Technical Information

Open Service Access: Advantages and opportunities in service provisioning on 3G Mobile Networks

Proposal for Enhancements to the Parlay/OSA Specifications

Authors:

Michael Walkden (editor), Nick Edwards, David Foster (BT)

Milan Jankovic, Borislav Odadzic (Community of Yugoslav PTT)

Gaute Nygreen, Geir Gylterud (Telenor)

Corrado Moiso, Sergio M. Tognon (Telecom Italia Lab)

Brendan de Bruijn, Eric Prigent (Broadcom)

Abstract

This document was prepared by EURESCOM project P1110 and is intended as input to the Parlay Requirements process. The document consists of a number of specification enhancement definitions relating to the following areas:

- Modifications to the existing specifications
- Proposals for new framework/ service interfaces
- New specification features

The Project P1110 participants will pursue the development and solution of the issues defined in this report and wish to provide this document in order to help co-ordinate their actions along side those of the Parlay specification development teams so as to reduce overlap.

EDIN 0216-1110

Project P1110

For full publication

September 2001

EURESCOM PARTICIPANTS in Project P1110 are:

- British Telecommunications plc
- Community of Yugoslav PTT
- *eircom* plc
- Swisscom AG
- T-Nova Deutsche Telekom Innovationsgesellschaft mbH
- Telecom Italia Lab
- Telenor AS

[Project title] Open Service Access: Advantages and opportunities in service provisioning on 3G Mobile Networks

[Document title] Proposal for Enhancements to the Parlay/OSA Specifications

Editor: Michael Walkden, BT

Project leader: Oddvar Risnes, Telenor AS

Project supervisor: Uwe Herzog, EURESCOM GmbH

EURESCOM published project result; EDIN 0216-1110

Table of contents

1 Introduction

The purpose of this document is to provide a number of enhancement proposals to be discussed during the forthcoming Parlay requirements meetings. The proposals are presented as:

- Modifications to the existing specifications
- Proposals for new framework/ service interfaces
- New specification features

For each issue proposed in this document, a detailed definition is provided along with examples of scenarios in which the solution provides improvements to the API. The EURESCOM P1110 project team will work towards providing solutions to these issues and enhancements and it is the intention of the Project to present its results to the owning standards bodies of related API specifications such that this work is not duplicated and so that it can provide valuable input into the API specification improvement process.

The following issues are addressed in the following chapters (Chapters 2-8):

- "Balancing Up" of Interfaces
- Framework Information Model
- Framework Management Tool
- Support for an OLO Environment
- Contribution on Parlay "lite"
- Protocol APIs
- Data Hosting Service Interface for User Profile and Application Data

2 'Balancing Up' of Interfaces

- Category: Enhancement of Framework and Service Interfaces

2.1 Issue Definition

Many of the Parlay interfaces are highly asymmetric between application and gateway. As the capabilities of terminals connected to networks continues to grow, network based applications will require greater awareness of these capabilities. Likewise, as new network protocols such as SIP, which are peer-to-peer in nature, are developed, new types of functionality and control will become possible. The 'Balancing up' of interfaces is concerned with identifying areas where the asymmetry of Parlay may cause limitation in functionality or feature interaction problems.

Although many of these asymmetries are particularly apparent on SIP networks, many aspects identified will not be restricted to SIP.

The work is aimed at identifying and suggesting changes to Parlay to help in solving feature interaction problems and to support fully the flexibility which new networks, protocols and terminals enable.

2.1.1 Scenarios

This section provides scenarios and examples of situations where more balanced interfaces might be beneficial.

At present the work is expected to have a broad impact across the service interfaces, for example.

2.1.1.1 Call Control

- An application can create a new call leg within a call, but a network cannot create a call leg and notify an application that a call leg has been created. Terminals and protocols which allow a client to add an additional party to the call should be able to do so, making the application aware of the new party.
- An application can attach and detach call legs from a call, but the application cannot be notified that the call legs have been attached or detached by the connected party. This causes problems because if a caller detaches from a call, then the application is not aware of this. In a symmetrical model, the media stream could be connected or disconnected by either application or client, and the application could be notified of changes

2.1.1.2 Mobility interfaces

- The mobility interfaces allow an application to query the status and location of an address. However, an application cannot request notification of incoming requests for the status or location of address and return a response. A scenario where this causes a problem is where a unified communications application provides a 'one-number' service, and a second application requests the status or location of the user. The unified communications application must have be able to receive status requests and respond to them.

