



1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26

## Linux based 3G Specification

## Multimedia Mobile Phone API

### Preface

Document: CELF\_MPP\_Preface ~~FR2\_20060606~~

- Deleted: FR1
- Deleted: 20060301
- Deleted: 20060301

**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](mailto:MppApiComments@tree.celinuxforum.org)

## Revision History

Revision	Comment	Reviewer	Editor	Date
2.2	Prepared for Tamagawa meeting		NEC, Panasonic	2005.9.28
2.2.1	Prepared for San Francisco meeting, new format		Scott Preece	2005.10.31
2.2.3	Integrated issue resolutions from Reference Architecture and added Common Types and Programming Model sections.		Scott Preece	2005.12.20
2.2.4	Updated to clarify that Application ID and Client ID are the same thing.		Scott Preece	2006.2.1
FR1	First version for Formal Review		Scott Preece	2006.3.1
<u>FR2</u>	<u>Revised to resolve first group of Formal Review comments</u>		<u>Scott Preece</u>	<u>2006.6.6</u>

28	<b>0. Introduction.....</b>	<b><u>4</u></b>	Deleted: 4
29	0.1.1 Circuit Switched Communication Service.....	<u>4</u>	Deleted: 4
30	0.1.2 Packet Switched Communication Service .....	<u>4</u>	Deleted: 4
31	0.1.3 Short Message Service.....	<u>4</u>	Deleted: 4
32	0.1.4 Equipment Service.....	<u>4</u>	Deleted: 4
33	<b>0.2 Structure of API Documents.....</b>	<b><u>4</u></b>	Deleted: 4
34	0.2.1 Introduction Section .....	<u>4</u>	Deleted: 4
35	0.2.2 Primitives Section.....	<u>4</u>	Deleted: 4
36	0.2.3 Functions Section .....	<u>4</u>	Deleted: 4
37	<b>0.3 Terminology and abbreviations.....</b>	<b><u>5</u></b>	Deleted: 4
38	<b>0.4 References .....</b>	<b><u>9</u></b>	Deleted: 5
39	0.4.1 Normative.....	<u>9</u>	Deleted: 9
40	0.4.2 Informative .....	<u>9</u>	Deleted: 9
41	<b>1. Programming Model.....</b>	<b><u>10</u></b>	Deleted: 9
42	<b>1.1 Events and Notifications.....</b>	<b><u>10</u></b>	Deleted: 10
43	1.1.1 Spontaneous Events .....	<u>10</u>	Deleted: 10
44	1.1.2 Result Events .....	<u>10</u>	Deleted: 10
45	1.1.3 Application IDs.....	<u>10</u>	Deleted: 10
46	1.1.4 Callback Notification Functions .....	<u>10</u>	Deleted: 10
47	1.1.5 Registering.....	<u>11</u>	Deleted: 10
48	<b>1.2 Synchronous Service Interfaces.....</b>	<b><u>11</u></b>	Deleted: 10
49	<b>1.3 Asynchronous Service Interfaces .....</b>	<b><u>11</u></b>	Deleted: 10
50	<b>2. Common Primitives.....</b>	<b><u>12</u></b>	Deleted: 11
51	<b>2.1 Constants.....</b>	<b><u>12</u></b>	Deleted: 12
52	<b>2.2 Enums .....</b>	<b><u>12</u></b>	Deleted: 12
53	2.2.1 CelfMpStatus.....	<u>12</u>	Deleted: 12
54	<b>2.3 Data Types and Structures.....</b>	<b><u>12</u></b>	Deleted: 12
55	2.3.1 CelfMpCallRef .....	<u>12</u>	Deleted: 12
56	2.3.2 CelfMpEvent .....	<u>12</u>	Deleted: 12
57	2.3.3 CelfMpCallback.....	<u>13</u>	Deleted: 12
58	2.3.4 CelfMpAppId.....	<u>13</u>	Deleted: 12
59			Deleted: 13
			Deleted: 13

# 0. Introduction

This Preface to the CELF Mobile Phone API describes the overall structure of the API specification of the Telephony Service for 3G multimedia mobile telephone based on Linux. It also provides an introduction to some common concepts and terminology used in the Specifications and defines some common datatypes.

This document is the work of the CE Linux Forum’s Mobile Phone Profile Working Group [MPPWG].

