Leo One

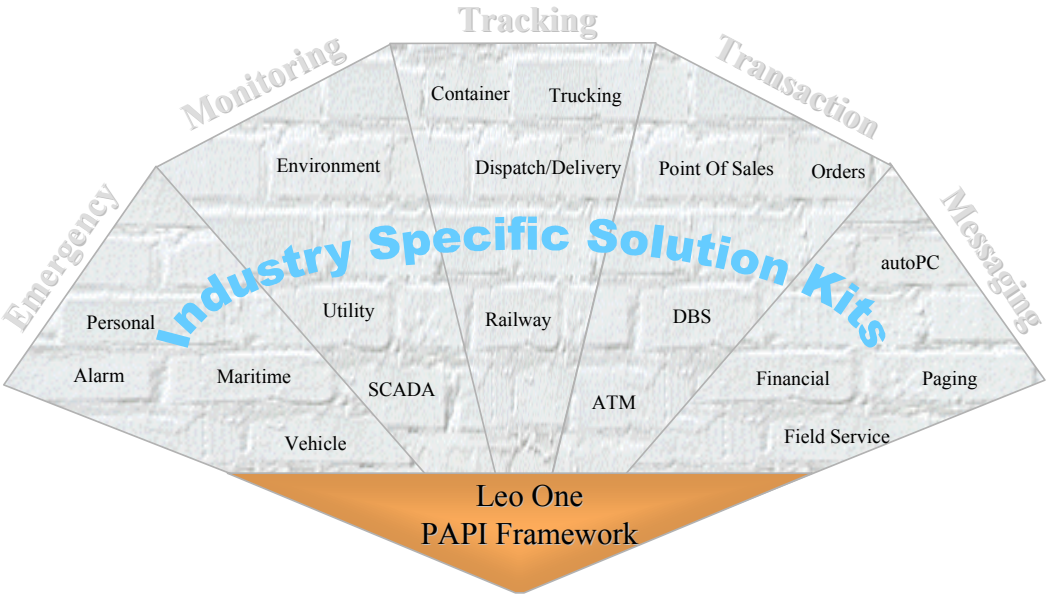
Public Application Programming Interface

Functional Requirements Specification

DRAFT

1 Purpose

This specification is a market requirements driven document that identifies the required functionality and objectives of a Leo One Public Application Programming Interface (PAPI) and development framework. The goal of the PAPI and framework is to facilitate application development and provide the foundation for numerous industry specific solutions in Leo One target markets.



2 Scope

This document is intended for those concerned with application development using the Leo One network. It is a marketing requirements document, not an engineering design specification, and avoids Leo One engineering or network nomenclature. It does not provide a complete understanding of the Leo One network architecture and lower level interfaces. It does not specify industry-specific or higher-level development kits, though does refer to these as example. The reader is expected to be familiar with the concepts of a Short Message Service (SMS) and Short Messaging Service Center (SMSC), which the PAPI framework is modeled.

The following documents are referenced:

- **[Leo One Network Segment Spec] description of Leo One Network architecture.**
- **[Leo One protocol Specification] description of Leo One Network protocol.**
- **[GSM SMS Point to Point 03.40] description of the point to point Short Message Service**
- **[SMPP Protocol Specification v3.4] description of the Short Message Peer to Peer Protocol**
- **[SMPPP Protocol Specification v1.1] description of the SMPP Provisioning Interface Spec.**

3 Concept Overview

3.1 The "*Leo One Global Short Message Service*"

Leo One is a small, low-earth orbiting satellite constellation designed for reliable two-way short message delivery. The 48-satellite Leo One constellation provides 24-hour coverage and the capability for near real-time operation on a global basis. It will be optimized to address the market for Short Messaging Service (SMS) solutions that place high value on a single worldwide network dedicated to business requirements.

SMS messaging standards are proving to be an ideal, lightweight communication solution for many applications and application protocols. Leo One will support SMS standards and solutions with a singular set of value added services characterized by:

- Superior availability (global and ubiquitous coverage)
- Superior reliability (e.g. all messages are acknowledged)
- Enhanced SMS functionality (e.g. Multicast)
- Secure communications (multiple layers of encryption)

Though a sophisticated satellite technology Leo One is a straightforward Global SMS architecture. Much less complex than current terrestrial SMS services that support roaming via a complex interweaving of competing networks originally design for voice services. As shown in Figure 1, Leo One subscriber devices anywhere in the world will be in constant radio view of the orbiting satellites. Satellites will communicate via a ground-based network infrastructure and set of receiving antenna stations, approximately 20, placed worldwide.

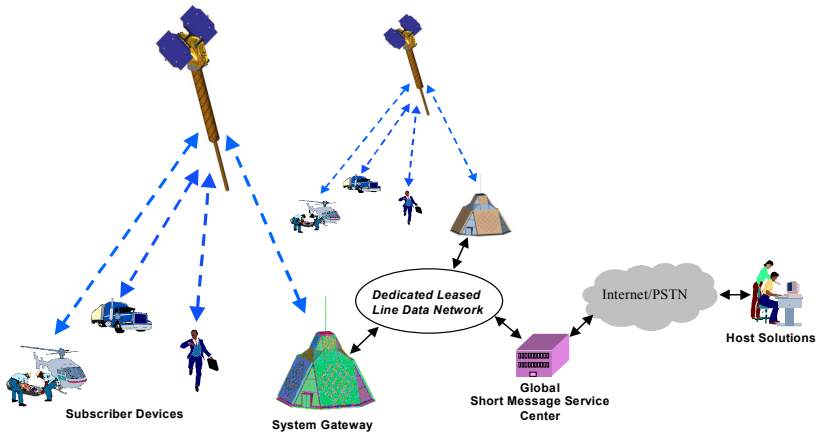


Figure 1: The Leo One Global SMS Architecture

Short messages to and from subscriber units are managed through *Short Messaging Service Centers* (SMSC), licensed and operated by authorized Leo One service providers.