2.1.2 Requirements

- To identify aspects of Parlay where asymmetries can cause limitations. This includes call control, mobility, terminal capabilities and user interaction interfaces
- To collate the information from the comparisons and to identify service scenarios where the existing interfaces are too restrictive and where symmetrical behaviour would be advantageous
- To ensure that any symmetrical interfaces proposed do not compromise compatibility with existing asymmetric networks

2.1.3 Solution Elements

Modified Parlay Interface class diagrams, interface definitions and data types or recommendations via reports.

3 Framework Information Model

- Category: ***Framework Enhancements***

3.1 Issue Definition

An Information Model for the Parlay/OSA Framework (FIM, Framework Information Model) can answer to several needs, coming from different actors in the possible business models, but the first reason to define such a model is to have an agreed, common view and a "common language" to address the same concepts when analysis related to complex data structures, relationships and objects, have to be done. Since work done so far, both on standard and on the vendors' side, indicates that the Framework functionality is the heart of the Parlay Gateway system, the availability of a Framework Information Model appears more and more relevant.

The needs of new interfaces (e.g. User profile service interface), or to improve the existing ones, makes the FIM definition relevant also on the standard side. In particular, a clear definition of the objects belonging to the framework domain, and how they interact, can help and shorten the analysis/modeling phase of possible new Parlay services; consequently the definition of new interfaces and improvements/enhancements of existing ones can take advantage of a FIM in terms of speed, easiness, completeness etc.

Vendors planning to develop Parlay Gateways have surely to face the problem of defining data/information models of their system, including a model of the framework functionality, but such models will be obviously tied to the specific implementation. In addition, parts of a FIM are already, implicitly, defined in the specification of current framework APIs, such as service subscription, service registration, and service properties.

The Framework Information Model, objective of the present work, to be useful in different contexts, should be carefully structured and defined at a proper detail level: not so deep to impose implementation constraints and not so abstract to hide aspects (namely entities, relationships, objects or other) that are needed to achieve the aforesaid objectives.

3.1.1 Scenarios

This section describes two possible scenarios/contexts that could take advantage of the FIM.

The Service Registration is a good example of a scenario that could take advantage in using a Framework model. Before a service can be accessed and used, it has to be registered in the Framework, where information on the available services is contained. The registration procedure is described in the specifications, as well as some relationships among the involved entities (namely Service Supplier Administrator, Framework, the Service that has to be registered); moreover some of the involved objects can be deduced from the interface definition, but a clear, high level view of the whole procedure is not available. Part of this issue could be covered by FIM.

The second scenario a FIM can be useful, is related to Service Subscription (this subject is treated in paragraph 4.17 of "Framework Interfaces, Client Application View – Version 2.1" specification). Even though the Service Subscription phase is described by the specification in some detail (e.g. in terms of a "Subscription Business Model", defining high level actors and relationships, with pictures and text), it is not simple to analyse all the relationships among the various involved entities (e.g. the Enterprise Operator, the Framework and the Client Application, that have precise roles in the Service Subscription phase), or to have a clear vision, as an example, of all the SAG (Subscription Assignment Group) aspects. A more formal and detailed representation, including the involved Framework entities that can be defined in the Framework model, can be helpful.

3.1.2 Advantages

As it was mentioned before, various advantages can be obtained in defining a FIM:

- all entities involved in Parlay activities can benefit in having a shared, detailed and well defined vision of the framework structure;
- NOs and ASPs could refer to the FIM when framework related SLA aspects have to be addressed;
- application suppliers and NOs can have a clearer vision on the possibility to customise the subscriptions to service interfaces;
- areas that are not fully covered by Parlay/OSA specifications as well as new set of API can be identified and analysed starting from a Framework model;
- Vendors that plan to develop Parlay/OSA gateways or that need to add new functionality can refer to a known FIM.

3.1.3 Requirements

A useful Framework Information Model should be able to support analysis (as mentioned before both standard-oriented and implementation oriented) of several aspects. In particular the following functionalities should be addressed:

1. Service Subscription and subscription management: objects and relationships description
2. Service Interface and Framework Interface subscription by applications: each involved entity should be correctly defined, together with the configuration data related to application subscription. Possible aspects related SLA constraints to applications subscription should be addressed as well.
3. Services and service interfaces registration and configuration.
4. Service discovery
5. Usage data management (e.g.. logging data, Service Interface usage data)

Services and Service Interfaces configurations are typically done through properties. A property is composed of a name, a type and a value (data or policy). Moreover, each property can be tied to validation rules. The model should be able to cover these aspects (and consider their relationship with service properties).

