Capability-based modeling of Services in Mobile Systems

Hrishikesh Sharma, Kishore Kumar O.
Sasken Communication Technologies Ltd.

Contents

Abstract

As the digital wireless communication systems evolve towards 3rd Generation systems, the usage of such devices continues to diversify beyond, say, making and receiving a call from anywhere. To support applications for such uses, standardization bodies(e.g. 3GPP) are actively involved in defining range of new services(e.g. MMS). The scale and heterogeneity of services, which need to be considered while designing solution for a mobile device, has therefore increased; and is expected to increase further, as standardization progresses. Easiness of integrating another service in a product line becomes an important goal of architecting and designing process. In this paper, we propose a new 'framework'ed approach suitable for UMTS era, which we used to meet this goal in our middleware design project, MobiTAAL. This approach uses capability definitions for synthesis of a group of services. The focus of this approach is on defining the service components on the User Equipment side of mobile systems. We also discuss the results from modeling effort for the middleware, which we designed. This work was done in partial collaboration with Indian Institute of Technology, Delhi, by Sasken Communication Technologies Ltd.

1 Introduction

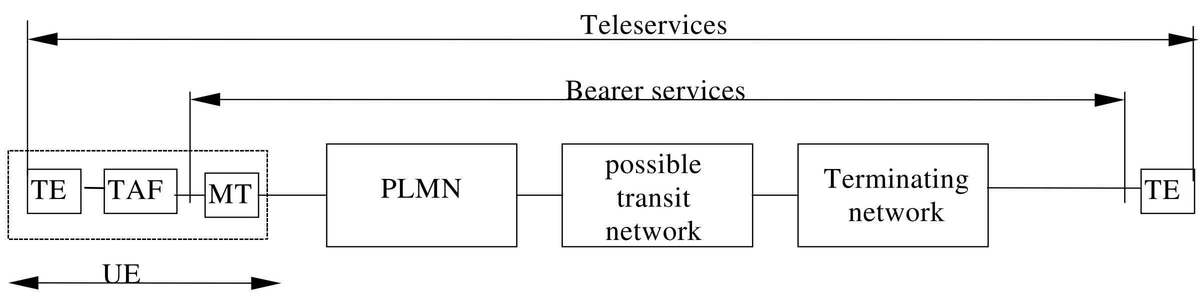
A service in context of mobile **handheld** devices is a set of functions offered to a user by an organization, typically the network service provider. Here the user should be taken as any logical, identifiable entity, which uses such services. Services, by definition, are driven by the needs of the user. Hence there are some models available[8] for service engineering in (mobile) telecommunication systems.

1.1 Basic Services Model

The most commonly used model in GSM evolution systems is the basic model, in which services are divided in two broad categories:

- bearer services, which provide the capability of transmission of signals between various access points; and
- teleservices, which provide the complete capability, including endpoint equipment(handheld) functions, for communication between users according to protocols established by agreement between network operators. Such communication is required for some kind of information access.

The communication link between the access points may consist of a public land mobile network(PLMN), one or more transit networks and a terminating network. The following figure[8] illustrates these definitions.



UE: User Equipment
MT: Mobile Termination
TE: Terminal Equipment
TAF: Teminal Adaption Function

Figure 1: Service Architecture in GSM networks

A third category called supplementary service modifies or supplements a basic teleservice. Consequently, it cannot be offered to a user as a stand-alone service.

1.2 Limitations of Basic Services Model

Standardization body for existing wireless communication systems such as GSM have largely standardized the complete sets of teleservices, applications and supplementary services, which they provide. As a consequence of this detailed service engineering efforts in standardization, *substantial* re-engineering of internal design is often required to enable new services to be provided. This limitation occurs because the "solution" designers limit themselves to support an optimal implementation of the plethora of such standardized "services", making the design somewhat inflexible. Therefore the cost of design changes for new services is high. Prevalence of functionally similar devices having comparable costs has made it more difficult for operators to deploy new/alternative services, and differentiate from one another. But such differentiation is what drives marketing efforts, thus creating a paradoxical situation. An example of such kind of service engineering is the way supplementary services got defined, one-by-one by various operators, and hence resulted in phones having same services, though with varying menus.