3.1.1 Leo One Short Message Service Center

Like all SMS architectures, Leo One global SMS is a store and forward messaging service, where all messages and message services are centrally managed via a *Short Message Sevice Center* (SMSC).

As shown in Figure 3, typical SMSC products are designed to work with different network types (such as GSM, paging and CDPD) and allow easy integration and management of multiple network gateways.

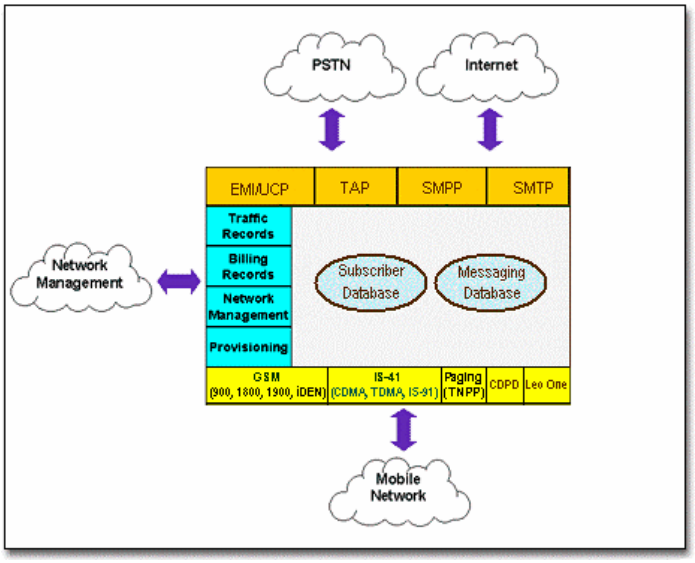


Figure 3 - Logica's Telepath SMSC Supporting Multiple Networks

SMSC configurations and management functions differ based on vendor offerings and service provider needs. The typical off-the-shelf SMSC will manage billing information, monitor message traffic and control access for each of its supported networks. Often, service providers will offer applications and services such as (email to SMS gateway functions) built in SMSC offerings as well (though technically speaking they are client applications/services, outside the scope of the SMSC core functionality). Below in Figure 2 shows a physical architecture of a Leo One service provider facility. The Leo One *Home Gateway (in yellow)* like any other network specific gateway, will an integrated part of the standard SMSC architecture.

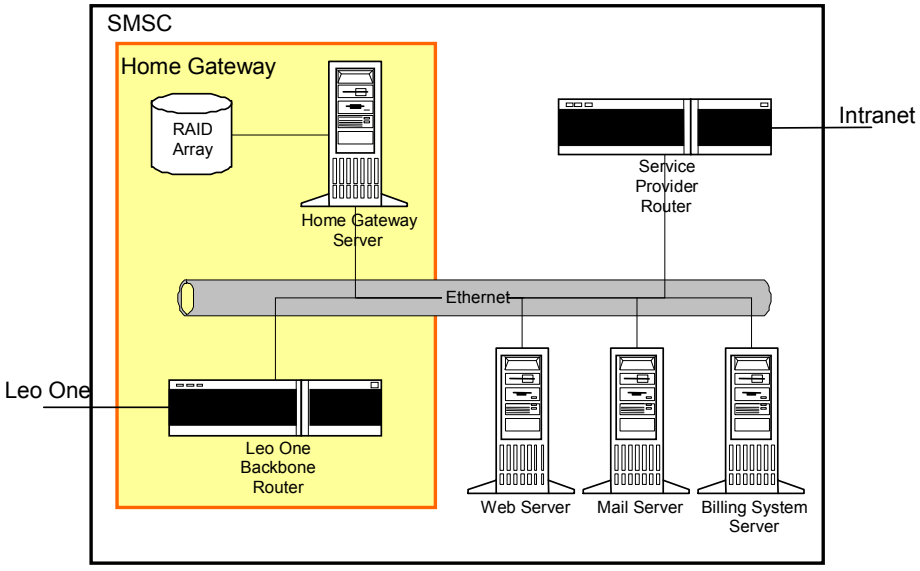


Figure 2 - The Leo One Service Provider Facility

3.2 Primary Objectives

The main objective of the Leo One Public Application Programming Interface (PAPI) is to provide developers and solution providers an easy development path using existing short message services, standards and tools. Thus, the requirements of a PAPI is by definition driven by the services and facilities of an SMSC, and will conform to standard SMS interfaces.

From the developer's perspective, the programming interface to Leo One will be the same as the interface to a typical SMSC supporting any other SMS network. A Leo One enabled SMSC will give service providers the ability to offer a true Global service that is cost effective and reliable.

As shown in Figure 3 , the underlying complexities of the Leo One network (satellites, protocols and gateway architecture) are hidden from the application on both the host and device end. The network interface from this perspective is like that of any other SMS network, with familiar SMS interface implementations on both the device and host end.