The major sections of the API in the current release are described below. This Preface and the individual Service chapters are considered normative; a conforming implementation is required to satisfy statements written in “SHALL” form. However, conformance may be determined on a service-by-service basis.

## 0.1.1 Circuit Switched Communication Service

The Circuit Switched Communication Service (CS Service) API [CS] provides access to functionality for call control, call state management, tone control, and log processing. This chapter includes the Voice communication service, the Video communication service, and the Unrestricted Digital data Communication service.

## 0.1.2 Packet Switched Communication Service

The Packet Switched Communication Service (PS Service) API [PS] provides access to functionality for packet call control and for sending and receiving data packets. This chapter includes the PPP dial-up communication service and the IP connection data transfer service.

## 0.1.3 Reference Architecture

The Reference Architecture [RefArch] is an illustrative, non-normative description of a commonly understood way of implementing mobile handsets using a Linux-based application environment. A conforming implementation is not required to conform to the reference architecture.

# 0.2 Structure of API Documents

Each specification chapter defines the API for a major sub-area of functionality. The content of each chapter is divided into:

1. Introduction – An overview of the service, placing it in context.
2. Primitives – Definitions of the data types, constants, and enumerations used in the API definitions.
3. Functions – Definitions of the individual functional interfaces provided by the service.

## 0.2.1 Introduction Section

An introduction to the functionality available through the API of the service described by the chapter.

## 0.2.2 Primitives Section

This section is subdivided into sub-sections for Data Types and Structures and for Constants. In each case, the primitive is named, its use is described, and its formal definition (as would appear in a header file) is given.

## 0.2.3 Functions Section

Each function appears as a separate section. The information given for each function includes:

Symbol	The formal (programming) name of the function.
Syntax	Syntax used in programming in C language
Argument	Arguments of API function in C language
Return value	Return value of API function in C language

**Comment [sep1]:** 2 Since the Reference Architecture document identifies several other functional components of the Telephony Service (e.g. PIM, Data Exchange, etc.) this section should state the enumerated services (i.e. CS, PS, SMS and ES) are not exhaustive. DONE

**Comment [sep2]:** 53 Define Conformance requirements DONE - PARTIALLY

**Comment [sep3]:** 3 State which parts of this document are Normative (and non-Normative). DONE

**Deleted:** sections of the

**Deleted:** Telephony

**Comment [sep4]:** 4 For each of these sections, add a Reference to the related specification and add those references to section 0.4. For the ES and SMS section, state those documents are a Work In Progress (and not publicly available). TBD - Added slots and references, but need to populate slots; deleted ES and SMS.

**Formatted:** Heading 3

**Deleted:** <#>Short Message Service¶  
The Short Message Service (SMS Service) API provides access to functionality for sending and receiving messages using the SMS protocol.¶  
<#>Equipment Service¶  
The Equipment Service API provides access to functionality for setting and reading various status information and operating modes of the handset (Earphone mode, Manner mode, Dial call restriction, Battery level, etc.).¶

**Formatted:** Bullets and Numbering

**Deleted:** I

- 99 Include file File name to be included in Programming
- 100 Functional description Definition and detail explanation of API function

### 101 0.3 Terminology and abbreviations

102 The following words, phrases, and acronyms have specific meanings within the context of the API.

word	explanation
32K AV communication	Communication mode with AV at the speed of 32Kbps
32K data communication	Data communication mode at a stable communication speed of 32Kbps. Unlimited digital 32K communication.
64K AV communication	Communication mode with AV at the speed of 64Kbps
64K data communication	Data communication mode at a stable communication speed of 64Kbps. Unlimited digital 64K communication.
accumulated reset	Resetting of the accumulated duration data. The handset stores data on the total duration of all calls .
API	Application Program Interface
<u>APN</u>	<u>Access Point Name</u>
App or Application	Application program; a program run in user space.
ASF	Advanced Streaming Format
automatic incoming call	Operating mode in which the handset automatically accepts incoming calls, without the user accepting each call by a manual operation..
automatic transmission	Placing a call by keying in all the digits and then initiating the connection. Same as "on-hook originating".
call duration	The duration of a voice call.
call quality alarm	The indication that radio reception from the network has deteriorated and the call is likely to be dropped.
call reference	An identifier for a particular call. This identifier is assigned by the network or mobile phone, and used in the call-management APIs to operate on a particular.
<u>C Plane</u>	<u>Control Plane – the subset of components in a network architecture that are responsible for controlling connections.</u>
CS	Circuit Switched operation; a mode of communication in which a dedicated channel is maintained between the handset and the remote party and the call content is routed over that identified channel.