1.3 Our Approach

Sasken has been a leading provider of mobile terminal software, especially the protocol stack and middleware. Technology-wise, we have chosen to focus on the various GSM evolution technologies, i.e. GSM, GPRS and UMTS(3G). Right from inception, our focus has been on making re-usable components for all mobile software, to tackle the cost escalation factor described above. This paper covers the method and results for the middleware solution, called MobiTAAL.

In a brief technical report[7], the working body of 3GPP suggested usage of logical components called "service capabilities" for handset design. We did a detailed interpretation of what it could imply in context of handset design, and came up with our own model. We were successfully able to use the results of this modeling effort in our middleware, which is successfully deployed in certain ranges of handsets worldwide. To the best of our knowledge, this effort is the first concrete effort of designing middleware using a framework-ed approach of "service capabilities".

2 Service Capability Model

In this model, there are three kinds of capabilities[4], which span the functions of mobile handheld system. They are terminal capabilities, service capabilities and access capabilities. Service capabilities are defined as those functional elements that can be used either singly or in combination to deliver services to the user. The characteristic of service capabilities is that their logical function can be defined in a way that is *independent of the implementation* of the mobile system. Examples: a data bearer of 144 kbps; a high quality speech teleservice; an IP teleservice; a capability to forward a speech call. We have used such kind of abstraction to model and design reusable components for middleware.

Based on such a classification, the applications (or clients) are defined as services, which are designed using service capability features. The concept of service is a distributed phenomenon in mobile handheld systems. While many of the services are mainly controlled by the network, they can only be offered to the user on the terminal side. Hence many of the service capabilities reside in the middleware component of the handheld, and can be made visible to the applications through an application interface.

These service capability features are further distinguished in mainly two ways(refer [4] for details):

- **Framework service capability features:** these shall provide commonly used utilities, which are necessary for the non-framework service capability features to be accessible, secure, resilient and manageable. They provide functionality that is independent of any particular type of service. Examples of such procedures are authentication, registration, notification, etc. As evident, the corresponding service capabilities set up the context of usage of various other services.

- **Non-Framework service capability features:** these shall enable the applications to make use of the functionality of the underlying network capabilities (e.g. capabilities for user location services). Thus they can be dependent on the type of service in question. In a generation of evolution-wise related technologies, such features overlap for various technologies. Hence our job was made easy, because we chose the modeling for GSM evolution technologies.

3 Premises for using the Model

We applied the above model by doing an *abstract characterization* of run-time behavior(functions) of *GSM evolution* mobile systems. Based on the run-time behavior, the service components were identified and developed in such a way, that they covered most of the possible service permutations.

We define and use the following new terms:

Context-based information access: is characterized by out-of-band signaling[1] by mobile device, prior to the information exchange. This logical context is called a ***call***.

Context-less information access: is characterized by store-and-forward mechanism for the information to be exchanged. The unit of storage, or forwarding, is known as a ***message, or packet***[2].

The characterization is done by including following steps, which can be envisaged as part of device operation(the steps form just a partial order):

1. System booting
2. Restoration of context and data preserved across power cycles
3. Activation of Subscription
4. Registration into network
5. (Anytime) configuration or re-configuration of information transfer and access attributes
6. Setup of context, for context-based information access and exchange
7. Access and exchange of information(continuous, or intermittent)
8. Context-less information access and exchange, based on activation of store-and-forward mechanism

[1] Signaling is the transfer of information between communication components necessary to provide a service and maintain it.

[2] The difference between a packet-access to information and message-based access is that message access is typically short, and comprises of only small units of information. Packet-access implies a switched context-less, but continuous flow of information.

9. Asynchronous notifications of various events, which can impact the mode of information access. Such notifications typically bring changes in configuration and structure of information access.

10. Change in control of information, and it's associated context, if any. At times, SIM applications can take over the control from the applications, which have human interaction involved.

11. Change in interface mechanism[3] for above steps

12. System shut down

4 Stepwise Function Description

As pointed out in [2], context awareness is an integral part of the any mobile middleware. Hence, for each *runtime* step identified in previous section, we provide the functional details worked out by us. This description was used to put up the service capability model for MobiTAAL.

