16.1.2 Argument**

1909 **Name:** com_type

1910 **Type:** CelfMpCsBtype

1911 **I/O:** |

1912 **Description:**

1913 Communication type as defined in section 1.2.4.

1915 **Name:** mode

1916 **Type:** CelfMpCsMop

1917 **I/O:** |

1918 **Description:**

1919 Operation type

1920 CELF_MP_CS_MOP_RSV_DISC: Disconnect the hold call

1921 CELF_MP_CS_MOP_DISC_AND_RSP: Response after disconnection

1922 CELF_MP_CS_MOP_RSV_AND_RSP: Response after hold (including operation for
1923 switching a call)

1924 CELF_MP_CS_MOP_CR_DISC: Disconnect call specified by the call reference

1926 **Name:** call_reference

1927 **Type:** CelfMpCallRef

1928 **I/O:** |

1929 **Description:**

1930 Call reference of the call to be disconnected

1931 Valid only if CELF_MP_CS_MOP_CR_DISC is specified for the second argument.

1932

1933

1934 **16.1.3 Return Value**

1935 **Type:** CelfMpStatus

1936 **Description:**

1937 `celf_mp_cs_mp_call()` **shall** return one of the values defined:

1938 CELF_MP_STATUS_OK: successful completion

1939 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

1940 CELF_MP_STATUS_ERR: Other unsuccessful completion.

1941

1942 **16.1.4 Include File**

1943 `/usr/include/celf/mp_cs.h`

1944

1945 **16.1.5 Functional Description**

1946 This function receives the request to operate for each call, when communication is made
1947 with multiple calls.

1948

1949 The operation is as follows depending on CELF_MP_CS_MOP:

1950 - CELF_MP_CS_MOP_RSV_DISC

1951 If a hold call exists, this hold call is disconnected.

1952

1953 - CELF_MP_CS_MOP_DISC_AND_RSP

1954 If a conversation call exists and if another call status is incoming or hold, the conversation
1955 call transits to disconnect status and another call transits to conversation status.

1956 See detail below.

1957 (1) Under conversation and incoming

1958 This status is that 1st call is in conversation, and 2nd call is incoming.

1959 The result is that 1st call is released, and 2nd call is conversation.

1960 (2) Under conversation and hold

1961 This status is that 1st call is in conversation, and 2nd call is hold.

1962 The result is that 1st call is released, and 2nd call is conversation.

1963 (3) Under conversation, hold, and incoming

1964 This status is that 1st call is in conversation, that 2nd call is hold, and that 3rd call is
1965 incoming.

1966 The result is that 1st call is released, that 2nd call maintains hold, and that 3rd call is
1967 conversation.

1968 (4) Under response hold

1969 This status is not changed.

1970

Classification: *Circuit-Switched Service*

1971 - CELF_MP_CS_MOP_RSV_AND_RSP
1972 If a conversation call exists and if another call status is incoming or hold,
1973 the conversation call transits to hold status and another call transits to conversation
1974 status.
1975 See detail below.
1976 (1) Under conversation and incoming
1977 This status is that 1st call is in conversation, and 2nd call is incoming.
1978 The result is that 1st call is hold, and 2nd call is conversation.
1979 (2) Under conversation and hold
1980 This status is that 1st call is in conversation, and 2nd call is hold.
1981 The result is that 1st call is hold, and 2nd call is conversation.
1982 (3) Under conversation, hold, and incoming
1983 This status is that 1st call is in conversation, that 2nd call is hold, and that 3rd call is
1984 incoming.
1985 The result is that 1st call is hold, that 2nd call is in conversation, that 3rd call maintains
1986 incoming.
1987 (4) Under response hold
1988 This status is that 1st call is hold.
1989 The result is that 1st call is in conversation.
1990
1991 - CELF_MP_CS_MOP_CR_DISC It is disconnect the call specified the call reference.
1992
1993 Because the function is an immediate return function, to confirm the complete result,
1994 including the negotiation with the network, it should be issued
1995 `self_mp_cs_notification_start()` to obtain the communication status.

1996

17. On-Hook Originating

1997

17.1 Symbol: `celf_mp_cs_originating_on_hook`

1998

17.1.1 Syntax

1999

```
CelfMpStatus celf_mp_cs_originating_on_hook (
```

2000

```
    CelfMpAppId      app_id,
```

2001

```
    CelfMpCsConReq  con_req);
```

2002

2003

17.1.2 Argument

2004

Name: `app_id`

2005

Type: `CelfMpAppId`

2006

I/O: `I`

2007

Description:

2008

Application identifier.

2009

2010

Name: `con_req`

2011

Type: `CelfMpCsConReq`

2012

I/O: `I`

2013

Description:

2014

Communication request type as defined in section 1.3.9.

2015

2016

17.1.3 Return Value

2017

Type: `CelfMpStatus`

2018

Description:

2019

`celf_mp_cs_call_reject()` shall return one of the values defined:

2020

`CELF_MP_STATUS_OK:` successful completion

2021

`CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid

2022

`CELF_MP_CS_ONHOOK_DENY:` On-hook originating is impossible

2023

`CELF_MP_CS_ONHOOK_STATUS_ERR:` Error due to communication conflict

2024

`CELF_MP_CS_ONHOOK_OB_CR:` Excess of the maximum number of calls

2025

`CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2026

2027

17.1.4 Include File

2028

`/usr/include/celf/mp_cs.h`

2029

2030 **17.1.5 Functional Description**

2031 This function receives the request to start an outgoing call with the specified dial number.
2032 The communication status should be Standby.

2033
2034 The dial number is specified by "dial_buf" and "subaddr_buf" in the "con_req" structure.
2035

2036 If the character string, "184" or "186", is placed at the head of dial data, this character
2037 string is deleted.

2038 Whether the originating dial number is notified or not, it is identified by "notice".

2039
2040 The dial data and subaddress stores the following ASCII codes.

- 2041 1 : 0 x 31 2 : 0 x 32 3 : 0 x 33
2042 4 : 0 x 34 5 : 0 x 35 6 : 0 x 36
2043 7 : 0 x 37 8 : 0 x 38 9 : 0 x 39
2044 * : 0 x 2a 0 : 0 x 30 # : 0 x 23

2045
2046 Because the function is an immediate return function, to confirm the complete result,
2047 including the negotiation with the network, it should be issued
2048 `celf_mp_cs_notification_start()` to obtain the communication status.

2049
2050 The originating request during low voltage is disabled.

2051 **18. Get Call Reference**

2052 **18.1 Symbol: celf_mp_cs_get_call_reference**

2053 **18.1.1 Syntax**

```
2054 CelfMpStatus celf_mp_cs_get_call_reference (  
2055     CelfMpAppId      app_id,  
2056     CelfMpCsChanNum channel_num);
```

2057

2058 **18.1.2 Argument**

2059 **Name:** app_id

2060 **Type:** CelfMpAppId

2061 **I/O:** I

2062 **Description:**

2063 Application identifier.

2064

2065 **Name:** channel_num

2066 **Type:** CelfMpCsChanNum

2067 **I/O:** O

2068 **Description:**

2069 Channel number information as defined in section 1.2.4.

2070

2071 **18.1.3 Return Value**

2072 **Type:** CelfMpStatus

2073 **Description:**

2074 `celf_mp_cs_get_call_reference()` **shall** return one of the values defined:

2075 CELF_MP_STATUS_OK: successful completion

2076 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid

2077 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2078

2079 **18.1.4 Include File**

2080 `/usr/include/celf/mp_cs.h`

2081

2082 **18.1.5 Functional Description**

2083 This function gets the call reference in use.

2084

Classification: *Circuit-Switched Service*

2085 A value within 0 to 255 is set to "ChanNum_00", "ChanNum_01" and "ChanNum_02". If
2086 channel is not used, CELF_MP_CS_CHAN_NOUSE is set as the call reference.
2087
2088 Three channels correspond to three calls in multiple calls.

DRAFT

2089 **19.Start DCF message notification**2090 **19.1 Symbol: celf_mp_cs_DCF_notification_start**2091 **19.1.1 Syntax**

```

2092 CelfMpStatus celf_mp_cs_DCF_notification_start (
2093     CelfMpAppld      app_id,
2094     CelfMpDCFSet     event_set,
2095     CelfMpCallback   callback_func);
2096 
```

2097 **19.1.2 Argument**2098 **Name:** app_id2099 **Type:** CelfMpAppld2100 **I/O:** I2101 **Description:**

2102 Application identifier.

2103

2104 **Name:** event_set2105 **Type:** CelfMpCsDCFSet2106 **I/O:** I2107 **Description:**

2108 Notification event set. Events that are classified as belonging to one of the
 2109 `CelfMpCsDCFSet` class **may** be registered to have a callback function called when the
 2110 event occurs for the application identified by `app_id`. Classes of events are enabled by
 2111 setting the corresponding bit in `event_set`:

2112

2113 The event classes are defined as follows:

2114 `CELF_MP_CS_DCF_DISP` Display-related message2115 `CELF_MP_CS_DCF_HISTORY` History-related message2116 `CELF_MP_CS_DCF_TONE1` Tone 1-related message2117 `CELF_MP_CS_DCF_TONE2` Tone 2-related message2118 `CELF_MP_CS_DCF_ETC` Other messages2119 `CELF_MP_CS_CLASS_ALL` All notified

2120

2121 A callback **may** be registered for all classes of events using special event class
 2122 `CELF_MP_CS_CLASS_ALL`, however to reduce overhead it is recommended that only the
 2123 needed event classes **should** be registered.

2124

2125 **Name:** callback_func

2126 **Type:** CelfMpCallback

2127 **I/O:** I

2128 **Description:**

2129 The callback function, which **shall** be called when an event occurs from one of the classes
2130 in `event_set`.

2131

2132 19.1.3 Return Value

2133 **Type:** CelfMpStatus

2134 **Description:**

2135 `celf_mp_cs_DCF_notification_start ()` **shall** return one of the values defined:

2136 CELF_MP_STATUS_OK: successful completion

2137 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid

2138 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2139

2140 19.1.4 Include File

2141 `/usr/include/celf/mp_cs.h`

2142

2143 19.1.5 Functional Description

2144 This function starts the monitoring the DCF message on the voice communication or AV
2145 communication.

2146

2147 The occurrence of the event is notified to the application, specified by `app_id`.

2148

2149 The messages to be notified are described below.

2150

2151 Display-related message:

2152 -Notification of starting display during CCP outgoing

2153 -Notification of starting display during CCP incoming

2154 -Notification of starting display during CCP calling

2155 -Notification of starting display during CCP connecting

2156 -Notification of starting display during CCP communication

2157 -Notification of ending CCP That is to notifies of release of a CCP call.