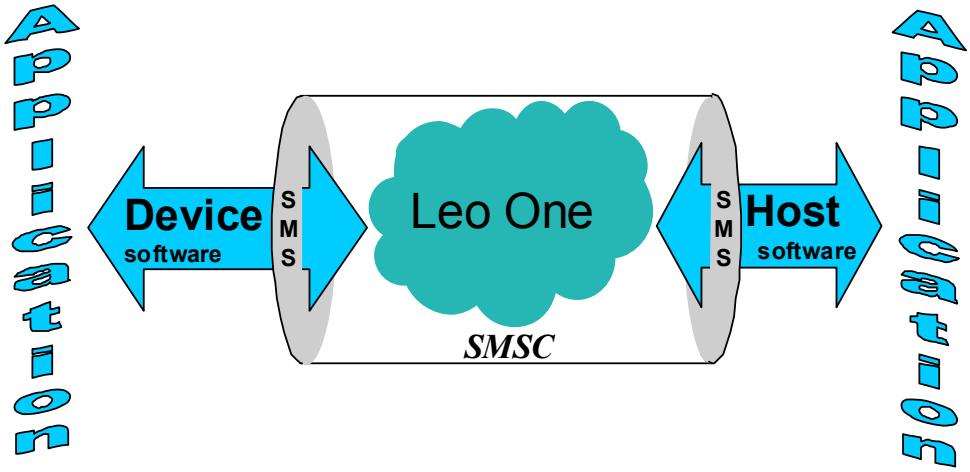


Figure 3 - Applications use SMS interfaces to access Leo One services

3.3 Customer Needs - Case Studies

3.3.1 FleetBoard

Customers of Leo One service providers (or Leo One service providers themselves) will typically be "solution providers" for specific corporate application needs.

An example of a solution provider (also called an Application Service Provider - ASP) that will benefit from Leo One service is Mercedes Benz's *FleetBoard*, a solution provider for the trucking industry.

Today FleetBoard provides Web host application services for corporate customers to manage their fleet operations. They are a full solution provider, providing FleetBoard hosted application, management and connectivity services to and from FleetBoard equipped vehicles.

The Web hosted application services provides all the features and power that a custom client server solution would provide within a corporation, but without the cost and complexities of managing it internally. Corporate users at the enterprise integrate their backend databases via

standard Internet technologies, and run applications hosted via any Web enabled platform (e.g. a PC), such as the service application shown here running within a Netscape browser.

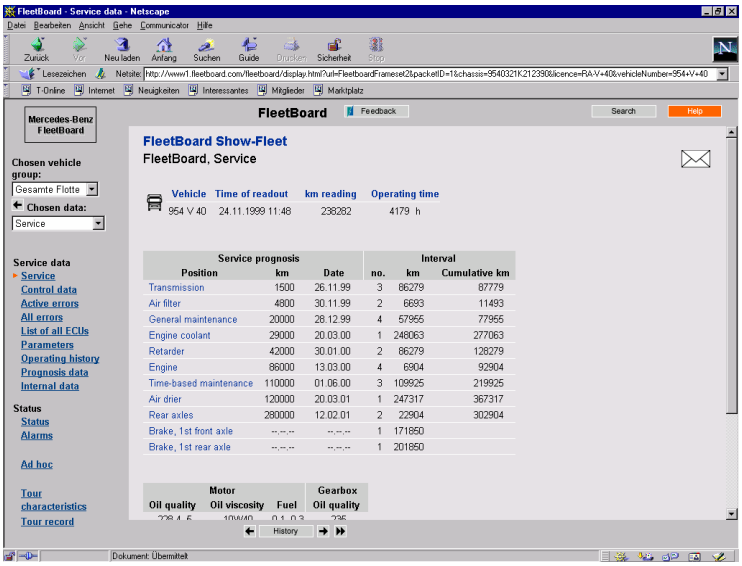


Figure 4 - A FleetBoard Web Client Application

The FleetBoard vehicle application provides many onboard capabilities as shown here.

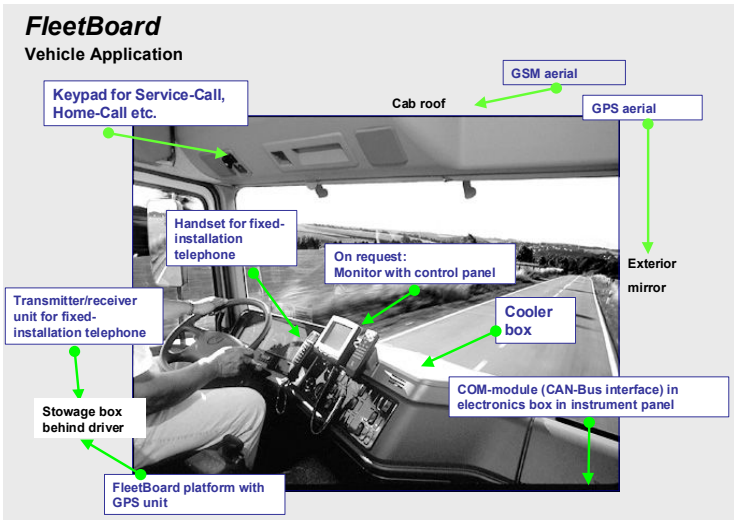


Figure 5 - The FleetBoard Vehicle Application

An embedded computer in the cab manages the various sensors, devices and the Human Machine Interface, and communicates to and from FleetBoard's host service via an SMS communicator (i.e. GSM modem).