4.1 System Power up

In this process, some configuration is restored in local context of the threads within corresponding middleware service entity. A request is made to an asynchronous peripheral(NVM) to read the saved context, and data, if any. The data is provided to the service entity asynchronously.

The data in question here can be local data, or data which is part of a store-and-forward mechanism. For example, message stores, dialing number tables, personalization indicators etc. can be restored.

4.2 Activation of Subscription

When SIM is inserted in the handheld device, certain subscriber signaling procedures are done, directly via application towards the user. After these verification operations, the SIM service table is available for the service entity to read. There is a concept of service activation table within the service entity. The data storage dependence(SIM can also be treated as a non-volatile storage device, with optional and mandatory partitions) will be referenced within the service activation table. Based on such references, alternate storage decisions(for context and data) can be made at initialization time.

The service activation table works on the aspects of service provision, service enabling and service handling capability(provided as part of terminal profile) at run time.

If enabled, then at the time of subscription activation, subscriber personalization checks can be done. This is based on the private key called IMSI. Table entries are matched by service domain, in case the personalization is active. In case table matching fails, the mobile system is made to operate in emergency mode.

[3] Can be static as well as run-time. See section 4.10.

Certain capabilities such as dialing numbers are activated only if user wants it, and there is storage on SIM(and nowhere else) for the same.

4.3 Power-on Registration

After context re-storing at boot time, the mode of selection of network is re-enabled within service entity. The radio resource entity searches for visible networks, and forms a cumulative list. In case of automatic selection mode, whenever a network is found in the priority order(typically derived from SIM), it selects that network for registration.

For the domain of circuit-switched connections, the mobility management entity does the registration procedure autonomously, on the selected network. The user may do registration in packet-switched connection domain anytime, on the network selected at that moment. Local information is generated and provided for registration, to mobility management entity for that purpose.

In case of manual selection, a list is provided by radio resource entity, via the service entity and the application, to the user. The user may select one of these, and inform service entity to proceed with the selection and subsequent registration on the selected network.

A pre-selected network may be provided by application, in case it decides to ignore any lists of network for selection. After a certain period, the service entity should use this pre-selected network as the chosen one, and proceed with the selection and subsequent registration.

The list should be filtered by service entity, to provide only those networks for choice, to which the system has been personalized for the information access.

It is possible that the selected network is no longer visible by radio after selection. In this case, the mobile system is made to operate in emergency mode.

4.4 Configuration(or re-configuration) of Information Transfer and Access Attributes

There are lots of configurations(similar to profiles in [2]), which will be held by the individual service components for reacting in various contexts. Following are few important functions related to usage of such configurations:

1. **Configuration, or reconfiguration of information *exchange* attributes for context-based information access.** The context can be characterized by different kinds of active data elements, whose nature is pre-specified. When the context is being set up, this configuration is used. This configuration has to be provided to context-creation entities in applications, on a per to-be-activated context basis. The active configuration can also be modified or reconfigured. In some case, the network decides to do the reconfiguration, and the context manager component has to simply accept it. In other cases due to user's reaction to context, context managers are provided with the new would-be-configuration plus a trigger. The trigger is made to try out protocol-based negotiation for configuration modification with the network.

2. **Configuration, or reconfiguration of information *transfer* attributes for context-less information access.** The description is on similar lines as previous point. For example, this could include configuring the validity period, the service center etc. for a short

message, which travels out to it's destination. The entity in charge of creating and ensuring delivery of message should be provided access to such stored configuration.

3. **Storage for information transfer and/or access attributes.** This information is used by context managers, or message deliverer entities. As described previously, it is used for setting or re-setting the information exchange context[4], or piggybacking along with context-less data flow[5]. This data may be provided just before the methods used by corresponding entities, or may be stored somewhere(pre-specifying) to be provided to these entities for their usage. Eventually, these entities should know where to get the attributes from.

4. **Type of context for information exchange.** At times, during the configuration as identified in (2), application may just provide one or more attributes required to characterize the context. In fact, it may provide one or more "similar" attributes. These are not the real context attributes, but which can lead to derivation of the actual attributes. Based on such disclosures, the context for information exchange can be calculated by a provider entity. This context is then used by context managers. QoS mapping from traffic type, and derivation of bearer capability for circuit-switched connections is example of this.