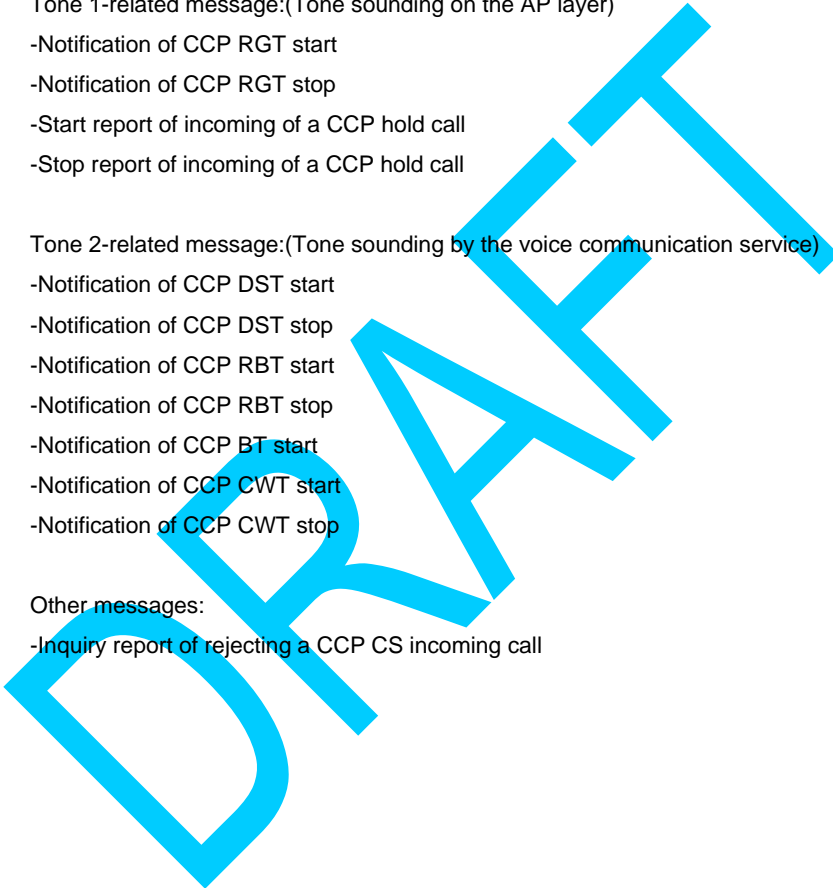
2158 -Notification of starting CCP disconnection (on the mobilephone) display

2159 -Notification of starting display of CCP disconnection (on the network) display

2160 -Notification of rejecting CCP outgoing

2161 -Notification of CCP hold

- 2162 -Notification of releasing CCP hold
- 2163
- 2164 History-related message:
- 2165 -Notification of registering CCP outgoing call history
- 2166 -Notification of registering CCP absence incoming call history
- 2167 -Notification of registering CCP incoming call history
- 2168
- 2169 Tone 1-related message:(Tone sounding on the AP layer)
- 2170 -Notification of CCP RGT start
- 2171 -Notification of CCP RGT stop
- 2172 -Start report of incoming of a CCP hold call
- 2173 -Stop report of incoming of a CCP hold call
- 2174
- 2175 Tone 2-related message:(Tone sounding by the voice communication service)
- 2176 -Notification of CCP DST start
- 2177 -Notification of CCP DST stop
- 2178 -Notification of CCP RBT start
- 2179 -Notification of CCP RBT stop
- 2180 -Notification of CCP BT start
- 2181 -Notification of CCP CWT start
- 2182 -Notification of CCP CWT stop
- 2183
- 2184 Other messages:
- 2185 -Inquiry report of rejecting a CCP CS incoming call



2186 20. Stop DCF message notification

2187 20.1 Symbol: `celf_mp_cs_DCF_notification_stop`

2188 20.1.1 Syntax

```
2189 CelfMpStatus celf_mp_cs_DCF_notification_stop(  
2190     CelfMpAppId      app_id,  
2191     CelfMpDCFSet     event_set);
```

2192

2193 20.1.2 Argument

2194 **Name:** `app_id`

2195 **Type:** `CelfMpAppId`

2196 **I/O:** `I`

2197 **Description:**

2198 Application identifier.

2199

2200 **Name:** `event_set`

2201 **Type:** `CelfMpCsDCFSet`

2202 **I/O:** `I`

2203 **Description:**

2204 Notification event set. Events that are classified as belonging to one of the
2205 `CelfMpCsDCFSet` class. Classes of events are enabled by setting the corresponding bit in
2206 `event_set`:

2207

2208 The event classes are defined as follows:

2209 `CELf_MP_CS_DCF_DISP` Display-related message

2210 `CELf_MP_CS_DCF_HISTORY` History-related message

2211 `CELf_MP_CS_DCF_TONE1` Tone 1-related message

2212 `CELf_MP_CS_DCF_TONE2` Tone 2-related message

2213 `CELf_MP_CS_DCF_ETC` Other messages

2214 `CELf_MP_CS_CLASS_ALL` All notified

2215

2216 20.1.3 Return Value

2217 **Type:** `CelfMpStatus`

2218 **Description:**

2219 `celf_mp_cs_DCF_notification_stop()` shall return one of the values defined:

2220 `CELf_MP_STATUS_OK`: successful completion

2221 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid
2222 CELF_MP_STATUS_ERR: Other unsuccessful completion.
2223

2224 20.1.4 Include File

2225 /usr/include/celf/mp_cs.h

2226

2227 20.1.5 Functional Description

2228 This function stops notifying of the DCF message on voice communication or AV
2229 communication.

DRAFT

2230 21.Voice Message Notification

2231 21.1 Symbol: `celf_mp_cs_voice_msg_notify`

2232 21.1.1 Syntax

```
2233 CelfMpStatus celf_mp_cs_voice_msg_notify (  
2234 CelfMpCsRecMsg rec_status);  
2235
```

2236 21.1.2 Argument

2237 **Name:** `rec_status`

2238 **Type:** `CelfMpCsRecMsg`

2239 **I/O:** |

2240 **Description:**

2241 CELF_MP_CS_REC_MSG_START: Start of a voice message

2242 CELF_MP_CS_REC_MSG_STOP: Stop of a voice message

2243

2244 21.1.3 Return Value

2245 **Type:** `CelfMpStatus`

2246 **Description:**

2247 `celf_mp_cs_call_voice_msg_notify()` **shall** return one of the values defined:

2248 CELF_MP_STATUS_OK: successful completion

2249 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

2250 CELF_MP_STATUS_REC_STAT_ERR: Recording failed

2251 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2252

2253 21.1.4 Include File

2254 `/usr/include/celf/mp_cs.h`

2255

2256 21.1.5 Functional Description

2257 This function must be called before the communication state is changed to "under
2258 conversation."

2259 After the start notification, a stop notification **must** be issued, when the voice message is
2260 stopped.

2261 **22.Hold Tone Start**

2262 **22.1 Symbol: celf_mp_cs_hold_tone_start**

2263 22.1.1 Syntax

```
2264 CelfMpStatus celf_mp_cs_hold_tone_start (  
2265     CelfMpAppld app_id);  
2266
```

2267 22.1.2 Argument

2268 **Name:** app_id

2269 **Type:** CelfMpAppld

2270 **I/O:** I

2271 **Description:**

2272 Application identifier.
2273

2274 22.1.3 Return Value

2275 **Type:** CelfMpStatus

2276 **Description:**

2277 `celf_mp_cs_hold_tone_start()` shall return one of the values defined:

2278 CELF_MP_STATUS_OK: successful completion

2279 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

2280 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid

2281 CELF_MP_STATUS_ERR: Other unsuccessful completion.
2282

2283 22.1.4 Include File

2284 `/usr/include/celf/mp_cs.h`
2285

2286 22.1.5 Functional Description

2287 This function starts to sound a hold tone during a call.

2288 **23.Hold Tone Stop**

2289 **23.1 Symbol: celf_mp_cs_hold_tone_stop**

2290 **23.1.1 Syntax**

2291 CelfMpStatus celf_mp_cs_hold_tone_stop (
2292 CelfMpAppld app_id);

2294 **23.1.2 Argument**

2295 **Name:** app_id

2296 **Type:** CelfMpAppld

2297 **I/O:** I

2298 **Description:**

2299 Application identifier.

2301 **23.1.3 Return Value**

2302 **Type:** CelfMpStatus

2303 **Description:**

2304 celf_mp_cs_hold_tone_stop() **shall return one of the values defined:**

2305 CELF_MP_STATUS_OK: successful completion

2306 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid

2307 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid

2308 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2310 **23.1.4 Include File**

2311 /usr/include/celf/mp_cs.h

2313 **23.1.5 Functional Description**

2314 This function stops to sound a hold tone during a call.

2315

2316 **24. Get 64K / AV Communication Status**

2317 **24.1 Symbol: celf_mp_cs_get_UD_com_stat**

2318 **24.1.1 Syntax**

2319 CelfMpStatus celf_mp_cs_get_UD_com_stat (
2320 CelfMpUDComStatus res_status);
2321

2322 **24.1.2 Argument**

2323 **Name:** res_status

2324 **Type:** CelfMpUDComStatus

2325 **I/O:** O

2326 **Description:**

2327 returns one of the values defined:

- | | |
|-----------------------------------|---------------------------|
| 2328 CELF_MP_CS_UD_STOP: | Under stop |
| 2329 CELF_MP_CS_UD_RUN: | Under communication |
| 2330 CELF_MP_CS_UD_CALLED: | Under incoming |
| 2331 CELF_MP_CS_UD_CALLING: | Under outgoing |
| 2332 CELF_MP_CS_UD_DISCONNECT: | Under disconnection |
| 2333 CELF_MP_CS_UD_CALLING_ALERT: | Under calling |
| 2334 CELF_MP_CS_UD_HOLD: | Under hold |
| 2335 CELF_MP_CS_UD_ERR: | Error in UD Communication |