**Comment [sep5]:** 5 Add: C-Plane (used in Ref Arch), MPP Server (used in section 1.1.1), APN (used in PS spec), U-Plane (used in PS spec) DONE

**Comment [sep6]:** 52 Add H234 and H324M to the Preface document's terminology section and add their references to Section 0.3. in RA DONE

Classification: Mobile Phone API

DCF	Device Control Function. The module that provides the following functions: <ul style="list-style-type: none"> <li>• Mobile phone control via AT commands.</li> <li>• Monitoring S-IF message and notice status change event to service.</li> <li>• To notice MTF (block which exchange message between TAF-NW) when sets up receive denial.</li> </ul>
DTMF	Dual Tone Multi Frequency. The tones generated to correspond to key presses while a CS connection is open (off-hook originating). On digital connections, the tones may be represented by designated codes rather than encoded audio.
Earphone (external option)	Controls whether audio is routed to an attached earphone (headset) or to a built-in loudspeaker.
emergency originating restriction	A network condition in which call from handsets are not accepted because an emergency requires all of the available network capacity..
Engine	Application Engine; a software module providing “backend” processing to support a service interface.
external AV communication	Videophone communication using a USB connection cable, etc., to connect terminal and external equipment (such as a PC@personal computer) to the handset).
FLASH	Macromedia Flash Player; the engine that execute Flash programs.
<a href="#">H234 and H324M</a>	<a href="#">3G multimedia services – H324M is video telephony, H234 is encryption key management</a>
high priority communication mode	The display mode in which an alert or icon is displayed in case of <ul style="list-style-type: none"> <li>(a) an incoming packet switched communication when circuit switched communication is active, or</li> <li>(b) an incoming circuit switched communication when packet switched communication is active.</li> </ul>
hold tone	A tone or melody that sounds when a voice call or AV call is changed to hold status.
HTTP	Hyper Text Transfer Protocol
I/F	Interface
IMEI	IMEI (International Mobile Station Equipment Identity). A unique number allocated to each individual mobile station (handset).
internal AV communication	Videophone communication between terminals.

IR	Infra-Red
JAM	Java Application Manager
JVM	Java Virtual Machine
Kernel	Linux Kernel
keypad dial lock	When this function is set, the handset does not allow voice or videophone calls by dialing phone numbers, extension number, or SIP. Dialing from previously stored "Phonebook" entries and from the "Dialed calls" or "Redial" entries remains possible.
LCD	Liquid Crystal Display
Low-voltage alarm	The alarm sounded to indicate that the battery is about to run out of power.
manner mode	Manner mode provides a quick and convenient way of muting the terminal's ring tones and keypad sound to avoid disturbing people around you.
manual transmission	Same as off-hook originating
MAW	Monitoring and Watching
<u>MPPWG</u>	<u>The CE Linux Forum's Mobile Phone Profile Working Group, which defined this specification and the related Service specifications.</u>
MSB	Mobile Software Bus
multiple calls	It is the combination of maximum three call. The conversation, hold and incoming call is at most one call.
noise canceller	A function that reduce ambient transmitted over a connection so that the other party can hear the voice more clearly.
normal originating restriction	When this mode is set, outgoing calls are permitted only to designated special numbers.
number notification	On option that determines whether the handset's telephone number is sent to the other party when a call is initiated.
OBEX	Object Exchange protocol
OCR	Optical Character Recognition
off-hook originating	Placing a call by keying the digits after pressing the start button; when five seconds have elapsed since the last input digit the call is initiated.
on-hook originating	Placing a call by pressing the start button after inputting all dial digits.
out-of-communication area	The mode of operation when the handset is unable to establish communication with the network because it is out of the service area or the signal is too weak or there is no network with which the handset is