5. **Type of information.** For certain information exchange, this is required as well. It can be configured under the terminal capabilities, or terminal profile. For example, the language support for messaging depends on the availability of corresponding font library(in MExE, it is possible to change the language support, after downloading fonts). Otherwise speaking, rough information about terminal capabilities such as support for audio/video display(EMS), language, fax etc. is needed in case of functionality of handheld equipment is split[5]. At the time of setup of **remotely** initiated context for information exchange or transfer, these tables can be looked up by the entities dealing with either the **remotely** initiated context, or the message receiver entity.

6. **Number of outstanding information flows.** This configuration is required to upper bound both the context-based and context-less information transfers. In case of context-based transfer, this defines the overall contexts, which can be managed by the corresponding entities. In case of context-less transfer, it defines the maximum number of store-and-forward units to be allocated and managed by the database entity. The former decision is typically only at initialization time; the latter decision can be (re-)configured by application towards the storage entity anytime.

7. **Storage for context-less information.** The information logged here may be used by filters. It may also be used for display. It may be used for transmission, and/or replying, in which case information is forwarded to the message deliverer entity. The storage type and the units may be configured. The user entities should be aware of this storage type.

8. **Forwarding mechanism for context-less information.** This configuration is provided by user itself, and can be used to configure the message receiver entity for immediate transfer of the message.

9. **Configuration of addressing and address plans for information exchange.** Address configuration is true for both context-based and context-less information exchange. For context-based information exchange, the addressing is specified at the time of creation of context, and is used by context managers. For context-less information transfer, addressing is used by deliverer entity, at the time of sending out the message. There are many possible

---

[4] Signaling data
[5] Information data

schemes and types. For example, international or national numbering type can be configured and provided to circuit context managers. Similarly, E.164 or Ipv4 numbering plan can be provided to context-less or packet-context managers, respectively. Finally, there may be local scheme, such as directory scheme based on ADN(abbreviated dialing numbers), which can be provided to maintain a table. These tables can be used to store the configured addresses for information exchange.

10. **Configuration for information access domain.** At times, the application will wish to only have information access and/or transfer within a physical or logical networking domain. In that case, filters are created and specified, to weed out contexts, which do not belong to this domain. Alternatively, for context-less information transfer, there will be active filters on the meta-data such as data coding scheme, which accompanies the context-less information. These configurations may be active, or inactive, based on the user's choice. The entity, which owns the service activation table, acts as a filter-providing entity for domain screening. Classic examples of such screening is BDN(barred dialing numbers), FDN(fixed dialing numbers), ACL(APN control list), CUG(closed user group), TFT(traffic flow template: a collection of packet filters), language preference for cell broadcast, and broadcast (agency) identifier(alternatively the message identifier) for cell broadcast.

11. **Configuration for information mapping.** This holds true for context-less information flow only, which is store-and-forwarded. For example, EMS message contents, email-via-message contents, contents of messages in language different from that expected by terminal etc. need to be mapped to different domains. Such mappings are configured for the overall storage. The entities, which fetch the stored information, which route the information while immediately delivering to the application, or which construct a message to be sent, need to use this mapping algorithm. This algorithm is a non-learning algorithm, and hence does not have context-data associated to it.

12. **Mode of network selection.** The user may anytime force a change in the configuration of network selection mode. If it is from manual to automatic, then the user has just relinquished the control, and that needs to be simply conveyed to the radio resource entity. If it is from automatic to manual, then the same interactions as described previously for power-on registration hold true. Only thing omitted is the domain to which attachment happens; for the location information for each domain is seamlessly transferred to the (new) network.

13. **Keys and Passwords.** These are required for activation of certain services. There is an indicator, and an associated key or password for certain services such as personalization, supplementary services etc. This table is maintained according the user's inputs. At times such as registration of certain supplementary services, invocation of personalization procedures by the entities dealing with the associated procedures, these tables are fetched and used.