A further need is that the model should be easily extensible, to allow Service Interfaces incremental introduction. Each Service Interface is characterized by set of properties. Some of such properties can be defined by standards, others can be proprietary, marking out a particular implementation.

3.1.4 Solution Elements

The FIM definition will consist of a formal UML description, in terms of set of Class Diagrams describing objects and relationships; a textual description of the objects and their attributes will be given as well.

In addition to the model, other outputs can be proposals of enhancements to existing APIs as well as new APIs. As an example, a suggestion of API to manipulate some of the identified Framework objects, if useful, can be done.

4 Framework Management Tool

- Category: ***Extension of framework functions***

4.1 Issue Definition

The Information Model[1] for Framework Functions should be accessible from some sort of management tool. To make this possible one should define the API to configure and access the data model.

Information from off-line Service Level Agreements and information needed for on-line Service Agreements should be entered via this API.

4.1.1 Scenarios

This section describes a scenario where the extension will make the work of the network operator easier.

When a third party wants to access a service a Service Level Agreement (SLA) between the third party and the network operator. This contract gives a detailed description of all aspects of the deal, such as the extent of the contract (which services should be accessible and the usage allowed), the responsibilities of the network operator and the third party, and actions to be taken if one of the parties does not keep their part of the deal. Many of the points in the SLA needs to be transferred to the Framework so that it can supervise that the contract is respected by the third party and also take action if the contract is not respected. With a Framework Management Tool API (together with a Framework Management Tool) this process would be easier and the possibility to transfer the data from one Framework to another Framework (i.e. if one wants to change vendor) would be there.

4.1.2 Advantages

The advantages lie in easier management of Parlay/OSA access:

- Quick and easy set-up of access from new third parties.
- Notification (or service denial) if a third party reaches its allowed use, can be sent to both the network operator and the third party.
- Easy change from one vendor to another, because the Management tool uses specific standardised interfaces.

4.1.3 Requirements

The Framework management tool API should provide access to the Framework Information Model that should contain these (and possibly other) aspects of the Service Level Agreement:

- The SCSs open to access by the third party (or part of SCSs functionality).
- Maximum traffic/usage of SCSs
- Actions to be taken when maximum traffic/usage is reached.
- Restrictions in the use of methods
- Restrictions in visibility of certain information (parameters) and restrictions in modification of certain parameters

The Framework management tool API must be designed to be able to access and make changes to data concerning the agreements between network operator and third party whilst the Framework is running.

[1] This is the Information Model for Framework Functions enhancement proposed by Telecom Italia Lab

The Framework management tool API must be designed to be independent of vendor and implementation language.

4.1.4 Solution Elements

The definition of the Framework management tool API should be based on the Framework Information Model.

5 Support for an OLO Environment

- Category: new Service Interface

5.1 Issue Definition

In today's regulatory environment, in which incumbent operators are increasingly being forced to open their networks to Other Licensed Operators (OLOs), the role an OSA Service Gateway could play in this context should be explored. For example, the OLO could be given access to a particular part of the incumbent's network using an OSA gateway to access this functionality. An OSA Gateway could be placed at the Point of Interconnect between the two networks for this purpose. There might be the necessity to enhance the OSA interfaces with a package specifically geared to OLO access to the network.

The initial requirement for the PNO would be to explore the necessity and feasibility of this approach. Secondly, a more detailed scenario should be drafted as to what functionality should be exposed on an OSA Point of Interconnect Gateway and how to capture these requirements in a set of interfaces.

The approach of giving an OLO access to the incumbent's network via an OSA Gateway will provide a transparent and manageable way of

1. Fulfilling the regulatory requirements of giving the OLO access to certain network capabilities.
2. Providing a cost-effective, manageable and state-of-the-art range of Interconnect services to the OLO with attached increases in revenue.

5.1.1 Scenarios

This section provides scenarios and examples of situation where the OSA approach to Interconnect service provisioning could be applied and might be beneficial.

5.1.1.1 Service Level Agreements

The OSA API, when suitably enhanced, could be used for the signing of online Service Level Agreements between the OLO and the incumbent operator. For example, when a customer requests the provisioning of Carrier Pre-Selection, the OLO could sign an SLA with the incumbent for provisioning this service using the relevant OSA APIs.