2337 **24.1.3 Return Value**

2338 **Type:** CelfMpStatus

2339 **Description:**

2340 celf_mp_cs_get_UD_com_stat() **shall** return one of the values defined:

- | | |
|--------------------------|--------------------------------|
| 2341 CELF_MP_STATUS_OK: | successful completion |
| 2342 CELF_MP_STATUS_ERR: | Other unsuccessful completion. |

2344 **24.1.4 Include File**

2345 /usr/include/celf/mp_cs.h
2346

2347 **24.1.5 Functional Description**

2348 This function refers to the communication status of 64K communication or AV communication.

2349 25. Get internal/external AV Communication Status

2350 25.1 Symbol: `celf_mp_cs_get_AV_com_stat`

2351 25.1.1 Syntax

```
2352 CelfMpStatus celf_mp_cs_get_AV_com_stat (  
2353 CelfMpUDComStatus res_status);
```

2354

2355 25.1.2 Argument

2356 **Name:** `res_status`

2357 **Type:** `CelfMpUDComStatus`

2358 **I/O:** O

2359 **Description:**

2360 returns one of the values defined:

2361	<code>CELF_MP_CS_AV_IN_STOP:</code>	Under stop
2362	<code>CELF_MP_CS_AV_IN_RUN:</code>	Under communication
2363	<code>CELF_MP_CS_AV_IN_CALLED:</code>	Under incoming
2364	<code>CELF_MP_CS_AV_IN_CALLING:</code>	Under outgoing
2365	<code>CELF_MP_CS_AV_IN_DISCONNECT:</code>	Under disconnection
2366	<code>CELF_MP_CS_AV_IN_CALLING_ALERT:</code>	Under calling
2367	<code>CELF_MP_CS_UD_IN_HOLD:</code>	Under hold
2368	<code>CELF_MP_CS_AV_OUT_STOP:</code>	Under stop
2369	<code>CELF_MP_CS_AV_OUT_RUN:</code>	Under communication
2370	<code>CELF_MP_CS_AV_OUT_CALLED:</code>	Under incoming
2371	<code>CELF_MP_CS_AV_OUT_CALLING:</code>	Under outgoing
2372	<code>CELF_MP_CS_AV_OUT_DISCONNECT:</code>	Under disconnection
2373	<code>CELF_MP_CS_AV_OUT_CALLING_ALERT:</code>	Under calling
2374	<code>CELF_MP_CS_UD_OUT_HOLD:</code>	Under hold

2375

2376 25.1.3 Return Value

2377 **Type:** `CelfMpStatus`

2378 **I/O:** O

2379 **Description:**

2380 `celf_mp_cs_get_AV_com_stat()` **shall** return one of the values defined:

2381 `CELF_MP_STATUS_OK:` successful completion

2382 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2383

2384

2385 25.1.4 Include File

2386 /usr/include/celf/mp_cs.h

2387

2388 25.1.5 Functional Description

2389 This function refers to the communication status of internal or external AV communication.

DRAFT

2390 **26. Get Communication Status**

2391 **26.1 Symbol: celf_mp_cs_get_com_stat**

2392 **26.1.1 Syntax**

```
2393 CelfMpStatus celf_mp_cs_get_com_stat (  
2394     CelfMpAppld      app_id,  
2395     CelfMpCsRcvScene * rcv_type,  
2396     CelfMpCsComStatus res_status);  
2397
```

2398 **26.1.2 Argument**

2399 **Name:** app_id

2400 **Type:** CelfMpAppld

2401 **I/O:** I

2402 **Description:**

2403 Application identifier.

2404

2405 **Name:** rcv_type

2406 **Type:** CelfMpCsRcvType

2407 **I/O:** O

2408 **Description:**

2409 Incoming call type:

2410 CELF_MP_CS_RCV_TYPE_COMPETE_TRN: Communication Conflict

2411 CELF_MP_CS_RCV_TYPE_RSV_RETURN: Reestablish held call

2412 CELF_MP_CS_RCV_TYPE_CALL_BACK: Network Call Back

2413 CELF_MP_CS_RCV_TYPE_NORMAL: Normal

2414 CELF_MP_CS_RCV_TYPE_NONE: No Incoming Call

2415

2416 **Name:** res_status

2417 **Type:** CelfMpCsComStatus

2418 **I/O:** O

2419 **Description:**

2420 returns one of the values defined:

2421 Current communication status

2422 CELF_MP_CS_COM_STATUS_WAIT: Standby

2423 CELF_MP_CS_COM_STATUS_RCV: Under incoming

2424 CELF_MP_CS_COM_STATUS_TRN: Under outgoing

Classification: *Circuit-Switched Service*

2425	CELF_MP_CS_COM_STATUS_DLV:	Under calling
2426	CELF_MP_CS_COM_STATUS_TLK:	Under conversation
2427	CELF_MP_CS_COM_STATUS_HLD:	Under response hold
2428	CELF_MP_CS_COM_STATUS_DUMMY1:	Under off-hook
2429	CELF_MP_CS_COM_STATUS_RLS:	Under release
2430	CELF_MP_CS_COM_STATUS_TLK_RCV:	Under conversation and incoming
2431	CELF_MP_CS_COM_STATUS_TLK_TRN:	Under conversation and outgoing
2432	CELF_MP_CS_COM_STATUS_TLK_DLV:	Under conversation and calling
2433	CELF_MP_CS_COM_STATUS_TLK_RSV:	Under conversation and hold
2434	CELF_MP_CS_COM_STATUS_TLK_RLS:	Under conversation and release
2435	CELF_MP_CS_COM_STATUS_TLK_RSV_RCV:	Under conversation, hold, and incoming
2436	CELF_MP_CS_COM_STATUS_RCV_AV:	Under incoming of an AV call
2437	CELF_MP_CS_COM_STATUS_TRN_AV:	Under outgoing of an AV call
2438	CELF_MP_CS_COM_STATUS_DLV_AV:	Under calling of an AV call
2439	CELF_MP_CS_COM_STATUS_TLK_AV:	Under conversation of an AV call
2440	CELF_MP_CS_COM_STATUS_HLD_AV:	Under response hold of an AV call
2441	CELF_MP_CS_COM_STATUS_RLS_AV:	Under release of an AV call
2442	CELF_MP_CS_COM_STATUS_DUMMY2:	Under AV off-hook
2443	CELF_MP_STATUS_APP_ID_ERR:	Application ID is not valid.
2444	CELF_MP_CS_ERR:	Abnormal end
2445		

26.1.3 Return Value

2447 **Type:** CelfMpStatus

2448 **I/O:** O

2449 **Description:**

2450 `celf_mp_cs_get_com_stat()` shall return one of the values defined:

2451 CELF_MP_STATUS_OK: successful completion

2452 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2453

26.1.4 Include File

2454 `/usr/include/celf/mp_cs.h`

2456

26.1.5 Functional Description

2458 This function returns the incoming call status, when the current call is (a) under incoming
 2459 status or (b) under conversation and incoming status.

2460

2461

27. Start Line Status Notification

2462

27.1 Symbol: `celf_mp_cs_line_status_notification_start`

2463

27.1.1 Syntax

2464

```
CelfMpStatus celf_mp_cs_notification_start (
```

2465

```
    CelfMpAppId      app_id,
```

2466

```
    CelfMpCsMtype    event_set,
```

2467

```
    CelfMpCallback   callback_func);
```

2468

2469

27.1.2 Argument

2470

Name: `app_id`

2471

Type: `CelfMpAppId`

2472

I/O: |

2473

Description:

2474

Application identifier.

2475

2476

Name: `event_set`

2477

Type: `CelfMpCsMtype`

2478

I/O: |

2479

Description:

2480

Notification event set. Events that are classified as belonging to one of the

2481

`CelfMpCsNotifySet` class **may** be registered to have a callback function called when

2482

the event occurs for the application identified by `app_id`. Classes of events are enabled by

2483

setting the corresponding bit in `event_set`.

2484

`CELf_MP_CS_MONITOR_LINE_STATUS:` Line status change notification

2485

`CELf_MP_CS_MONITOR_RESTRICT:` Restriction status change notification

2486

`CELf_MP_CS_MONITOR_RSSI:` Receive level change notification

2487

`CELf_MP_CS_MONITOR_ALL:` All notified

2488

2489

Name: `callback_func`

2490

Type: `CelfMpCallback`

2491

I/O: |

2492

Description:

2493

The callback function, which **shall** be called when an event occurs from one of the classes

2494

in `event_set`.

2495

2496 **27.1.3 Return Value**

2497 **Type:** CelfMpStatus

2498 **Description:**

2499 `celf_mp_cs_notification_start()` **shall** return one of the values defined:

- | | | |
|------|------------------------------|--------------------------------|
| 2500 | CELF_MP_STATUS_OK: | successful completion |
| 2501 | CELF_MP_STATUS_APP_ID_ERR: | Application ID is not valid |
| 2502 | CELF_MP_STATUS_MON_TYPE_ERR: | Monitor type is not valid |
| 2503 | CELF_MP_STATUS_ERR: | Other unsuccessful completion. |

2504

2505 **27.1.4 Include File**

2506 `/usr/include/celf/mp_cs.h`

2507

2508 **27.1.5 Functional Description**

2509 This function starts the monitoring the line status.

2510 The occurrence of the event is notified to the application, specified by `app_id`.

2511 The events to be notified are described below.

2512

2513 1. Line status change notification:

2514 This event notifies that the line status is changed.

2515 The line status is the out-of-communication area status and the within-communication
2516 area.

2517

2518 2. Restriction status change notification:

2519 This event notifies that a restriction status is changed.

2520 The restriction means that the incoming call or the outgoing call is restricted by the network
2521 due to traffic congestion.

2522

2523 3. Receive level change notification:

2524 This event notifies that the receive level is changed.

2525 The receive level is the intensity of electromagnetic wave. The intensity is four levels,
2526 high, mid, low and zero (out of area).

2527

2528 See section 1. for structure definitions and values.

2529

2530

28. Stop Line Status Notification

2531

28.1 Symbol: `celf_mp_cs_line_status_notification_stop`

2532

28.1.1 Syntax

2533

```
CelfMpStatus celf_mp_cs_notification_stop (
```

2534

```
    CelfMpAppId      app_id,
```

2535

```
    CelfMpCsMtype    event_set);
```

2536

2537

28.1.2 Argument

2538

Name: `app_id`

2539

Type: `CelfMpAppId`

2540

I/O: |

2541

Description:

2542

Application identifier.