A architecture of the vehicle application shown in Figure 6 shows the serial line connection to a GSM communicator, which is controlled via a standard AT modem device control language[1].

[1] See GSM 7.05 Equipment (DTE - DCE) interface for Short Message Service (SMS)

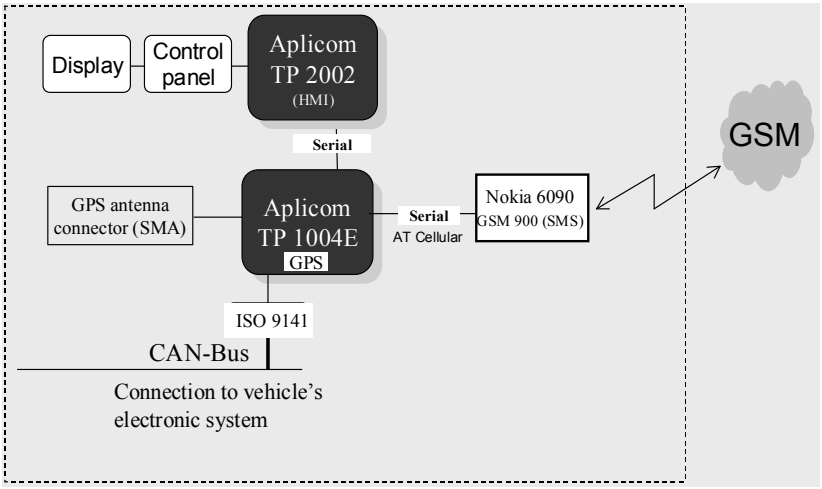


Figure 6 - FleetBoard Vehicle Application Block Diagram

How SMS messages are transferred to and from customer host takes an understanding of the SMS transfer protocol beyond the scope of this document, however, let's review the big picture from device to customer host end.

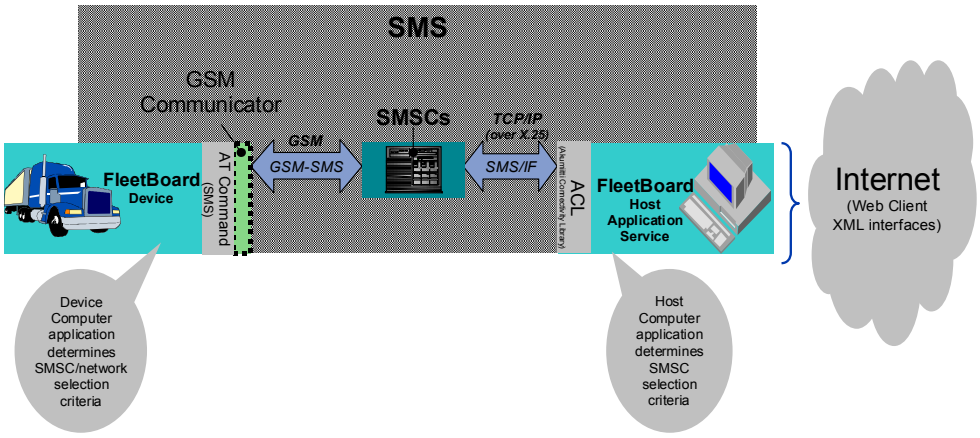


Figure 6 - FleetBoard Vehicle Application Block Diagram

As shown starting on the left of this picture, the vehicle applications will pass messages via the GSM communicator using an AT command library. SMS messages are then sent through the GSM mobile telephone network or networks and eventually received and stored by a destination SMSC. The SMSC then forwards the message on to the solution provider's (in this case FleetBoard's host server) via a standard TCP/IP connection link.

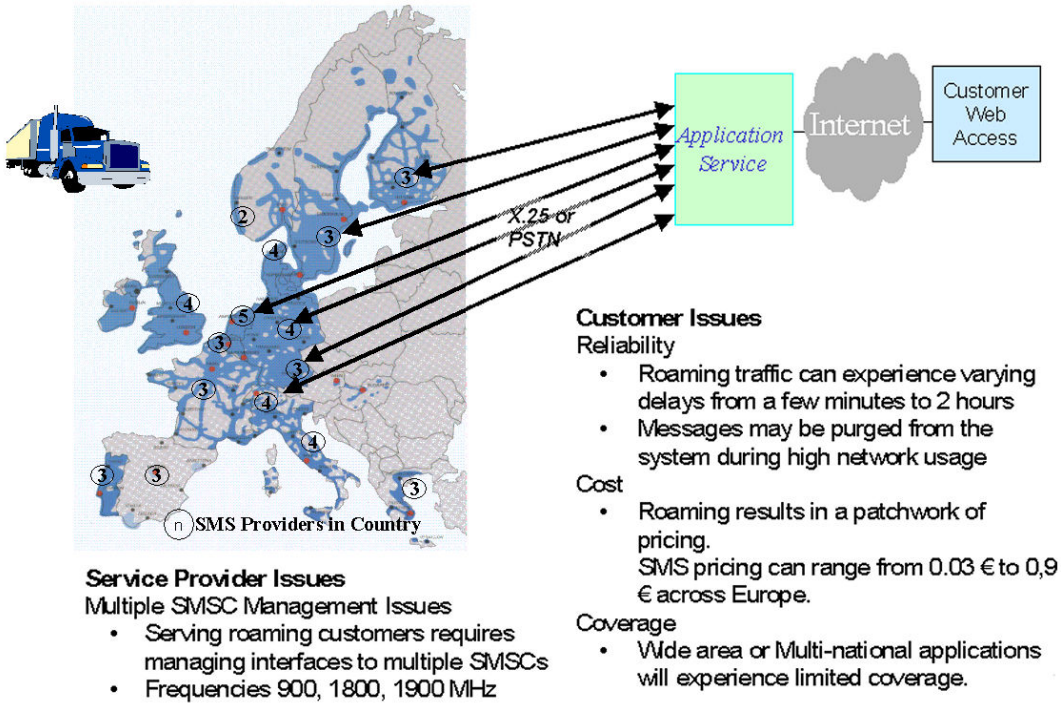
The interface between the SMSC to the FleetBoard server could be one of several specific standard Host-to-SMSC interfaces supported by various vendors. FleetBoard uses third party middleware messaging software (called ACL from Akumiitti Ltd.) that supports the various SMSC interfaces. However, the non-proprietary Short Message Peer to Peer (SMPP) interface has become the defacto standard and is supported by virtually all the SMSC product offerings, therefore many developers will not require middleware.