5.1.1.2 Provisioning of Interconnect Services

The OSA API could, after the signing of a service level agreement, be used by the OLO to set or change subscriber data in the incumbent's network nodes. For example, the OLO could set up a subscriber with Carrier Pre-selection by using the relevant interfaces to request the subscriber data to be changed within the subscriber's local exchange.

Alternatively, the OLO could change routing tables relevant to the routing of the calls to its networks using the API.

This approach would increase efficiency in provisioning services to OLOs.

5.1.2 Requirements

The OLO Service Interface should address the following requirements. The user (OLO) must be able to use the interface to:

- Sign SLAs with the incumbent operator regarding privileges it has on the network, such as access to bandwidth or databuild in network nodes.
- Sign SLAs with the incumbent operator regarding class of service for a subscriber.

- Query the status of the databuild in a network node, depending on whether an SLA exists for this privilege.
- Request the change the databuild in a network node, depending on whether an SLA exists for this privilege.
- Query the status of a subscriber, depending on whether an SLA exists for this subscriber.
- Request the change the status of a subscriber, depending on whether an SLA exists for this subscriber.

In addition, some elements of charging might have to be built into the interfaces.

5.1.3 Solution Elements

The OLO Service Interface could be implemented by working closely with the User Profile Enhancement, as this work also aims to address the accessibility of network data to external parties.

6 Contribution on Parlay 'lite'

- Category: ***General enhancement***

6.1 Issue Definition

The Parlay / OSA specifications are large and complex; with the consequence that developers will require a long time to learn and understand them before they can use them effectively. The specification design forces applications and gateways to be multithreaded, which adds to the complexity of programming and debugging.

Parlay also uses complex data types, this means that it cannot easily be used in certain computer languages such as Visual Basic. There needs to be a 'cut-down' ("lite") version of the specification. This new specification needs to be as broad as the existing Parlay specifications, i.e. all interfaces need to be considered and slimmed down as necessary.

6.1.1 Scenarios

A 'lite' version of the Parlay/OSA interfaces should be sufficient for the service providers to implement most services known today in an easy manner in addition to providing new functionality in an easy way as new interfaces are specified.

6.1.2 Advantages

The possible advantage for the Network operator is that the interfaces will be more widely accepted and used by service providers. Simpler interfaces also mean that the probability of erroneous implementations will be less.

The possible advantages for the service providers are that the interfaces will be easier to use and to understand.

6.1.3 Requirements

All interfaces need to be considered to see if a lightweight version could be produced. The data types used by the standards are also too complex and could be simplified in a lightweight version. For example, the current address data type currently has six data members. These could be reduced to one. There are loads more examples like this. Ideally, there should be no complex data types would simplify the interface interactions.

6.1.4 Solution Elements

A thorough examination of all interfaces has to be performed to make a selection of:

- Most relevant methods
- Most relevant parameters

This will be used as a basis for defining the enhancements that has to be done to be able to provide a 'lite' version of the interfaces.

An evaluation has to be made on how these 'lite' interfaces are to be offered.

7 Protocol APIs

- Category: ***General***

7.1 Issue Definition

This enhancement is related to the problem that Parlay's generic nature makes the relationship with other technologies such as Session Initiation Protocol (SIP) difficult to define. The issue arises as to how well the functionality of new protocols, SIP in particular can be mapped onto the current Parlay interfaces. This enhancement suggestion aims to explore the mapping from SIP up to Parlay, so that possible weaknesses in the Parlay interfaces may be identified and improvements suggested to the Parlay group.

7.1.1 Scenarios

The Parlay Call Control specification has been created with the goal of defining a technology independent call model allowing call routing and management. The Call Control model is based on the traditional IN model.

At the same time, Internet experts have designed the Session Initiation Protocol (SIP), which can also enable applications with call control on a communications network. SIP has been created as an application-layer protocol for creating, modifying and terminating calls. This functionality can be exploited by a Parlay Gateway.

The key idea behind Parlay is that network specific information is hidden from Application Developers. However, the Network Operator would also like to expose the full functionality of a SIP-enabled network on the API. The proposal therefore entails the determination of set of SIP-specific services, which shall include SIP mobility aspects, and mapping these onto Parlay Interfaces. These mappings will serve to highlight areas where the interfaces may be improved so that the full functionality of SIP enabled networks can be exposed to the Application Developer.