	allowed to register.
phone-answering message	A message sent to the calling party when the handset can not respond to an incoming call.
phone-answering message service	A network-side service that provides for recording messages from callers when the handset is not in service..
PIM lock	A handset mode in which the user has indicated that no access is allowed to personal-information resources, such as "Phonebook", "Schedule", "Mail", "Messenger", and "Presence".
PIN	Personal Identification Number
PS	Packet Switched network
receive level	The receive level is the strength of the radio signal received from the network.
reconnection tone	The tone that sounds when the handset reconnects to the network after being out of service.
SCA	Stream Control API
SD	SD memory card
SDFS	SD File System
secret mode	A handset mode that controls whether personal information resource display or hide those entries that have been marked by the user as secret.
SMS	Short Message Service
special number	A number to connect with a service center in the network.
SS	A Supplementary network service accessible using the SS protocol, which encodes a service code as a four-part data string starting with '*', '#' or '*#*' and ending with '#'. The service code is either a standardized 3GPP code or a code defined by operator (USSD).
SSL	Secure Socket Layer
supplementary service	An optional service provided by the network and available to the handset through special signalling.
TAF	Terminal Adaptation Function. The module that connects handset functions to communication services.
<u>U Plane</u>	<u>User Plane – the subset of components within a network architecture that are responsible for the transfer of user data.</u>



UIM	Same as USIM (Universal Subscriber Identity Module).The removable hardware module that contains information identifying a network account plus various kinds of user-defined information (phone book entries, messages, service-specific information, applications, etc.).
USSD	Unstructured Supplementary Service Data a network- specific supplementary service code.
WDC	Watching Device Condition
within-communication area	The condition when the handset is in service area and able to communicate with the network.

103

## 104 0.4 References

### 105 0.4.1 Normative

106 [\[CS\]](#)

107 [\[Preface\] This document.](#)

108 [\[PS\]](#)

109 [\[RefArch\]](#)

110

### 111 0.4.2 Informative

112 [\[MPPWG\]](#)

Comment [sep7]: 6 Add: MPPWG (referenced in section 0.) DONE

Formatted: Reference Entry

Formatted: Normal

DRAFT

113

# 1. Programming Model

114

The MPP API defines both synchronous and asynchronous interfaces. Synchronous interfaces return a result directly to the calling program, whose execution is blocked until the function returns. Asynchronous interfaces return a result directly, but the result indicates only whether the request was successfully initiated. The actual result of an asynchronous service request is received as an event notification sometime after the request has been made. Asynchronous operations are used when the delay involved in processing a request is likely to be long for the client to block and be unable to do other work.

115

116

117

118

119

120

## 1.1 Events and Notifications

121

MPP API clients can register with service providers to receive notification when specified events occur. The API implementation delivers notifications by calling a function (the callback function) specified by the client at the time the client registered the request for notification. Registration is persistent; a client remains registered until it explicitly unregisters or exits.

122

123

124

125

126

127

128

The event-delivery model is widely used throughout the interface, both for delivering the results of asynchronous service requests ("result events") and for notifying clients of events that occur in the system ("spontaneous events"). For instance, a client can register to receive notification when an incoming call arrives from the network.

129

### 1.1.1 Spontaneous Events

130

131

132

133

134

135

Spontaneous events are the means by which clients become aware of activities of the network or of anomalous situations in the device (such as low battery conditions). Spontaneous-event notifications are multicast; when a spontaneous event occurs, the server implementation calls the notification callback functions of all clients that have registered for notification of the given event. While multiple clients may register for notification of a given event, each may register only one callback for that event – each registration replaces any previous registration by that client.

136

### 1.1.2 Result Events

137

138

139

140

141

Result events are the means by which clients receive the results of asynchronous operations. When the server completes processing of an asynchronous service request, it calls the notification callback function most recently registered for that event by the client. The client may register only one callback for a given result; each registration replaces any previous registration. Result Events are delivered only to the application that requested the service.

142

### 1.1.3 Application IDs

143

144

145

146

In order for events to be delivered to the right client, each client provides an application ID to the server when it registers for notifications. The ID is a unique integer value associated with each application or server that needs to receive events. No special semantics are associated with the value, but it must be unique for each client.

147

### 1.1.4 Callback Notification Functions

148

149

150

151

When an event occurs, the service implementation calls the callback notification function(s) registered for that event. The function is called with one argument, a pointer to a CelfMpEvent structure, which contains a fixed part with members that identify the type and subtype of the event and an open part that contains data fields appropriate to the specific event type.

152

153

154

The function is called in the process context of the client, so the client's internal namespace is available in writing the function. The method by which the system arranges for the process to be called in the client's context is outside the scope of the API definition.