3.3.1.1 Adding Leo One Global SMS

Guaranteeing the quality of service that Fleetboard's customer's require presents many challenges. There are many issues with providing a high quality global service built upon a voice-oriented infrastructure made up of many local networks with competing interests. Though, SMS roaming

does exist and works well enough for many consumers' needs just as cellular voice services do, sending and receiving of SMS across multiple networks is fraught with complexity and many points of failure.

No coverage, and roaming outside of the subscriber's home network, not only has technical issues, but it also increases costs (roaming charges). Perhaps most encumbering is it puts the solution provider into the business of dealing with multiple networks and assuring service with cooperative agreements. Huge issues since there can be dozens of networks to deal with as shown in the following diagram of Europe.



Thus, there is a need for global SMS by the solution provider to offer predictable reliability and cost on a worldwide basis.

Adding Leo One to the vehicle solution is done with additional serial port connection to the Leo One communicator. To be as compatible with the application software, the control and command language of the Leo One communicator will be essentually the same as a GSM modem's.

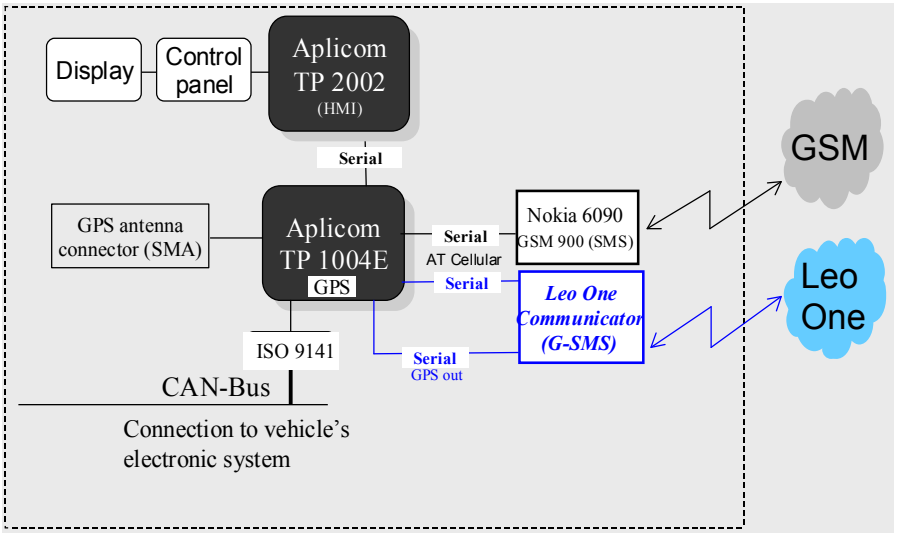
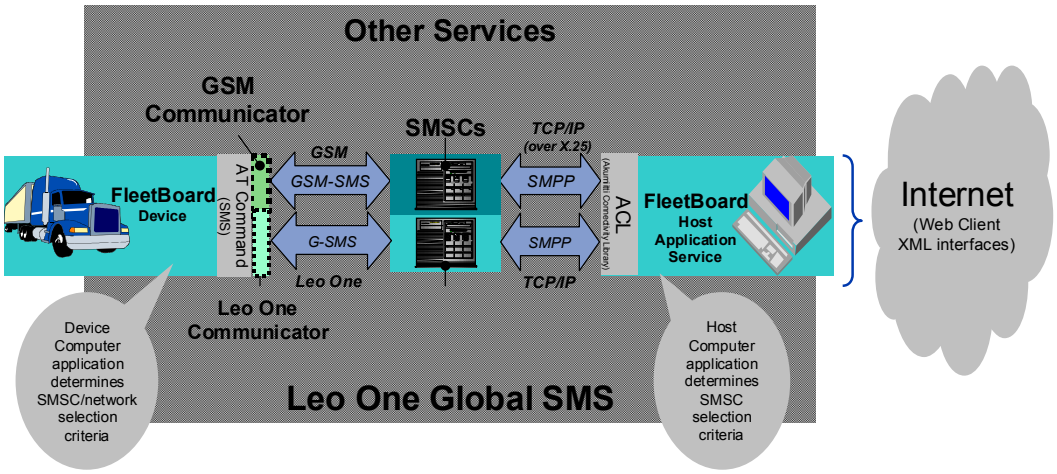