Figure 1 shows how a Parlay gateway will fit into a SIP network. On the IP network side, the Gateway will be seen as performing the functions of a SIP Proxy and User Agent (UA). Internally, in the gateway, the SIP messages are mapped onto Parlay using the protocol API. Note that for each type of network, such as PSTN, UMTS and IP Multimedia, a Protocol API has to be provided to map the network specific functions to generic Parlay functions.

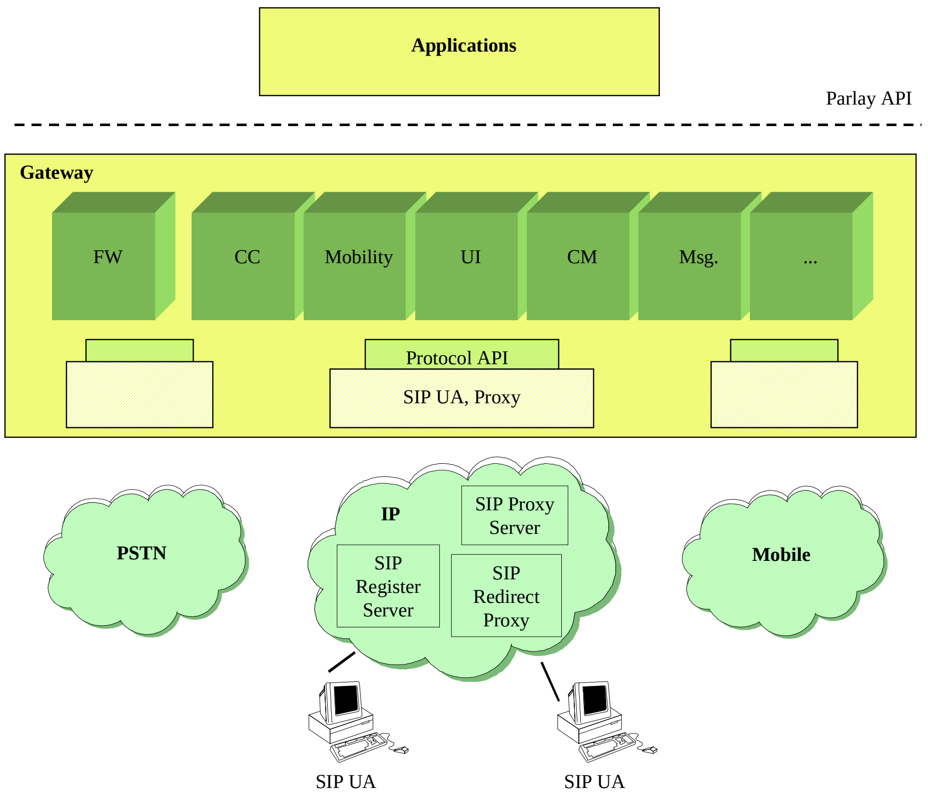

Figure 1 Diagram of a SIP Enabled Parlay Gateway

7.1.2 Requirements

1. Determining SIP specific functionality and services.
2. Mapping this functionality onto the Parlay API.
3. Identifying weaknesses in the mapping and suggesting enhancements to the Parlay group.

Identifying the role of related technologies, in particular the JAIN-SIP API.

8 Data Hosting Service Interface for User Profile and Application Data

- Category: ***New service interface***

8.1 Issue Definition

This enhancement is related to the definition of a new service interface to manage the hosting/housing of data defined and used by Parlay client applications.

Parlay/OSA APIs could be seen as a way to "sell" network capabilities and functions traditionally under the control of a network operator to external applications.

Examples are:

- Generic Call Control APIs could be seen as a way to offer SSF functions;
- User Interaction APIs could be seen as a way to offer Special Resource Functions related to IVRs, advanced IVRs, etc.;
- Messaging APIs could be seen as a way to offer Special Resource Functions related to e-mail, voice mail, unified messaging, etc.

Through these APIs an external application provider can develop services that integrate different network capabilities managed by the network operator, without needing to deploy its own network equipment.

The SDF (Service Data Function) could be an additional capability that could be offered through Parlay APIs. SDF in the traditional IN architecture is used to access (in read and in write mode) data related to an IN service. In general SDF are implemented by systems that fulfil requirements concerning high-availability, reliability, fault-tolerance and real-time.

A Parlay API could be defined in order to allow:

- third parties to access, in a secure, controlled and, possibly, accountable way, SDF-like functions;
- network/gateway providers to house third party data.