14. **Configuration of data management.** In perspective of service components, this holds true for context-less information only. The received or to-be-sent data may be decompressed, or compressed, respectively. The data, while storing(if required), will not be uncompressed data[6]. Hence the storage entity is triggered to compress or uncompress the data, while storing or retrieving. In the similar way, the storage entity is triggered to segment long information buffer into pieces, and store. Hence it is also triggered for re-

[6] This is because only compressed data can be meaningfully segmented, and a long message is always stored after segmenting and forming of a set of TPDUs.

assembling pieces, which are part of bigger buffer, before giving it back to the retriever entity.

4.5 Setup of context, for context-based information access and exchange

The center of actions here are the context managers. The set-up of context may be done for asynchronous or synchronous data transfer; depending on the properties of application involved in end-to-end information exchange. For a packet-access based context, the mode of information exchange does not matter.

At times, network may indicate it's wish to send in-band announcements or tones, in which case codec should be activated by the service coordinator.

For a remote context, which is being set up by an entity within an old network, the context manager should be able to derive the type of service required, and the attributes provided for the service context. It may then modify service type, service attributes, or both; and re-negotiate it with the network.

The context manager hence keeps a default attribute table, in case it needs to negotiate both service type, and service attributes, for a context being set up by network or the remote entity.

There may be a networking component, like a layer 2 entity or a device driver, which will influence certain attributes of an information exchange context. The examples include protocol configuration options, UIL2P, LLI, serial interface parameters such as number of start bits, stop bits etc. are attributes, which influence the interworking at the gateway level in packet or circuit domains. The reason for this influence is because such entities interwork with context managers, for the sake of information exchange. Data may be continuously transacted between these entities, and communication stack within the mobile system.

One of the attributes for such a context definition is the end-point identifier(s), commonly known as address. For routing purposes in network, or for directory search purposes within ADN tables, such identifiers are represented as a sequence of digits. The address can also be treated as a sequence of sub-sequences of digits; and each sub-sequence has a different semantics defined according to the network operations requirements.

Sub-addressing is an extra, optional capability, and may be used for addressing a group of network nodes having a common public address.

The methods available to manage these contexts are mainly of three natures: set-up or activation of context, attribute modification of an active context, and clearance or deactivation of a context. Implementations of these methods differ for different technologies, and different domains.

The context for information transfer may be a piece-wise active set of sub-contexts. In such a case, the linkage between these sub-contexts should be such that they are activated or deactivated together.

For speech calls only, application may use DTMF method for speedy setup. In this case, key depressions done at application level, are emulated and transported using signaling primitives, directly to the network.

For supplementary services only, password is one of the optional attributes, used with various methods of service operations.

4.6 Access and exchange of information for context-based transfer

In the local context, the data is transparently exchanged between the application(the prime generator and consumer of information), and the communication entity such as the protocol stack. The communication entity will then be involved with the network to actually transact the information in the global context.

In few specific cases, wherein a small piece of data needs to accompany a setup attempt of a context, invocation of USSD or UUS is used as a supplementary service by context manager. If a particular piece of signalling provided by application cannot be decoded by service coordinator, then it should treat this as USSD data. It shall then give this data to context manager, who will transfer it to the network.

4.7 Context-less information access and exchange

For store-and-forward nodes, the availability status of next possible point-to-point hop helps in making decisions about storing and forwarding operations.

It should be possible for message receiving entity, to be able to make decision of storing it, and then notifying it; or directly sending it to user via the application. There are cases, when instead of directly sending to user, the message is sent to SIM.

The message store[7] for the messages should know required methods of compression, decompression, segmentation, re-assembly, packing and unpacking of messages.

Various other operations can be *transported* along with the header, or meta-data part of messages, such as reply path, message waiting etc. For outgoing message, only transportation is required. For incoming messages, there are operations, to which the application should respond. The service capability provided here is <transportation of operations, across applications running at different nodes>.

Sorting filters should be available for the use by both the message store, and message forwarding entities. These filters operate on the meta-data of a set of messages(the set may be a singleton set), and they work according to the settings provided by the application. These filters can be time-based, access-based[8], and so on.

There should be context-less, time-invariant transformation methods available with the forwarding entity. These transformations could be application-specific, and transform the message data(which can be possibly email, animation, audio-clip etc.) to a representation domain, the syntax of which is known globally. There may be run-time transformations, such as character-set translations, applicable on the message(s) data.