Comment [sep8]: 20 What about the ownership? Can an application terminate a call previously initiated by another application? RESOLVED

Comment [sep9]: 22 Typo FIXED

Deleted: :

Comment [sep10]: 23 Where does the MPP API client register? REWORDED

Deleted: is delivered through a call by the service to

Comment [sep11]: 24 Sentence not easy to understand REWORDED

Deleted: when

Deleted: was registered

Comment [sep12]: 8 The text in this section implies an application may register more than one callback for a single event, however, Section 1.1.2 imposes an architectural constraint of only one callback per event (NB "... callback functions of all ..."). CLARIFIED

Comment [sep13]: 32 multicase: probably should be ... [1]

Comment [sep14]: 7 Must ... [2]

Comment [sep15]: 25 MPP ... [3]

Deleted: multicast

Deleted: MPP

Deleted: server

Deleted: requested

Comment [sep16]: 33 Is t ... [4]

Comment [sep17]: 9 The t ... [5]

Comment [sep18]: 17 If t ... [6]

Deleted: Each time

Deleted: a client registers for a t ... [7]

Comment [sep19]: 16 All o ... [8]

Comment [sep20]: 26 Inst ... [9]

Deleted: (client ID)

Comment [sep21]: 19 Sh ... [10]

Deleted: MPP server

Comment [sep22]: 27 ... [11]

### 155 1.1.5 Registering

156 A client requests notification of particular events by calling a registration function (which usually has a  
 157 name that starts with “start” and ends with “notification”), providing a application ID, event mask, and call  
 158 back function pointer as arguments. The event mask indicates which of the events provided by the  
 159 particular service the client is requesting notification for. There is a separate notification\_...\_start() function  
 160 for each cluster of services in the MPP API; for instance, the SMS service has a registration service  
 161 separate from the packet-switched communications service.

**Comment [sep23]:** 28 Why are client ID and application ID used for the same thing FIXED - application ID used

**Deleted:** client

## 162 1.2 Synchronous Service Interfaces

163 The processing of a synchronous request looks to the client like any other normal function call. The  
 164 implementation may do special processing to pass associated data between the client’s process context and  
 165 the service implementation’s process context, but that is outside the definition of the API. The client  
 166 process is blocked during the processing of the request and resumes execution with the assignment of the  
 167 provided result into the given variable (if appropriate).

**Deleted:** MPP server’s

## 168 1.3 Asynchronous Service Interfaces

169 To use an asynchronous service, the client must first call an interface to register to receive the notifications  
 170 associated with the service to be requested. The registration request would include a list of the events  
 171 requested and the callback function the server should call when the given events occur. A client may  
 172 register different callbacks for different events provided by the same service.

**Comment [sep24]:** 15 All of the asynchronous APIs specified in the various API specs must include a mechanism to cancel (outstanding) requests. DEFERRED - this is expected to change.

173 When the client makes an asynchronous request, it receives a result from that function call that indicates  
 174 whether the server accepted the request successfully. The client can then continue doing whatever  
 175 processing it has to do or can block waiting for the result to come back through a call to one of the callback  
 176 notification functions that it has registered.

177 When an event occurs (either completion of a service request or a spontaneous event), the MPP server will  
 178 check to see whether any clients are registered for that event and, if so, will arrange for the callback  
 179 notification functions that those clients registered against the event to be called in the application process  
 180 context.

## 2. Common Primitives

This section documents data types and values used throughout the sections of the API specification.

### 2.1 Constants

### 2.2 Enums

#### 2.2.1 CelfMpStatus

**Description:** Status returned by MPP API functions

**Definition:**

CELF\_MP\_STATUS\_OK: Successful completion  
 CELF\_MP\_STATUS\_APP\_ID\_ERR: Invalid Application ID  
 CELF\_MP\_STATUS\_EVENT\_SET\_ERR: The set of event is invalid  
 CELF\_MP\_STATUS\_CALL\_REF\_ERR: Invalid Call reference  
 CELF\_MP\_STATUS\_PS\_PDP\_TYPE\_ERR: Unsupported PDP type  
 CELF\_MP\_STATUS\_PS\_DENIED: Request rejected by network due to no  
 subscription to packet communication service  
 CELF\_MP\_STATUS\_ERR: Other error

**Comment [sep25]:** 10 Need a detailed description of "...CALL\_REF\_ERR". For instance what is an "Invalid Call reference"; what is its lexical constraint? TBD - coordination with PS/CS

**Comment [sep26]:** 1 Expand "...PS\_PDP\_TYPE\_ERR" to "...PS\_PDP\_TYPE\_UNSUPPORTED\_ERR" to clarify its intent. TBD - coordination with PS

### 2.3 Data Types and Structures

#### 2.3.1 CelfMpCallRef

**Description:** Call Reference for the current call. Service requests that establish calls return the Call Reference identifying that call.