Such an API could be used by an external application (or a group of applications managed by the same service provider) to store some critical data (e.g., configuration data, user profile data, etc.); in this way, the applications can reuse the reliable systems of a network operator without the need to deploy dedicated DBs or directories.

The data stored and retrieved though such an API are specific of a single application (or of a group of applications managed by the same service provider). Examples are:

- data related to a user/subscriber of the application: subscription data, configuration data, personalisation data, history/profiling data, logging data, etc.
- Application specific data: calendar data, instant messaging data, network call center data, etc.
- Data related to the configuration of the applications.

Therefore, each application/application provider has to define its own data model, according to the requirements of the application.

According to this scenario, an API to support "Application User Profile Housing" should fulfil the following requirements:

- it is application independent;
- support to write/modify, read, create and delete application data stored in network operator systems;

- secure and controlled access: the application can only access the storage space allocated to them and no other applications can access the space allocated to another application;
- Support to (simple) query/search on the stored data.

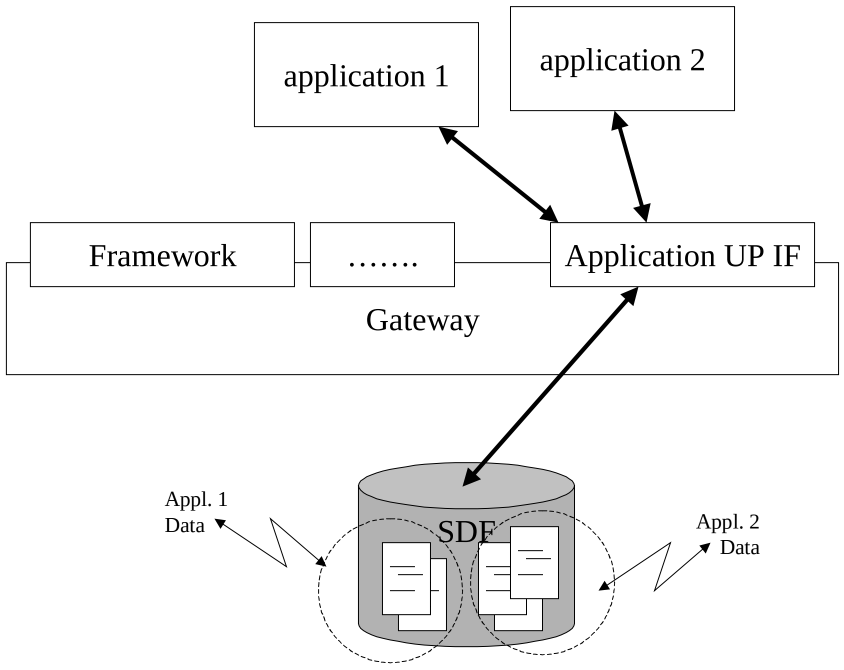

Figure 2: Interface for Data Housing

Extension: User Profile for specific business cases

The User Profile Service Interface could be used in addition to access data, which are horizontal to specific applications and service providers, defined in order to address specific business cases (e.g., VHE, personalized networks, etc.). In this case, the User Profile Service Interface must be coupled with a data model specific to the particular application scenarios to be addressed.

Some of the accessible data could, in addition, be under the control of the Network Operator (e.g., Nicknames, Password, List of Subscribed Services, etc.) and modified under its control (e.g., through/by its systems). The Network Operator could "certify" the correct usage of these data (by subscribers/users and service providers).

A first example of such data model is the one defined by the PAM forum and underlying the Parlay Presence&AvailabiltyManagement API, which is mainly addressing presence status and routing preferences.

Another example is the data model the CC/PP (Composite Capability/Preference Profile) defined by W3C: it is a general, yet extensible framework for describing user preferences and device capabilities to be applied to access/search the World Wide Web.

Additional examples are data models to support the business model based on Virtual Home Environment concepts, such as those under definition in 3GPP for the mobile network scenario; examples are the data that form the profile of a VHE subscriber (e.g., the list of subscribed applications, the list of used terminals, the account preferences, the customisation data, nicknames, password, etc.).