Figure 6 - FleetBoard Vehicle Application Block Diagram

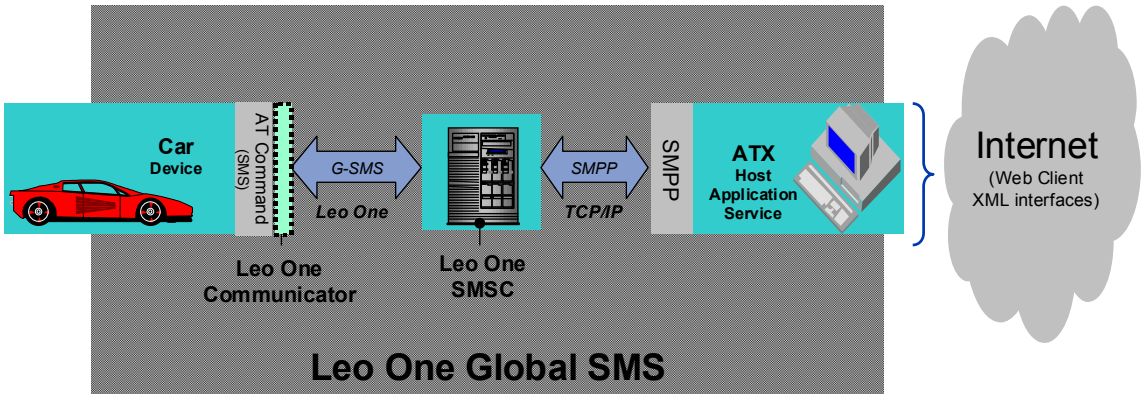
The FleetBoard application will then have two service options (Leo One and GSM) as shown below. Network selection will depend on criteria built within the application.



Note there are two separate network services and paths shown here, no intermingling of messages between networks within the system, and no magical routing decisions passing SMS messages between each different network type. Applications either use one or the other network service. The SMSCs maybe completely integrated within a single service provider facility, or they could be separate SMSCs manages by different service providers.

3.3.2 Passenger Cars

Some solution providers will not require or want terrestrial SMS networking. For example Daimler/Chrysler considers its core emergency telematics solution to be very cost sensitive. The added cost of an embedded terrestrial communicator is not practical. Leo One is the only cost-effective global solution. In this case the architecture is practically the same as shown before, albeit much simpler.



4 Interface Architecture Requirements

4.1 Common Message elements

The message data units on both the host and device ends shall have the following standard SMS elements:

- Priority (End-to-end Message Delivery Times)
- Less than 1 minutes (near real-time)
- Priority 2: Less than 5 minutes
- Priority 1: 5-30 minutes
- Priority 0: greater than 30 minutes
- Data coding type of the short message.
- Validity period - indicates the time period during for SMSC to consider the short message valid
- Service-Center-Time-Stamp - time of arrival of the short message at the center.
- Protocol-Identifier : refers to the higher layer protocol being used
- More-Messages-to-Send : information there is one or more messages waiting to be delivered.
- Schedule Delivery

4.2 Functions and Protocol

The device and host end interface frameworks will support the following class of functions.

- Control - initialization and application registration, local h/w & process configuration/control.
- Message Request - request data delivery to and from the network.

- Message Notify - indication to application of message delivery/status.

The SMS functions and protocols are based on the GSM SMS (device end) and SMPP (host end) standards previously referenced. The required operations and functionality are listed below.

Note that the phrases "Host Servers" or "Host Applications could mean applications/services that run within a service provider facility as well as customer host (enterprise) solutions.

4.2.1 Device End

The device control language between data terminal equipment and data communication equipment will be standard AT cellular/SMS. This requires the Leo One communicator to have an AT translation layer at a minimum, as with cellular modem radios. Translation, like all vendor specific modems are implementation dependant, it is not require that all communicators (cellular or Leo One) respond exactly the same in all implementations but follow the general guidelines.

Though the AT commands required are shown here along with the stated functions, it is envisioned integrated application/communicator solutions (sharing the same CPU) will implement these operations in different programmatic ways and not require this specific mechanism.

4.2.1.1 Control - interface for initialization, local h/w & process configuration/control.

The following control functions shall be supported:

- AT+CSCA Service Center Address
- AT+CNMI New Message Indications to TE
- ATE Command Echo
- AT+IFC DTE-DCE Local Flow Control
- AT+CPMS Preferred Message Storage
- AT+REG Is device registered with network?
- AT+COPS Network identification
- AT+CGMI Request Manufacturer Identification
- AT+CGMM Request Model Identification
- AT+CGMR Request Revision Identification
- AT+CGSN Request Product Serial Number Identification
- AT+CSQ Signal Quality

4.2.1.2 Request - request data delivery to and from the network.

The following request functions shall be supported:

- AT+CMGR Read Message (string is Message)
- AT+CMGS Send Message (string is Message)
- AT+CMGL List Messages
- AT+CMGD Delete Message

4.2.1.3 Notify - indication to application of message delivery/status.

The following status and notification functions shall be supported:

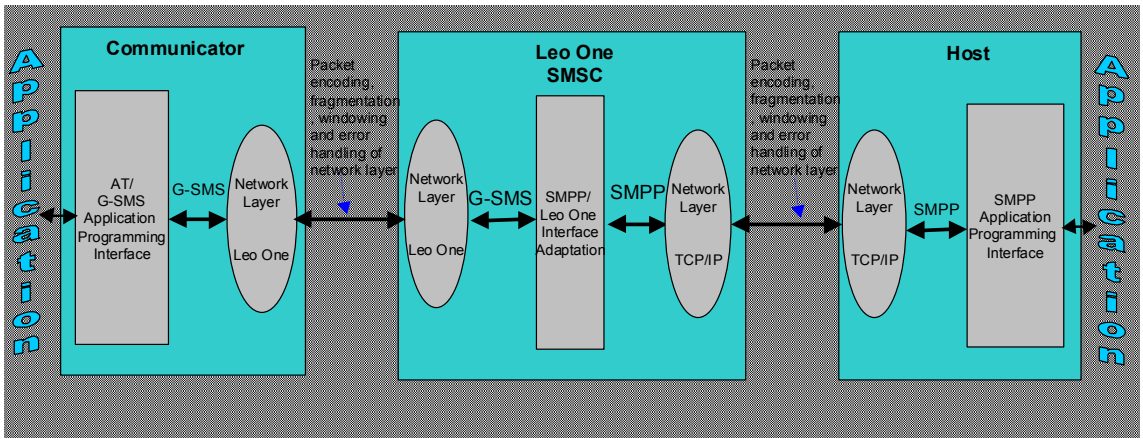
- +CMTI, new message
- +CDS new status

4.2.2 Host End

The underlying transport interface between the Leo One enabled SMSC and a host server application will be based on a TCP/IP or X.25 network connection.

The message protocol will be based on SMPP, an application layer protocol and not intended to offer transport functionality. Therefore, the underlying network connection will provide reliable data transfer from point to point including packet encoding, windowing, flow control and error handling.

The following diagram illustrates the generic SMPP interface implementation between a Host and Leo One enabled SMSC.



The Leo One adaptation layer shown above will handle the translation to and from the Leo One protocol stack, (and adjustments such as segmentation of the messages before passing to and from the SMPP layer).

The complete SMPP interface specification is given in Appendix A.

Though the SMPP procedures are used here along with the required functions, it is envisioned other Host to SMSC interfaces that implement the same functionality may be supported.

4.2.2.1 Control - Host interface for initialization, local process control.

The bind function will be provided is to register an instance of a host server application with the SMSC system and request an SMPP session over the Leo One network connection for the submission or delivery of messages. Thus, the Bind operation may be viewed as a form of SMSC

login request to authenticate the host server entity wishing to establish a connection. A Host may bind to the SMSC as a Transceiver. The following status and bind functions shall be supported:

- bind_transceiver
- unbind
- enquire_link

4.2.2.2 Messages sent from Host Server to SMSC

A host server, which sends short messages to an SMSC, shall be connected to the SMSC as a Transmitter or a Transceiver.

The following host to SMSC messaging functions shall be supported:

- submit_sm - submit a short message to the SMSC for onward transmission to a specified device communicator
- data_sm - used to transfer the following types of special messages to the Host:
  - SMSC Delivery Receipt.
  - Delivery Acknowledgement.
  - Intermediate Notification.
- submit_multi - used to submit a message for delivery to multiple recipients or to one or more Distribution Lists (e.g. multicast).

In addition to submission of messages to the SMSC, a host server shall be able to perform the following functions using the message identifier returned by the SMSC in the message acknowledgement:

- query_sm - Query the SMSC for the status of a previously submitted message
- cancel_sm - Cancel delivery of a previously submitted message that is still pending
- replace_sm - Replace a previously submitted message that is still pending

Messages sent to the SMSC by a Host must, when received, be acknowledged response by the SMSC.

4.2.2.3 Message Response from SMSC to Host Server

The responses for a message submission to the SMSC will include a message identifier (which must be a unique handle assigned to that particular message). It will also include a status which informs the host whether the submitted message is valid (i.e. accepted by the SMSC for onward delivery) or invalid. In the latter case, the SMSC will return an appropriate error.

The following SMSC to Host responses shall be supported:

- submit_sm_resp - response to submit_sm
- data_sm_resp - response to data_sm

- query_sm_resp - response to query_sm
- cancel_sm_resp - response to cancel_sm
- replace_sm_resp - response to replace_sm

4.2.2.4 Provisioning Messages from SMSC to Host Server

The following provisioning functions shall be available for management of subscribers and distribution lists. Note these functions maybe private or public depending on Service Provider requirements.

- add_sub : issued by the host server to add a new subscriber to the system.
- del_sub : issued by the host server to delete the profile of an existing subscriber.
- mod_sub : issued by the host server to modify the profile of an existing subscriber.
- enquire_sub : issued by the host server to obtain the provisioning data stored for a subscriber.
- add_dl : issued by the host server to add a new distribution list to the system.
- mod_dl : issued by the host server to modify the profile of an existing distribution list.
- del_dl : issued by the host server to delete an existing distribution list from the system.
- list_dls : issued by the host server to list the distribution lists that a particular person has provisioned in the system.
- view_dl : issued by the host server to view the members of a particular named distribution list already provisioned in the system.

4.2.2.5 Notify - indication to Host Server of Message Delivery/Status.

The following functions shall be provided from the SMSC for Host status notification and message delivery:

- deliver_sm - issued by the SMSC to send a message to host server. Using this command, the SMSC may route a short message to the host server for delivery. In addition the SMSC uses the deliver_sm operation to transfer the following types of short messages to the host:

- SMSC Delivery Receipt. A delivery receipt relating to a a message which had been previously submitted with the submit_sm operation and the host had requested a delivery receipt.