2543

2544

Name: `event_set`

2545

Type: `CelfMpCsMtype`

2546

I/O: |

2547

Description:

2548

Mask of the events for which reporting is to be stopped.

2549

Notification event set. Events that are classified as belonging to one of the

2550

`CelfMpCsNotifySet` class **may** be registered to have a callback function called when

2551

the event occurs for the application identified by `app_id`. Classes of events are enabled by

2552

setting the corresponding bit in `event_set`:

2553

`CELF_MP_CS_MONITOR_LINE_STATUS:` Line status change notification

2554

`CELF_MP_CS_MONITOR_RESTRICT:` Restriction status change notification

2555

`CELF_MP_CS_MONITOR_RSSI:` Received signal strength change

2556

notification

2557

`CELF_MP_CS_MONITOR_ALL:` All notified

2558

2559

28.1.3 Return Value

2560

Type: `CelfMpStatus`

2561

Description:

2562

`celf_mp_cs_notification_stop()` **shall** return one of the values defined:

2563

`CELF_MP_STATUS_OK:` successful completion

2564

`CELF_MP_STATUS_APP_ID_ERR:` Application ID is not valid

2565

`CELF_MP_STATUS_MON_TYPE_ERR:` Monitor type is not valid

2566 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2567

2568 28.1.4 Include File

2569 /usr/include/celf/mp_cs.h

2570

2571 28.1.5 Functional Description

2572 This function ends notifying on the event of the line status.

2573

DRAFT

2574 **29. Get Reception Level**

2575 **29.1 Symbol: celf_mp_cs_get_reception_level**

2576 **29.1.1 Syntax**

```
2577 CelfMpStatus celf_mp_cs_get_reception_level (  
2578 CelfMpReceptionLevel result);
```

2580 **29.1.2 Argument**

2581 **Name:** result

2582 **Type:** CelfMpReceptionLevel

2583 **I/O:** O

2584 **Description:**

2585 returns one of the values defined:

- 2586 CELF_MP_CS_RSSI_LEVEL_NONE : No reception
- 2587 CELF_MP_CS_RSSI_LEVEL_LOW : Receive level (Lowest)
- 2588 CELF_MP_CS_RSSI_LEVEL_MEDIUM1 : Receive level
- 2589 CELF_MP_CS_RSSI_LEVEL_MEDIUM2 : Receive level
- 2590 CELF_MP_CS_RSSI_LEVEL_HIGH : Receive level (Highest)

2592 **29.1.3 Return Value**

2593 **Type:** CelfMpStatus

2594 **I/O:** O

2595 **Description:**

2596 `celf_mp_cs_get_reception_level()` **shall** return one of the values defined:

- 2597 CELF_MP_STATUS_OK: successful completion
- 2598 CELF_MP_STATUS_APP_ID_ERR: Application ID is not valid
- 2599 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2601 **29.1.4 Include File**

2602 `/usr/include/celf/mp_cs.h`

2604 **29.1.5 Functional Description**

2605 This function obtains the current reception level.

2606 Without the line status monitoring by calling the “Start line status monitoring”, it is possible
2607 to get the status of reception level.

2608

2609 **30. Get Line Status**

2610 **30.1 Symbol: celf_mp_cs_get_line_status**

2611 **30.1.1 Syntax**

```
2612 CelfMpStatus celf_mp_cs_get_line_status (  
2613     CelfMpCsAreaRefChgInf * net);  
2614
```

2615 **30.1.2 Argument**

2616 **Name:** net

2617 **Type:** CelfMpCsAreaRefChgInf

2618 **I/O:** I

2619 **Description:**

2620 Pointer to the struct used to hold line status information
2621

2622 **30.1.3 Return Value**

2623 **Type:** CelfMpStatus

2624 **Description:**

2625 celf_mp_cs_get_line_status() shall return one of the values defined:

2626 CELF_MP_STATUS_OK: successful completion

2627 CELF_MP_STATUS_ERR: Other unsuccessful completion.
2628

2629 **30.1.4 Include File**

2630 /usr/include/celf/mp_cs.h
2631

2632 **30.1.5 Functional Description**

2633 This function obtains the current line status.

2634 Without the line status monitoring by calling the “Start line status monitoring”, it is possible
2635 to get the status of line status.

2636 See section 1. for further information.
2637

2638 **31. Get Coverage Status**

2639 **31.1 Symbol: celf_mp_cs_get_coverage_status**

2640 **31.1.1 Syntax**

```
2641 CelfMpStatus celf_mp_cs_get_line_status (  
2642     CelfMpCsLineStatusEx* net,  
2643     CelfMpCsCoverage cover);  
2644
```

2645 **31.1.2 Argument**

2646 **Name:** net

2647 **Type:** CelfMpCsLineStatusEx

2648 **I/O:** I

2649 **Description:**

2650 Pointer to the struct used to hold line status information

2652 **Name:** cover

2653 **Type:** CelfMpCsCoverage

2654 **I/O:** O

2655 **Description:**

2656 Within- or out-of communication area status

2657 CELF_MP_CS_LINE_STATUS_IN: Within-communication area

2658 CELF_MP_CS_LINE_STATUS_OUT: Out-of-communication area

2660 **31.1.3 Return Value**

2661 **Type:** CelfMpStatus

2662 **Description:**

2663 celf_mp_cs_get_line_status() **shall** return one of the values defined:

2664 CELF_MP_STATUS_OK: successful completion

2665 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2667 **31.1.4 Include File**

2668 /usr/include/celf/mp_cs.h

2669

2670

31.1.5 Functional Description

2671

This function obtains the information on the current status of the within- and out-of-communication areas for current line.

2672

2673

(This function gets only information of inside or outside coverage area status.)

DRAFT

2674 32. Get Voice Mail Information

2675 32.1 Symbol: `celf_mp_cs_get_vm_info`

2676 32.1.1 Syntax

```
2677 CelfMpStatus celf_mp_cs_get_vm_info (  
2678 CelfMpCsVMNum * vm_num);
```

2679 32.1.2 Argument

2680 **Name:** `vm_num`

2681 **Type:** `CelfMpCsVMNum`

2682 **I/O:** |

2683 **Description:**

2684 Address of the storage area of the number of stored phone-answering messages

2685

2686 32.1.3 Return Value

2687 **Type:** `CelfMpStatus`

2688 **Description:**

2689 `celf_mp_cs_get_vm_info()` shall return one of the values defined:

2690 `CELF_MP_STATUS_OK:` successful completion

2691 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2692

2693 32.1.4 Include File

2694 `/usr/include/celf/mp_cs.h`

2695

2696 32.1.5 Functional Description

2697 This function obtains the storage status of phone-answering messages from nonvolatile
2698 memory.

2699 The storage status is the number of message of phone-answering.

2700

2701 **33.Set Voice Mail Information**

2702 **33.1 Symbol: celf_mp_cs_set_vm_info**

2703 **33.1.1 Syntax**

2704 CelfMpStatus celf_mp_cs_set_vm_info (
2705 CelfMpCsVMNum vm_num);
2706

2707 **33.1.2 Argument**

2708 **Name:** vm_num

2709 **Type:** CelfMpCsVMNum

2710 **I/O:** I

2711 **Description:**

2712 The number of stored phone-answering messages
2713

2714 **33.1.3 Return Value**

2715 **Type:** CelfMpStatus

2716 **Description:**

2717 celf_mp_cs_set_vm_info() **shall** return one of the values defined:

2718 CELF_MP_STATUS_OK: successful completion

2719 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2720 CELF_MP_STATUS_COM_TYPE_ERR: Communication type is not valid
2721

2722 **33.1.4 Include File**

2723 /usr/include/celf/mp_cs.h
2724

2725 **33.1.5 Functional Description**

2726 This function sets the storage status of phone-answering message to non-volatile memory.
2727
2728
2729

2730 **34. Get Call Selection**

2731 **34.1 Symbol: celf_mp_cs_get_call_select**

2732 **34.1.1 Syntax**

2733 CelfMpStatus celf_mp_cs_get_call_select (
2734 CelfMpCallSelect select);

2736 **34.1.2 Argument**

2737 **Name:** select

2738 **Type:** CelfMpCallSelect

2739 **I/O:** O

2740 **Description:**

2741 CELF_MP_CS_INCOMING_VOICE_ANSWERING Forward to the phone-answering
2742 message

2743 CELF_MP_CS_INCOMING_FORWARD Forward

2744 CELF_MP_CS_INCOMING_REJECT Reject (disconnect)

2745 CELF_MP_CS_INCOMING_NORMAL Receipt of an incoming call (normal
2746 incoming)

2747

2748 **34.1.3 Return Value**

2749 **Type:** CelfMpStatus

2750 **Description:**

2751 celf_mp_cs_get_call_select () **shall** return one of the values defined:

2752 CELF_MP_STATUS_OK: successful completion

2753 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2754

2755 **34.1.4 Include File**

2756 /usr/include/celf/mp_cs.h

2757

2758 **34.1.5 Functional Description**

2759 This function obtains the incoming call information from non-volatile memory.

2760 Refer "Set incoming function selection"

2761

2762

2763 **35.Set Call Selection**

2764 **35.1 Symbol: celf_mp_cs_set_call_select**

2765 **35.1.1 Syntax**

2766 CelfMpStatus celf_mp_cs_set_call_select (
2767 CelfMpCallSelect select);
2768

2769 **35.1.2 Argument**

2770 **Name:** select

2771 **Type:** CelfMpCallSelect

2772 **I/O:** I

2773 **Description:**

2774 CELF_MP_CS_INCOMING_VOICE_ANSWERING: Forward to the phone-answering
2775 message

2776 CELF_MP_CS_INCOMING_FORWARD: Forward

2777 CELF_MP_CS_INCOMING_REJECT: Reject (disconnect)

2778 CELF_MP_CS_INCOMING_NORMAL: Receipt of an incoming call (normal
2779 incoming)

2780

2781 **35.1.3 Return Value**

2782 **Type:** CelfMpStatus

2783 **Description:**

2784 celf_mp_cs_set_call_select() **shall** return one of the values defined:

2785 CELF_MP_STATUS_OK: successful completion

2786 CELF_MP_STATUS_ERR: Other unsuccessful completion.
2787

2788 **35.1.4 Include File**

2789 /usr/include/celf/mp_cs.h
2790

2791 **35.1.5 Functional Description**

2792 This function sets the incoming call information to nonvolatile memory.

2793 When an incoming call arrives during conversation mode, it is possible to save this
2794 incoming call information.
2795
2796

2797 **36.Set Service Information**

2798 **36.1 Symbol: celf_mp_cs_set_service_info**

2799 **36.1.1 Syntax**

```
2800 CelfMpStatus celf_mp_cs_set_service_info (  
2801     CelfMpRegNum    reg_no,  
2802     CelfMpCsSrvData * data);  
2803
```

2804 **36.1.2 Argument**

2805 **Name:** reg_no

2806 **Type:** CelfMpRegNum

2807 **I/O:** |

2808 **Description:**

2809 Registration number: 1 to 10

2810

2811 **Name:** data

2812 **Type:** CelfMpCsSrvData

2813 **I/O:** |

2814 **Description:**

2815 Pointer to additional service data

2816

2817 **36.1.3 Return Value**

2818 **Type:** CelfMpStatus

2819 **Description:**

2820 `celf_mp_cs_set_service_info()` **shall** return one of the values defined:

2821 CELF_MP_STATUS_REG_NUM_ERR: reg_no is out of range

2822 CELF_MP_STATUS_OK: successful completion

2823 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2824

2825 **36.1.4 Include File**

2826 `/usr/include/celf/mp_cs.h`

2827

2828 **36.1.5 Functional Description**

2829 This function registers the additional service information to the non-volatile memory,

2830

Classification: *Circuit-Switched Service*

2831 The additional service information is the service name and Dial data for accessing the
2832 service.
2833 The 'reg_no' is used as the key for accessing this additional service.
2834 The value range is from 1 to 10.
2835 See section 1. for additional information.
2836

DRAFT

2837 37. Get Service Information

2838 37.1 Symbol: `celf_mp_cs_get_service_info`

2839 37.1.1 Syntax

```
2840 CelfMpStatus celf_mp_cs_get_service_info (  
2841     CelfMpRegNum    reg_no,  
2842     CelfMpCsSrvData * data);
```

2843

2844 37.1.2 Argument

2845 **Name:** `reg_no`

2846 **Type:** `CelfMpRegNum`

2847 **I/O:** `I`

2848 **Description:**

2849 Registration number: 1 to 10

2850

2851 **Name:** `data`

2852 **Type:** `CelfMpCsSrvData`

2853 **I/O:** `I`

2854 **Description:**

2855 Pointer to additional service data

2856

2857 37.1.3 Return Value

2858 **Type:** `CelfMpStatus`

2859 **Description:**

2860 `celf_mp_cs_get_service_info()` **shall** return one of the values defined:

2861 `CELF_MP_STATUS_REG_NUM_ERR:` `reg_no` is out of range

2862 `CELF_MP_STATUS_OK:` successful completion

2863 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2864

2865 37.1.4 Include File

2866 `/usr/include/celf/mp_cs.h`

2867

2868 37.1.5 Functional Description

2869 This function obtains additional service information, specified by “`reg_no`”, from non-volatile
2870 memory.

2871

2872 See "Register additional service settings".

DRAFT

2873 **38.Delete Service Information**

2874 **38.1 Symbol: celf_mp_cs_del_service_info**

2875 **38.1.1 Syntax**

2876 CelfMpStatus celf_mp_cs_del_service_info (
2877 CelfMpRegNum reg_no);
2878

2879 **38.1.2 Argument**

2880 **Name:** reg_no

2881 **Type:** CelfMpRegNum

2882 **I/O:** I

2883 **Description:**

2884 Registration number: 1 to 10
2885

2886 **38.1.3 Return Value**

2887 **Type:** CelfMpStatus

2888 **Description:**

2889 celf_mp_cs_del_service_info() **shall return one of the values defined:**

2890 CELF_MP_STATUS_REG_NUM_ERR: reg_no is out of range

2891 CELF_MP_STATUS_OK: successful completion

2892 CELF_MP_STATUS_ERR: Other unsuccessful completion.
2893

2894 **38.1.4 Include File**

2895 /usr/include/celf/mp_cs.h
2896

2897 **38.1.5 Functional Description**

2898 This function deletes the additional service information specified by "reg_no" from non-
2899 volatile memory.

2900 **39.Remove Service Information**

2901 **39.1 Symbol: celf_mp_cs_remove_all_service_info**

2902 **39.1.1 Syntax**

2903 CelfMpStatus celf_mp_cs_remove_all_service_info (
2904 void);

2905 **39.1.2 Argument**

2906 None.

2907

2908 **39.1.3 Return Value**

2909 **Type:** CelfMpStatus

2910 **Description:**

2911 celf_mp_cs_remove_all_service_info() **shall** return one of the values defined:

2912 CELF_MP_STATUS_OK: successful completion

2913 CELF_MP_STATUS_ERR: Other unsuccessful completion.

2914

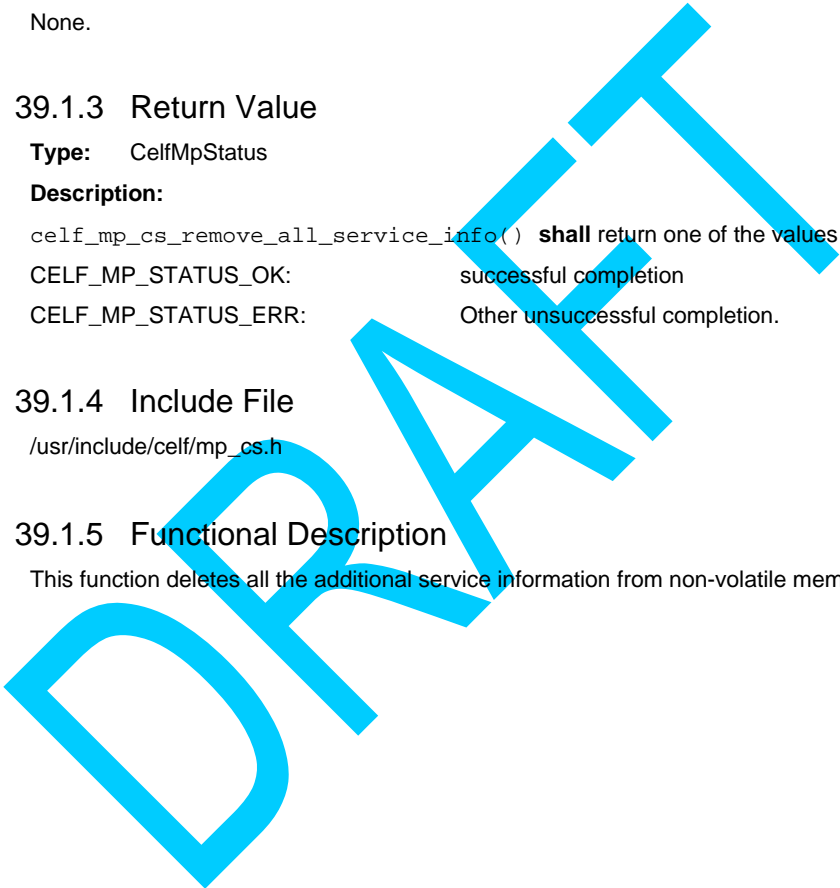
2915 **39.1.4 Include File**

2916 /usr/include/celf/mp_cs.h

2917

2918 **39.1.5 Functional Description**

2919 This function deletes all the additional service information from non-volatile memory.



2920 40. Set Response Message Settings

2921 40.1 Symbol: `celf_mp_cs_set_resp_msg`

2922 40.1.1 Syntax

```
2923 CelfMpStatus celf_mp_cs_set_resp_msg (  
2924     CelfMpRegNum    reg_no,  
2925     CelfMpCsSrvData * data);
```

2927 40.1.2 Argument

2928 **Name:** `reg_no`

2929 **Type:** `CelfMpRegNum`

2930 **I/O:** `I`

2931 **Description:**

2932 Registration number: 1 to 10

2934 **Name:** `data`

2935 **Type:** `CelfMpCsSrvData`

2936 **I/O:** `I`

2937 **Description:**

2938 Pointer to the additional response message data area.

2940 40.1.3 Return Value

2941 **Type:** `CelfMpStatus`

2942 **Description:**

2943 `celf_mp_cs_set_resp_msg()` shall return one of the values defined:

2944 `CELF_MP_STATUS_REG_NUM_ERR:` `reg_no` is out of range

2945 `CELF_MP_STATUS_OK:` successful completion

2946 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2948 40.1.4 Include File

2949 `/usr/include/celf/mp_cs.h`

2950

2951 40.1.5 Functional Description

2952 This function registers the additional response message information for the additional
2953 service to the non-volatile memory,

2954

Classification: *Circuit-Switched Service*

2955 When an additional service is activated, and a corresponding message from the network is
2956 received, this additional response message is sent to the network.
2957
2958 The additional response message information is the service name and Dial data, which is
2959 response message to send the network.
2960
2961 The "reg_no" is used as the key for accessing this additional response message.
2962 The value range is from 1 to 10.
2963
2964 For information about the structures, see section 1.
2965

DRAFT

2966 41. Get Response Message Settings

2967 41.1 Symbol: `celf_mp_cs_get_resp_msg`

2968 41.1.1 Syntax

```
2969 CelfMpStatus celf_mp_cs_get_resp_msg (  
2970     CelfMpRegNum    reg_no,  
2971     CelfMpCsSrvData * data);
```

2972

2973 41.1.2 Argument

2974 **Name:** `reg_no`

2975 **Type:** `CelfMpRegNum`

2976 **I/O:** `I`

2977 **Description:**

2978 Registration number: 1 to 10

2979

2980 **Name:** `data`

2981 **Type:** `CelfMpCsSrvData`

2982 **I/O:** `I`

2983 **Description:**

2984 Pointer to the additional response message setting data area

2985

2986 41.1.3 Return Value

2987 **Type:** `CelfMpStatus`

2988 **Description:**

2989 `celf_mp_cs_get_resp_msg()` shall return one of the values defined:

2990 `CELF_MP_STATUS_REG_NUM_ERR:` `reg_no` is out of range

2991 `CELF_MP_STATUS_OK:` successful completion

2992 `CELF_MP_STATUS_ERR:` Other unsuccessful completion.