[7] Could be SIM, NVM or ME
[8] Unread, read messages, etc.

4.8 Asynchronous notifications of various events

Notification, and subsequent action, is a manifestation of feedback mechanism. It is used by application to control the overall context and the mode of communication. This is done by applying changes in configuration and structure of information access. The action can be pre-configured by the user within the application, can be defined statically based on capabilities of the application/terminal, or can be done synchronously with the arousal of notification. We identify the following classes of notifications:

1. **Registration-related notifications.** This can provide notifications of event, which might lead to changes in registration of the application/terminal. For example, no network available, registration denied in the visited network, mobile terminal moving back into home network, removal of SIM etc. The typical action(adaptation) on such notifications is a try to change the network registration, and to block the information flow of various applications/sub-applications. For that reason, the registration controller should notify a character-based, or numeric identifier of the network to the application(based on identify provided by mobility management entity), whenever there is a change in network. The user should also be able to know the service availability in the registration area, in which it is registered. This can enable any application, with or without the mediation of user, to decide any changes required in registration. It can also allow an application to go into emergency calls only mode, in case of permanent loss of registration. Any interruption in registration leads to loss of services for application; and hence registration controller should notify any such interruption, and subsequent gain of registration, if any. Since the communication is based on radio, it should be possible to notify the perceived signal strength of the access frequencies in a cell. Based on such provision by radio resource entity, typically the user can, force a movement to another network, whenever there is a perceived degradation of service.

2. **Information exchange-context notifications.** For a context, which is being established by network, it should be possible to offer a notification to application, after the semantic checks are completed by context managers. The application may then decide, with or without the intervention of the user, to accept or reject the context establishment. Any supplementary services, which get activated/invoked during any phase of a circuit-context establishment(both mobile-originated, and mobile-terminated context establishment), and which the context managers can derive from information from connection management entities, should be notified. This can trigger many a decisions in application(directly because of user-provided trigger, or preset trigger).

3. **Message related notifications.** In case of storage of incoming message, it should be possible to notify to the application/terminal, of the arrival of a message. The user may then fetch that particular message, transfer the message etc. The application should be able to derive the transported operations, decoded and provided to application by the forwarding entity. Based on these transported operations, such as reply, message waiting etc., the application may take different types of actions. Finally, any overflow in message store(at the moment of storing yet another message) should be notified by the store, so that the application can try to offload some messages.

4. **Security related notifications.** This typically is based on ciphering indication. If the information exchange at radio level becomes unciphered, then the application should be notified by the radio resource entity. This can lead to decisions within application, such as rejecting network’s attempt to set up an information exchange context, or disabling application for setting up an information exchange context.

4.9 Generic access to stored information

The generic access can be due to user notifying it's wish to an application, for display of certain type of stored messages. With all the possible information about filtering and transformation, a request is made to forwarding entity. The forwarding entity then contacts message store, fetches the set of messages according to the filter specifications, does the transformation, and provides it back to the application.

The user may also decide to delete a message, fetch a message to post an asynchronous reply, etc. All these operations are routed via the forwarding entity.

At times, the user may revisit information such as the accumulated cost, attributes of a particular information exchange context, etc. All these read only accesses are performed by application, to the appropriate objects.

4.10 Interface Mechanisms

To do all the aforementioned operations, certain mechanisms for accessing the service capabilities are envisaged. Many operations are encapsulated into a specific format, such as a string, or command, and passed over to service domain. A parser parses and generates intermediate representation of the signaling data provided. For example, AT commands and MMI strings need to be parsed. A validator should be optionally supported, to check for semantic problems with the data provided. Once the operation to be done is derived, it should be possible for the service capabilities to be used to perform the operations. The result of the operation, irrespective of whether it was performed synchronously or asynchronously by service coordinator, and irrespective of whether it got performed remotely, say, by the network, should be formatted by a formatter. Such a response should then be passed back by the service coordinator, to the application.

The communication interface between application and service coordinator may behave differently at various times; and hence appropriate decisions are taken by service coordinator.