- Delivery Acknowledgement. The user data of the SME delivery acknowledgement is included in the *short_message* field of the deliver_sm
- Intermediate Notification.
- alert_notification

4.2.2.6 Messages sent from SMSC to Host Server

The SMSC will deliver short messages to a Host Application. In this case, the host must be connected to the SMSC as a Transceiver.

The following functions shall be available to send from an SMSC to a host application:

- deliver_sm
- data_sm

Mesages delivered to host server application by the SMSC must be acknowledged with a response by the host when received (exceptions to this rule are the alert_notification).

4.2.2.7 Provisioning Messages Sent from SMSC to Host Server

The following provisioning responses shall be available for management of subscribers and distribution lists. Note these functions maybe private or public depending on Service Provider requirements.

- add_sub_resp - Response to “add_sub”.
- del_sub_resp - Response to “del_sub”.
- mod_sub_resp - Response to “mod_sub”.
- enquire_sub_resp - Response to “enquire_sub”.
- mod_dl_resp - Response to “mod_dl”.
- add_dl_resp - Response to “add_dl”.
- del_dl_resp - Response to “del_dl”.
- list_dls_resp - Response to “list_dls”.
- view_dl_resp - Response to “view _dl”.

4.2.2.8 Message Response from Host Server to SMSC

The message response from a receiver must preserve the message transaction identifier (contained in the *sequence_number* parameter) sent by the SMSC. The response must also include the command status which informs the SMSC whether the message delivered to the host was valid (i.e. accepted by the Host application) or invalid. In the latter case, the host server application should return an appropriate error status.

The message responses that may be sent from a host receiver to the SMSC:

- deliver_sm_resp
- data_sm_resp

4.2.2.9 Error Handling

All functions consist of a request message and associated response, with the exception of the alert_notification (for which there is no response).

In all other cases, the receiving entity must return the associated response to an request message, indicating that the original message has been received at the destination. Until such a response is received by the originator, it must be assumed that the message has not been received at the destination.

In the event that the original request is found to contain an error, the receiving entity must return a response with an appropriate error code inserted in the in the response message.

4.2.2.10 SMPP Timers

To ensure the efficient exchange of message transactions, it is recommended that each SMPP session be managed using configurable timers on both the Host and SMSC communicating SMPP entities as follows:

- An SMPP session initiation timer to ensure that when an Host initiates an SMPP session, that this occurs within a specified period after opening a network connection to the SMSC.
- An SMPP session timer to enable either the Host or SMSC request the SMPP session status of the other communicating SMPP entity via the enquire_link command.
- An SMPP inactivity timer which should specify the maximum period after which time, if no SMPP messages are exchanged, the SMPP session may be dropped gracefully.
- An SMPP transaction timer which specifies the time lapse allowed between an SMPP request and the corresponding SMPP response.

4.2.2.11 Store and Forward Message Mode

SMPP allows for three message mode options. Leo One PAPI only requires the conventional approach to SMS which is to store the message in a SMSC storage area (e.g. message database) before forwarding the message for delivery to the recipient entity. With this model, the message remains securely stored until all delivery attempts have been made by the SMSC. This mode of messaging is commonly referred to as “store and forward”.

The “store and forward” delivery mechanism via the submit_sm operation, enables the Host server to send a message to the SMSC where it is stored until it is successfully delivered or until the message validity period expires.

The store and forward mode is also supported via the data_sm operation.

The “store and forward” message mode also facilitates subsequent functions on the stored short message such as query_sm, replace_sm and cancel_sm. The submit_sm function also facilitates “replace-if-present” functionality which requires that the original message be stored on the SMSC.

Note: To determine the eventual outcome of the SMS delivery, the Host server must request an SMSC Delivery Receipt in the submit_sm or data_sm operation.

4.2.2.12 Message Types

In addition to "normal" short messages, special messages can be transferred between the Host server and the SMSC in a *submit_sm, deliver_sm* or a *data_sm* operation. The message type is defined in the functional parameters:

SMSC Delivery Receipt

This message type is used to carry an SMSC delivery receipt. The SMSC, on detecting the final state of a registered message stored in the SMSC, should generate a receipt message addressed to the originator of the message. The SMSC Delivery Receipt is carried as the user data payload in the SMPP *deliver_sm* or *data_sm* operation.

The source address will be taken from the destination address of the original short message which generated the delivery receipt.

The destination address will be taken from the source address of the original short message which generated the delivery receipt.

Intermediate Notification

An intermediate notification is a special form of message that the SMSC may send to a host server for a mobile terminated message delivery. It provides an intermediate status of a message delivery attempt.

Typical uses are to report the outcome of the first delivery attempt that has failed but the message is still held in the SMSC for further delivery attempts.

4.3 Non-Requirements

- "SMS" the word itself has many different connotations, one of which is the SMS application on most European phones. This document has nothing to do with consumer SMS. It is about an implementation of the SMS messaging standard, specifically the application and transfer layers which are network and application independent.

- A GSM/SMS phone does not send a message to a Leo One terminal nor vice versa. I.e. no routing or switching decisions are required between different types of SMS networks within the SMSC (or gateway) configuration. Applications themselves on both ends decide which network service to use using the functional interfaces defined.