2993

2994 41.1.4 Include File

2995 `/usr/include/celf/mp_cs.h`

2996

2997 41.1.5 Functional Description

2998 This function obtains the additional response message information, specified by “`reg_no`”,
2999 from non-volatile memory.

3000

3001 See "Register response message settings".

DRAFT

3002 **42.Delete Response Message Settings**

3003 **42.1 Symbol: celf_mp_cs_del_resp_msg**

3004 **42.1.1 Syntax**

3005 CelfMpStatus celf_mp_cs_del_resp_msg (
3006 CelfMpRegNum reg_no);
3007

3008 **42.1.2 Argument**

3009 **Name:** reg_no

3010 **Type:** CelfMpRegNum

3011 **I/O:** I

3012 **Description:**

3013 Registration number: 1 to 10
3014

3015 **42.1.3 Return Value**

3016 **Type:** CelfMpStatus

3017 **Description:**

3018 celf_mp_cs_del_resp_msg() shall return one of the values defined:

3019 CELF_MP_STATUS_REG_NUM_ERR: reg_no is out of range

3020 CELF_MP_STATUS_OK: successful completion

3021 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3022

3023 **42.1.4 Include File**

3024 /usr/include/celf/mp_cs.h
3025

3026 **42.1.5 Functional Description**

3027 This function deletes the additional response message information, specified by "reg_no",
3028 from non-volatile memory.

3029 **43.Remove All Response Message Settings**

3030 **43.1 Symbol: celf_mp_cs_remove_all_resp_msg**

3031 **43.1.1 Syntax**

3032 CelfMpStatus celf_mp_cs_remove_all_resp_msg (
3033 void);
3034

3035 **43.1.2 Argument**

3036 None.
3037

3038 **43.1.3 Return Value**

3039 **Type:** CelfMpStatus

3040 **Description:**

3041 celf_mp_cs_remove_all_resp_msg() **shall** return one of the values defined:

3042 CELF_MP_STATUS_OK: successful completion

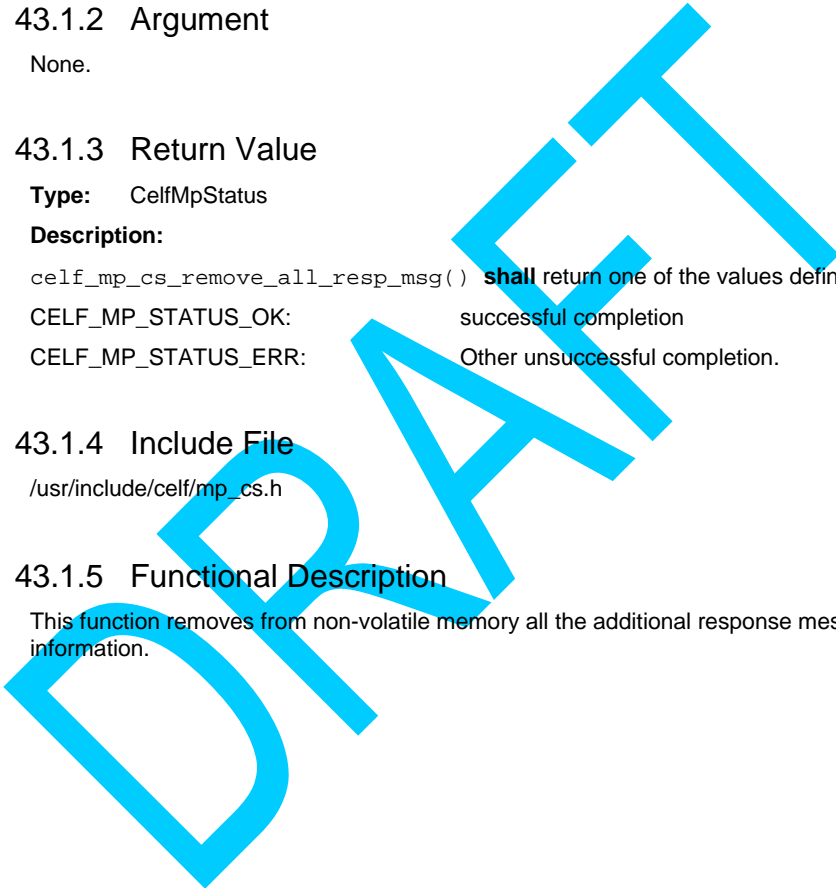
3043 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3044

3045 **43.1.4 Include File**

3046 /usr/include/celf/mp_cs.h
3047

3048 **43.1.5 Functional Description**

3049 This function removes from non-volatile memory all the additional response message
3050 information.
3051



3052 **44.Set Reconnection Tone**

3053 **44.1 Symbol: celf_mp_cs_set_reconnection_tone**

3054 **44.1.1 Syntax**

3055 CelfMpStatus celf_mp_cs_set_reconnection_tone (
3056 CelfMpCsReconnectionTone reconn);
3057

3058 **44.1.2 Argument**

3059 **Name:** reconn

3060 **Type:** CelfMpCsReconnectionTone

3061 **I/O:** I

3062 **Description:**

3063 Reconnection tone to be set

3064 CELF_MP_CS_RECONN_ON_T_OFF: Tone OFF

3065 CELF_MP_CS_RECONN_ON_T_LOW: Tone ON low tone

3066 CELF_MP_CS_RECONN_ON_T_HI: Tone ON high tone
3067

3068 **44.1.3 Return Value**

3069 **Type:** CelfMpStatus

3070 **Description:**

3071 `celf_mp_cs_set_reconnection_tone()` **shall** return one of the values defined:

3072 CELF_MP_STATUS_OK: successful completion

3073 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3074

3075 **44.1.4 Include File**

3076 `/usr/include/celf/mp_cs.h`
3077

3078 **44.1.5 Functional Description**

3079 This function sets the reconnection tone information to the non-volatile memory.
3080

3081 **45. Get Reconnection Tone**

3082 **45.1 Symbol: celf_mp_cs_get_reconnection_tone**

3083 **45.1.1 Syntax**

3084 CelfMpStatus celf_mp_cs_get_reconnection_tone (
3085 CelfMpCsReconnectionTone result);
3086

3087 **45.1.2 Argument**

3088 **Name:** result

3089 **Type:** CelfMpCsReconnectionTone

3090 **I/O:** O

3091 **Description:**

3092 returns one of the values defined:

3093 CELF_MP_CS_RECONN_ON_T_OFF: Tone OFF

3094 CELF_MP_CS_RECONN_ON_T_LOW: Tone ON low tone

3095 CELF_MP_CS_RECONN_ON_T_HI: Tone ON high tone
3096

3097 **45.1.3 Return Value**

3098 **Type:** CelfMpStatus

3099 **Description:**

3100 `celf_mp_cs_get_reconnection_tone()` **shall** return one of the values defined:

3101 CELF_MP_STATUS_OK: successful completion

3102 CELF_MP_STATUS_ERR: Other unsuccessful completion
3103

3104 **45.1.4 Include File**

3105 `/usr/include/celf/mp_cs.h`
3106

3107 **45.1.5 Functional Description**

3108 This function gets the reconnection tone information to the non-volatile memory.

3109 **46. Get Noise Cancel**

3110 **46.1 Symbol: `celf_mp_cs_get_noise_cancel`**

3111 **46.1.1 Syntax**

```
3112 CelfMpStatus celf_mp_cs_get_noise_cancel (  
3113     CelfMpCsNoiseCancel result);  
3114
```

3115 **46.1.2 Argument**

3116 **Name:** result

3117 **Type:** CelfMpCsNoiseCancel

3118 **I/O:** O

3119 **Description:**

3120 returns one of the values defined:

3121 CELF_MP_CS_ON: Noise canceller ON

3122 CELF_MP_CS_OFF: Noise canceller OFF

3123

3124 **46.1.3 Return Value**

3125 **Type:** CelfMpStatus

3126 **Description:**

3127 `celf_mp_cs_get_noise_cancel()` shall return one of the values defined:

3128 CELF_MP_STATUS_OK: successful completion

3129 CELF_MP_STATUS_ERR: Other unsuccessful completion

3130

3131 **46.1.4 Include File**

3132 `/usr/include/celf/mp_cs.h`

3133

3134 **46.1.5 Functional Description**

3135 This function gets the noise canceller status.

3136

3137 **47.Set Noise Cancel**

3138 **47.1 Symbol: celf_mp_cs_set_noise_cancel**

3139 47.1.1 Syntax

```
3140 CelfMpStatus celf_mp_cs_set_noise_cancel (  
3141 CelfMpCsNoiseCancel mode);  
3142
```

3143 47.1.2 Argument

3144 **Name:** mode

3145 **Type:** CelfMpCsNoiseCancel

3146 **I/O:** I

3147 **Description:**

3148 Reconnection tone to be set

3149 CELF_MP_CS_ON: Noise canceller ON

3150 CELF_MP_CS_OFF: Noise canceller OFF

3151

3152 47.1.3 Return Value

3153 **Type:** CelfMpStatus

3154 **Description:**

3155 `celf_mp_cs_set_noise_cancel()` shall return one of the values defined:

3156 CELF_MP_STATUS_OK: successful completion

3157 CELF_MP_STATUS_ERR: Other unsuccessful completion.

3158

3159 47.1.4 Include File

3160 `/usr/include/celf/mp_cs.h`

3161

3162 47.1.5 Functional Description

3163 This function sets the noise canceller off or on.

3164

3165 **48. Get Call Quality Alarm**

3166 **48.1 Symbol: celf_mp_cs_get_call_quality_alarm**

3167 **48.1.1 Syntax**

3168 CelfMpStatus celf_mp_cs_get_call_quality_alarm (
3169 CelfMpCsQualAlarm result);
3170

3171 **48.1.2 Argument**

3172 **Name:** result

3173 **Type:** CelfMpCsQualAlarm

3174 **I/O:** O

3175 **Description:**

3176 returns one of the values defined:

3177 CELF_MP_CS_QUALITY_ALM_OFF: Quality alarm OFF

3178 CELF_MP_CS_QUALITY_ALM_LOW: Quality alarm ON low tone

3179 CELF_MP_CS_QUALITY_ALM_HI: Quality alarm ON high tone
3180

3181 **48.1.3 Return Value**

3182 **Type:** CelfMpStatus

3183 **Description:**

3184 celf_mp_cs_get_call_quality_alarm() **shall** return one of the values defined:

3185 CELF_MP_STATUS_OK: successful completion

3186 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3187

3188 **48.1.4 Include File**

3189 /usr/include/celf/mp_cs.h
3190

3191 **48.1.5 Functional Description**

3192 This function gets the status of the call quality alarm sound.

3193 **49.Set Call Quality Alarm**

3194 **49.1 Symbol: celf_mp_cs_set_call_quality_alarm**

3195 49.1.1 Syntax

3196 CelfMpStatus celf_mp_cs_set_call_quality_alarm (
3197 CelfMpCsQualAlarm mode);
3198

3199 49.1.2 Argument

3200 **Name:** mode

3201 **Type:** CelfMpCsQualAlarm

3202 **I/O:** I

3203 **Description:**

3204 CELF_MP_CS_QUALITY_ALM_OFF: Quality alarm OFF

3205 CELF_MP_CS_QUALITY_ALM_LOW: Quality alarm ON low tone

3206 CELF_MP_CS_QUALITY_ALM_HI: Quality alarm ON high tone
3207

3208 49.1.3 Return Value

3209 **Type:** CelfMpStatus

3210 **Description:**

3211 celf_mp_cs_set_call_quality_alarm() **shall** return one of the values defined:

3212 CELF_MP_STATUS_OK: successful completion

3213 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3214

3215 49.1.4 Include File

3216 /usr/include/celf/mp_cs.h
3217

3218 49.1.5 Functional Description

3219 This function sets the call quality alarm sound.

3220 50. Get Noise Cancel Permit

3221 50.1 Symbol: `celf_mp_cs_get_noise_cancel_permit`

3222 50.1.1 Syntax

```
3223 CelfMpStatus celf_mp_cs_get_noise_cancel_permit (  
3224 CelfMpCsNoiseCancel result);
```

3226 50.1.2 Argument

3227 **Name:** result

3228 **Type:** CelfMpCsNoiseCancel

3229 **I/O:** O

3230 **Description:**

3231 returns one of the values defined:

3232 CELF_MP_CS_ON: Noise canceller permission

3233 CELF_MP_CS_OFF: Noise canceller non-permission

3235 50.1.3 Return Value

3236 **Type:** CelfMpStatus

3237 **Description:**

3238 `celf_mp_cs_get_noise_cancel_permit()` **shall** return one of the values defined:

3239 CELF_MP_STATUS_OK: successful completion

3240 CELF_MP_STATUS_ERR: Other unsuccessful completion.

3242 50.1.4 Include File

3243 `/usr/include/celf/mp_cs.h`

3245 50.1.5 Functional Description

3246 This function obtains whether noise canceller is permitted or not.

3247 51. Get High Priority communication mode

3248 51.1 Symbol: `celf_mp_cs_get_hi_prio_com`

3249 51.1.1 Syntax

```
3250 CelfMpStatus celf_mp_cs_get_hi_prio_com(  
3251 CelfMpCsHiPrioCom mode);  
3252
```

3253 51.1.2 Argument

3254 **Name:** mode

3255 **Type:** CelfMpCsHiPrioCom

3256 **I/O:** O

3257 **Description:**

3258 Connection priority currently set:

3259 CELF_MP_CS_COMPRI_NONE: No setting

3260 CELF_MP_CS_COMPRI_VOICE: Voice

3261 CELF_MP_CS_COMPRI_PACKET: Packet
3262

3263 51.1.3 Return Value

3264 **Type:** CelfMpStatus

3265 **Description:**

3266 `celf_mp_cs_get_hi_prio_com()` shall return one of the values defined:

3267 CELF_MP_STATUS_OK: successful completion

3268 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3269

3270 51.1.4 Include File

3271 `/usr/include/celf/mp_cs.h`
3272

3273 51.1.5 Functional Description

3274 This function returns the high priority communication mode setting in either to voice
3275 communication or packet communication. The priority gives the mandatory order of the
3276 processing.
3277

3278 52.Set High Priority communication mode

3279 52.1 Symbol: `celf_mp_cs_set_hi_prio_com`

3280 52.1.1 Syntax

```
3281 CelfMpStatus celf_mp_cs_set_hi_prio_com(  
3282 CelfMpCsHiPrioCom mode);  
3283
```

3284 52.1.2 Argument

3285 **Name:** mode

3286 **Type:** CelfMpCsHiPrioCom

3287 **I/O:** I

3288 **Description:**

3289 Connection priority to be chosen.

3290 CELF_MP_CS_COMPRI_NONE: No setting

3291 CELF_MP_CS_COMPRI_VOICE: Voice

3292 CELF_MP_CS_COMPRI_PACKET: Packet

3293

3294 52.1.3 Return Value

3295 **Type:** CelfMpStatus

3296 **Description:**

3297 `celf_mp_cs_set_hi_prio_com()` shall return one of the values defined:

3298 CELF_MP_STATUS_OK: successful completion

3299 CELF_MP_STATUS_ERR: Other unsuccessful completion.

3300

3301 52.1.4 Include File

3302 `/usr/include/celf/mp_cs.h`

3303

3304 52.1.5 Functional Description

3305 This function sets the high priority communication mode either to voice communication or
3306 packet communication. The priority gives the mandatory order of the processing.

3307 53. Get Phone Answering Sound Activation

3308 53.1 Symbol: `celf_mp_cs_get_vm_sound_status`

3309 53.1.1 Syntax

```
3310 CelfMpStatus celf_mp_cs_get_vm_sound_status (  
3311 CelfMpCsVmSound result);  
3312
```

3313 53.1.2 Argument

3314 **Name:** result

3315 **Type:** CelfMpCsVmSound

3316 **I/O:** O

3317 **Description:**

3318 returns one of the values defined:

3319 CELF_MP_CS_ON: Message sound ON

3320 CELF_MP_CS_OFF: Message sound OFF
3321

3322 53.1.3 Return Value

3323 **Type:** CelfMpStatus

3324 **Description:**

3325 `celf_mp_cs_get_vm_sound_status()` shall return one of the values defined:

3326 CELF_MP_STATUS_OK: successful completion

3327 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3328

3329 53.1.4 Include File

3330 `/usr/include/celf/mp_cs.h`
3331

3332 53.1.5 Functional Description

3333 This function gets the setting status.

3334 If the CelfMpCsVmSound is ON, the phone sounds, when the number of voice mail system
3335 increases.

3336 **54.Set Phone Answering Sound Activation**

3337 **54.1 Symbol: celf_mp_cs_set_vm_sound_status**

3338 **54.1.1 Syntax**

3339 CelfMpStatus celf_mp_cs_set_vm_sound_status (
3340 CelfMpCsVmSound mode);
3341

3342 **54.1.2 Argument**

3343 **Name:** mode

3344 **Type:** CelfMpCsVmSound

3345 **I/O:** I

3346 **Description:**

3347 CELF_MP_CS_ON: Message sound ON

3348 CELF_MP_CS_OFF: Message sound OFF
3349

3350 **54.1.3 Return Value**

3351 **Type:** CelfMpStatus

3352 **Description:**

3353 celf_mp_cs_set_vm_sound_status() **shall** return one of the values defined:

3354 CELF_MP_STATUS_OK: successful completion

3355 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3356

3357 **54.1.4 Include File**

3358 /usr/include/celf/mp_cs.h
3359

3360 **54.1.5 Functional Description**

3361 This function sets the phone sounds status whether the phone sounds or not.

3362 **55. Get Automatic Receive Status**

3363 **55.1 Symbol: celf_mp_cs_get_auto_rcv_status**

3364 **55.1.1 Syntax**

3365 CelfMpStatus celf_mp_cs_get_auto_rcv_status (
3366 CelfMpCsAutoRcv mode);
3367

3368 **55.1.2 Argument**

3369 **Name:** mode

3370 **Type:** CelfMpCsAutoRcv

3371 **I/O:** O

3372 **Description:**

3373 returns one of the values defined:

3374 CELF_MP_CS_ON: Automatic incoming call ON

3375 CELF_MP_CS_OFF: Automatic incoming call OFF
3376

3377 **55.1.3 Return Value**

3378 **Type:** CelfMpStatus

3379 **Description:**

3380 celf_mp_cs_get_auto_rcv_status() **shall** return one of the values defined:

3381 CELF_MP_STATUS_OK: successful completion

3382 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3383

3384 **55.1.4 Include File**

3385 /usr/include/celf/mp_cs.h
3386

3387 **55.1.5 Functional Description**

3388 This function obtains the status of automatic incoming call.

3389 **56.Set Automatic Receive Status**

3390 **56.1 Symbol: celf_mp_cs_set_auto_rcv_status**

3391 **56.1.1 Syntax**

3392 CelfMpStatus celf_mp_cs_set_auto_rcv_status (
3393 CelfMpCsAutoRcv mode);
3394

3395 **56.1.2 Argument**

3396 **Name:** mode

3397 **Type:** CelfMpCsAutoRcv

3398 **I/O:** I

3399 **Description:**

3400 CELF_MP_CS_ON: Automatic incoming call ON

3401 CELF_MP_CS_OFF: Automatic incoming call OFF
3402

3403 **56.1.3 Return Value**

3404 **Type:** CelfMpStatus

3405 **Description:**

3406 celf_mp_cs_set_auto_rcv_status() **shall** return one of the values defined:

3407 CELF_MP_STATUS_OK: successful completion

3408 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3409

3410 **56.1.4 Include File**

3411 /usr/include/celf/mp_cs.h
3412

3413 **56.1.5 Functional Description**

3414 This function sets the automatic incoming call status.

3415 **57. Get Automatic Timer**

3416 **57.1 Symbol: celf_mp_cs_get_auto_timer**

3417 **57.1.1 Syntax**

```
3418 CelfMpStatus celf_mp_cs_get_auto_timer(  
3419 CelfMpCsTimer result);
```

3420

3421 **57.1.2 Argument**

3422 **Name:** result

3423 **Type:** CelfMpCsTimer

3424 **I/O:** O

3425 **Description:**

3426 returns one of the values defined:

3427 1 to 120 seconds

3428

3429 **57.1.3 Return Value**

3430 **Type:** CelfMpStatus

3431 **Description:**

3432 `celf_mp_cs_get_auto_timer()` **shall** return one of the values defined:

3433 CELF_MP_STATUS_OK: successful completion

3434 CELF_MP_STATUS_ERR: Other unsuccessful completion.

3435

3436 **57.1.4 Include File**

3437 `/usr/include/celf/mp_cs.h`

3438

3439 **57.1.5 Functional Description**

3440 This function obtains the timer value of the automatic incoming call.

3441 The timer value is the duration of sounding of the ring alert.

3442 **58.Set Automatic Timer**

3443 **58.1 Symbol: celf_mp_cs_set_auto_timer**

3444 **58.1.1 Syntax**

3445 CelfMpStatus celf_mp_cs_set_auto_timer (
3446 CelfMpCsTimer time);

3448 **58.1.2 Argument**

3449 **Name:** time

3450 **Type:** CelfMpCsTimer

3451 **I/O:** I

3452 **Description:**

3453 1 to 120 seconds

3455 **58.1.3 Return Value**

3456 **Type:** CelfMpStatus

3457 **Description:**

3458 celf_mp_cs_set_auto_timer() **shall return one of the values defined:**

3459 CELF_MP_STATUS_OK: successful completion

3460 CELF_MP_STATUS_ERR: Other unsuccessful completion.

3462 **58.1.4 Include File**

3463 /usr/include/celf/mp_cs.h

3465 **58.1.5 Functional Description**

3466 This function sets the timer value of the automatic incoming call.

3467

3468

3469

3470

3471 **59. Get Reset Date**

3472 **59.1 Symbol: celf_mp_cs_get_reset_date**

3473 **59.1.1 Syntax**

3474 CelfMpStatus celf_mp_cs_get_reset_date(
3475 CelfMpCsDate * reset_date);
3476

3477 **59.1.2 Argument**

3478 **Name:** reset_date
3479 **Type:** CelfMpCsDate
3480 **I/O:** O
3481 **Description:**
3482 Accumulated date record
3483 See section 1. for details.
3484
3485

3486 **59.1.3 Return Value**

3487 **Type:** CelfMpStatus
3488 **Description:**
3489 `celf_mp_cs_get_reset_date()` shall return one of the values defined:
3490 CELF_MP_STATUS_OK: successful completion
3491 CELF_MP_STATUS_ERR: Other unsuccessful completion
3492 CELF_MP_STATUS_COM_TYPE_ERR: communication type not supported.
3493

3494 **59.1.4 Include File**

3495 `/usr/include/celf/mp_cs.h`
3496

3497 **59.1.5 Functional Description**

3498 This function obtains the date and time when the accumulated call duration was reset.
3499 The value is obtained from non-volatile memory.

3500 **60.Set Reset Date**

3501 **60.1 Symbol: celf_mp_cs_set_reset_date**

3502 **60.1.1 Syntax**

3503 CelfMpStatus celf_mp_cs_set_reset_date(
3504 void);
3505

3506 **60.1.2 Argument**

3507 None.
3508

3509 **60.1.3 Return Value**

3510 **Type:** CelfMpStatus

3511 **Description:**

3512 `celf_mp_cs_set_reset_date()` shall return one of the values defined:

3513 CELF_MP_STATUS_OK: successful completion

3514 CELF_MP_STATUS_ERR: Other unsuccessful completion

3515 CELF_MP_STATUS_COM_TYPE_ERR: communication type not supported.
3516

3517 **60.1.4 Include File**

3518 `/usr/include/celf/mp_cs.h`
3519

3520 **60.1.5 Functional Description**

3521 This function sets the current date and time as the reset date and time of the accumulated
3522 call duration.

3523 The value set to non-volatile memory.

3524 **61. Get Call Silent Time**

3525 **61.1 Symbol: celf_mp_cs_get_call_silent_time**

3526 **61.1.1 Syntax**

3527 CelfMpStatus celf_mp_cs_get_call_silent_time(
3528 CelfMpTime time);
3529

3530 **61.1.2 Argument**

3531 **Name:** time

3532 **Type:** CelfMpTime

3533 **I/O:** O

3534 **Description:**

3535 returns one of the values defined:

3536 0 to 99 seconds
3537

3538 **61.1.3 Return Value**

3539 **Type:** CelfMpStatus

3540 **Description:**

3541 celf_mp_cs_get_call_silent_time() **shall** return one of the values defined:

3542 CELF_MP_STATUS_OK: successful completion

3543 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3544

3545 **61.1.4 Include File**

3546 /usr/include/celf/mp_cs.h
3547

3548 **61.1.5 Functional Description**

3549 This function gets the duration between the arrival of incoming call and the start of
3550 sounding of the ring alert. This duration is called the silent time.

3551 This function is effective that the number of this incoming call is unregistered with the
3552 phone book.

3553 **62.Set Call Silent Time**

3554 **62.1 Symbol: celf_mp_cs_set_call_silent_time**

3555 **62.1.1 Syntax**

3556 CelfMpStatus celf_mp_cs_set_call_silent_time(
3557 CelfMpCsTimer time);
3558

3559 **62.1.2 Argument**

3560 **Name:** time

3561 **Type:** CelfMpCsTimer

3562 **I/O:** I

3563 **Description:**

3564 0 to 99 seconds
3565

3566 **62.1.3 Return Value**

3567 **Type:** CelfMpStatus

3568 **Description:**

3569 celf_mp_cs_set_call_silent_time() **shall** return one of the values defined:

3570 CELF_MP_STATUS_OK: successful completion

3571 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3572

3573 **62.1.4 Include File**

3574 /usr/include/celf/mp_cs.h
3575

3576 **62.1.5 Functional Description**

3577 This function sets the silent time.

3578 Refer to get calling operation start time.

3579 **63. Get Call Recorded**

3580 **63.1 Symbol: celf_mp_cs_get_call_recorded**

3581 **63.1.1 Syntax**

```
3582 CelfMpStatus celf_mp_cs_get_call_recorded(  
3583 CelfMpSetting mode);  
3584
```

3585 **63.1.2 Argument**

3586 **Name:** mode

3587 **Type:** CelfMpSetting

3588 **I/O:** O

3589 **Description:**

3590 returns one of the values defined:

3591 CELF_MP_CS_ON: Setting ON

3592 CELF_MP_CS_OFF: Setting OFF

3593

3594 **63.1.3 Return Value**

3595 **Type:** CelfMpStatus

3596 **Description:**

3597 celf_mp_cs_get_call_recorded() shall return one of the values defined:

3598 CELF_MP_STATUS_OK: successful completion

3599 CELF_MP_STATUS_ERR: Other unsuccessful completion.

3600

3601 **63.1.4 Include File**

3602 /usr/include/celf/mp_cs.h

3603

3604 **63.1.5 Functional Description**

3605 This function gets the setting condition of whether the silent call is recorded in the absent
3606 incoming call log, or not.

3607 The absent incoming call log is the log that records no-responded incoming call.

3608 The silent call is the incoming call, which disconnects within the silent time.

3609 Refer to "Get calling operation start time".

3610

3611

3612

3613 **64.Set Call Recorded**

3614 **64.1 Symbol: celf_mp_cs_set_call_recorded**

3615 **64.1.1 Syntax**

3616 CelfMpStatus celf_mp_cs_set_call_recorded(
3617 CelfMpCsSetting mode);
3618

3619 **64.1.2 Argument**

3620 **Type:** CelfMpCsSetting

3621 **I/O:** I

3622 **Description:**

3623 CELF_MP_CS_ON: Setting ON

3624 CELF_MP_CS_OFF: Setting OFF
3625

3626 **64.1.3 Return Value**

3627 **Type:** CelfMpStatus

3628 **Description:**

3629 celf_mp_cs_set_call_recorded() **shall return one of the values defined:**

3630 CELF_MP_STATUS_OK: successful completion

3631 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3632

3633 **64.1.4 Include File**

3634 /usr/include/celf/mp_cs.h
3635

3636 **64.1.5 Functional Description**

3637 This function sets the setting condition of whether the silent call is recorded in the absent
3638 incoming call log or not.

3639 Refer to "Get recording condition to absent incoming call log".

3640 **65.Set Radio**

3641 **65.1 Symbol: celf_mp_cs_set_radio**

3642 **65.1.1 Syntax**

3643 CelfMpStatus celf_mp_cs_set_radio(
3644 CelfMpCsSetting mode);
3645

3646 **65.1.2 Argument**

3647 **Type:** CelfMpCsSetting

3648 **I/O:** I

3649 **Description:**

3650 CELF_MP_CS_ON: Setting ON

3651 CELF_MP_CS_OFF: Setting OFF
3652

3653 **65.1.3 Return Value**

3654 **Type:** CelfMpStatus

3655 **Description:**

3656 celf_mp_cs_set_radio() **shall** return one of the values defined:

3657 CELF_MP_STATUS_OK: successful completion

3658 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3659

3660 **65.1.4 Include File**

3661 /usr/include/celf/mp_cs.h
3662

3663 **65.1.5 Functional Description**

3664 This function sets the setting for deactivating or reactivating the Radio RF.

3665 **66. Get Radio Status**

3666 **66.1 Symbol: celf_mp_cs_get_radio**

3667 **66.1.1 Syntax**

3668 CelfMpStatus celf_mp_cs_get_radio(
3669 CelfMpCsSetting mode);
3670

3671 **66.1.2 Argument**

3672 **Type:** CelfMpCsSetting

3673 **I/O:** O

3674 **Description:**

3675 CELF_MP_CS_ON: Setting ON

3676 CELF_MP_CS_OFF: Setting OFF
3677

3678 **66.1.3 Return Value**

3679 **Type:** CelfMpStatus

3680 **Description:**

3681 celf_mp_cs_get_radio() **shall** return one of the values defined:

3682 CELF_MP_STATUS_OK: successful completion

3683 CELF_MP_STATUS_ERR: Other unsuccessful completion.
3684

3685 **66.1.4 Include File**

3686 /usr/include/celf/mp_cs.h
3687

3688 **66.1.5 Functional Description**

3689 This function gets the status the Radio RF.
3690
3691