To provide compatibility to old-style terminals, a modem compatibility mode is provided for packet-switched context setup to be similar in syntax, as the interface for context setup for circuit-switched context.

4.11 Controlling by SIM

SIM is a physical module on a mobile device, which hosts many an embedded applications. At times, SIM uses the service capabilities built within the ME via an agent present on the ME side. For example, SIM can choose to do context-based or context-less information exchange, get data from user, via the application interacting with this agent, influence the application interfacing with the user(menu exporting) etc. At times, it tells the service coordinator to get the user's attempt of context setup, or message transmission to be verified by it. In such a case, the output of context manager, or message sender entity, is redirected to the agent.

4.12 System power down

In this process, an attempt is made by the mobility management entity, when triggered by an interrupt, to notify the network about the shutdown. This will help network to save unnecessary paging towards the mobile station. Some configuration can then be saved in a non-volatile memory, by the service coordinator.

5 Overview of Results

As can be seen from previous section, the details were made to use a lot of abstract terms. This made our job of identification of service capabilities easy. Further, by choosing to focus on describing everything that a mobile system does, we made sure that we did not miss out on identifying any *framework* service capabilities. From the end-to-end services' description given in standards, it is not straightforward to identify the framework service capabilities. The capabilities were further grouped into modules based on their interaction patterns. Further separation of concerns was done by grouping procedural and data-driven capabilities in separate modules. Based on their real-time requirements, we distributed them among various tasks. An event manager was added to the architecture to support the event loop. The modules were kept loosely coupled. This aided us while doing customer-specific architecture optimizations. We also defined a comprehensive API library for the middleware solution, to expose the features of the service capabilities.

Our results with customizing and integrating MobiTAAL in various handset solutions were encouraging. The complexity of incorporating new services in the framework, porting the architecture without much impact on the performance of the middleware did come down a lot. However, by postponing architectural optimization till the point of integration, there is a slight addition in the efforts required. Also, we found that at few places such as SMS storage management, there was over-generalization of the concept. We had to do significant code optimization later on in such (few) cases.

6 Conclusion

An attempt to design an extensible service framework for mobile handset solutions was presented in this paper. It is based on identifying instances of service capabilities as *embedded* in various *service functions* associated with a handheld mobile device. The goal was to model and design a single framework to cater to the services used in context of technologies related to each other in evolutionary ways. The technologies were the GSM evolution technologies, e.g. GSM, GPRS and UMTS as defined till Release 99 of the standardization body. To achieve so, it was necessary to describe the abstract service operations, from the user's perspective. With other mechanisms such as MExE and SIM Application Toolkit gaining popularity, the service environment is growing richer and richer. Our results have been good: and a future area of improvement is ameliorating necessity of few architectural changes. It can be said, hence, that the concept of using service capabilities to model software-based services of handheld device can be applied in commercial context, providing better results.

7 Acknowledgements

The authors are grateful to IIT Delhi, with whom part of the work was done under the purview of an industry-based master's thesis by a student.

8 References

[1] Michel Mouly, Marie-Bernadette Pautet. *The GSM System for Mobile Communications*. Telecom Publishing.

[2] L. Capra, W. Emmerich et al. "*Exploiting Reflection and Metadata to build Mobile Computing Middleware*". In Proc. of Workshop on Middleware for Mobile Computing, November 2001.

[3] Berger, M., M. Bouzid, et al. "An approach to agent-based service composition and its application to mobile business processes". In IEEE Transactions on Mobile Computing, September 2003.

[4] Hrishikesh Sharma. "*Ideas on Using Service Capabilities in Middleware Design*". Sasken Internal, May 2002.

[5] 3GPP Technical Report TR 22.944.
http://www.3gpp.org/ftp/Specs/archive/22_series/22.944/22944-500.zip

[6] 3GPP Technical Specification TS 23.060.
http://www.3gpp.org/ftp/Specs/archive/23_series/23.060/23060-400.zip

[7] 3GPP Technical Specification TS 22.101.
http://www.3gpp.org/ftp/Specs/archive/22_series/22.101/22101-400.zip

[8] 3GPP Technical Specification TS 22.105.
http://www.3gpp.org/ftp/Specs/archive/22_series/22.105/22105-400.zip