**Definition:** typedef unsigned char CelfMpCallRef;

**Comment [sep27]:** 12 Need a detailed description of "Call Reference" - what is its semantics; how does an application determine its value. FIXED

#### 2.3.2 CelfMpEvent

**Description:** MPP notification events structure

**Definition:**

```

typedef struct {
    int category ;
    int subtype ;
    int info ;
    int subinfo ;
    union {

```

**Comment [sep28]:** 29 Don't use int! - et seq TBD

**Comment [sep29]:** 13 Must describe the following fields of the CelfMpEvent structure: category, subtype, info and subinfo. For instance what is their semantic; what is their expected value (e.g. application determines the values or the values are API-specific); are the value spaces extensible (or fixed via some enum), etc TBD

**Comment [sep30]:** 30 Avoid usage of union TBD

```
220     ...  
221     messages structures  
222     ...  
223     } data ;  
224 } CelfMpEvent;  
225
```

Comment [sep31]: 14 Add details about the "messages" field e.g. the value space, etc. TBD

### 2.3.3 CelfMpCallback

**Description:** Pointer to a callback function

**Definition:** typedef void (\* CelfMpCallback )();

### 2.3.4 CelfMpAppId

**Description:** Application ID

**Definition:** typedef int CelfMpAppId;

DRAFT

Page 10: [1] Comment [sep13]	scott preece	6/9/2006 9:50:00 PM
32 multicaser: probably should be 'multicast' FIXED		
Page 10: [2] Comment [sep14]	scott preece	6/9/2006 9:54:00 PM
7 Must describe "MPP Server" in this section and add a definition to section 0.3. FIXED		
Page 10: [3] Comment [sep15]	scott preece	6/9/2006 9:53:00 PM
25 MPP server not explained anywhere. The MPP server seems to be an integral part of the SW. Will there be an open source implementation? FIXED		
Page 10: [4] Comment [sep16]	scott preece	6/9/2006 9:52:00 PM
33 Is this description of 'last registered callback' only applicable to result notifications, or would this also apply to spontaneous notifications? CLARIFIED		
Page 10: [5] Comment [sep17]	scott preece	6/9/2006 10:20:00 PM
9 The text in this section implies an application may register more than one callback for a single event, however, Section 1.1.2 imposes an architectural constraint of only one callback per event (NB "... callback functions of all ...").CLARIFIED		
Page 10: [6] Comment [sep18]	scott preece	6/9/2006 9:56:00 PM
17 If two different applications register for the same event, for different purposes, only one will be notified (call log and phone application). Letting the system as it, require the applications to know which event has been already registered on the system. This prevents any 3rd party application to be easily coded. CLARIFIED - the one-notification model applies only to Result Events, which necessarily go only to the client that requested the service.		
Page 10: [7] Deleted	scott preece	6/2/2006 12:32:00 PM
a client registers for a result event, the server drops any previous registration for that event		
Page 10: [8] Comment [sep19]	scott preece	6/9/2006 9:58:00 PM
16 All of the APIs that require an app_id result in an implementation of the API having a dependency on the - currently undefined - application framework. This dependency must be broken i.e. the APIs must not have a dependency on the application framework. DEFERRED - this part of the model will probably change		
Page 10: [9] Comment [sep20]	scott preece	6/9/2006 9:55:00 PM
26 Instead of an application ID, use an dynamically assigned handle. DEFERRED - this is an area where we will probably change the model.		
Page 10: [10] Comment [sep21]	scott preece	6/9/2006 10:01:00 PM

19 Should the application to re-register once notified, or when an application has register for an event, it remain registered until explicit de-registration. FIXED in section 1.1.

27 Mechanism unclear, how functions will run in different process context.RESOLVED - no change required.