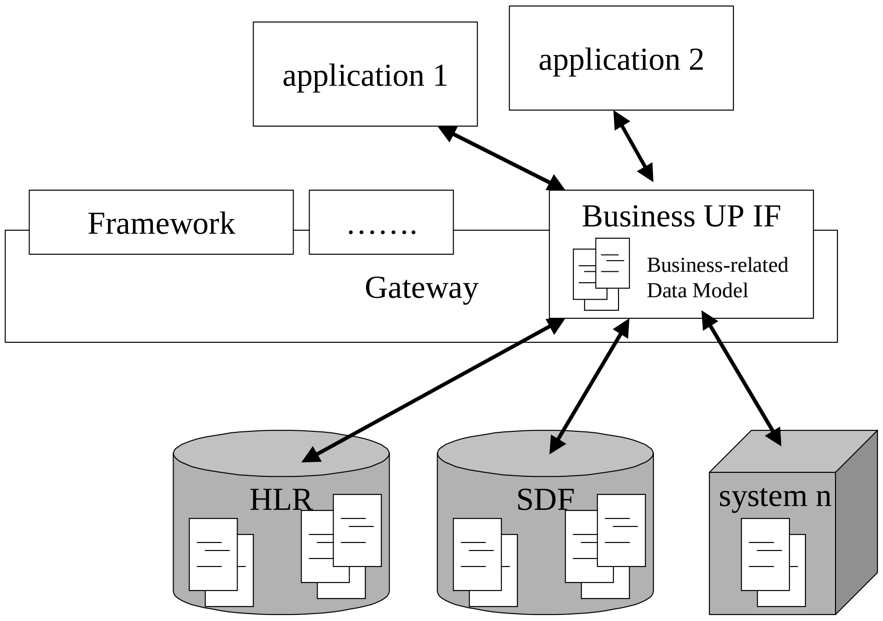

Figure 3: Interface for Business Case-specific User Profile

8.1.1 Scenarios

This section describes two possible service scenarios that could take advantage of the Data Housing Service Interface.

The first one is an application of Network Call Center. In addition to the usage of the relevant service interface (e.g., call control, user interaction), the application could use the User Profile Service Interface to store the information concerning the agents, the rules to select them, the profiling of the customers, etc.

The second service scenario is related to the application of the User Profile Service Interface to a specific business case. The Network Operator offers to the Service Providers some information on its Subscribers, by keeping them anonymous:

- the subscribers are identified by nicknames (the others service interface should be enhanced in order to map CLI, IP addresses, etc. to Nicknames);
- user profiles, associated to the nicknames, contains information such as the list of user preferences, etc.

Let assume that a user is interested in "science fiction" movies, books, etc.

When he accesses an application that provides information on cinemas, the application could access the data stored in the user profile associated to his Nickname in order to identify which are his preferences and return first the related movies (science fiction movies in this case). The same information could be accessed from applications (of other service providers) that deal with books, videogames, etc.

The user can access his user profile in order to change the stored information. In case of secure information (e.g., a password), the network operator can provide applications to do this in a trusted way.

8.1.2 Advantages

The possible advantage for the Network operator is to enrich the set of offered service interfaces, by allowing a richer support to the service providers.

The possible advantages for the service providers is to have access to a data storage function, that fulfils requirements concerning high-availability, reliability, fault-tolerance and real-time, without the need to deploy its own DB/directory systems.

The definition of data models specific to some application scenarios could support the development of new business models and the co-operation of multiple actors and roles.

8.1.3 Requirements

The User Profile Service Interface must address the following requirements:

1. *application independent*: it must be used by any application and must not refer to specific application scenarios;
2. *data model definition*: an application must be able to define a data model (e.g., by instantiating a "meta-model") in order to allow to store its data according to the preferred data schema;
3. *data handling operations:* support the following operations to handle the application data:
    - create new instances of the entities defined in the data model;
    - delete instances of the entities defined in the data model;
    - read the values of the instances of the entities defined in the data model;
    - store/write/modify the values of the instances of the entities defined in the data model;
    - search instances of the entities defined in the data model, matching simple queries (e.g., based on the entities values);
4. *data model changes*: an application must be able to modify the data model; the definition of the changes must also consider the rules for migrating the data values from the current to the new data model;
5. *data migration*: an application must be able to migrate the data from the current version to the new version of a data model;
6. *application sharing* (i.e., secure and controlled access): the application can only access the its data and no other applications can access the data of another application;

Some optional requirements are:

7. *handling of notifications of changes in data* (in case of group of applications owned by the same service provider): an application must be able to subscribe to be notified when specific data are modified (e.g., by another application) and the new values match some given conditions;
8. *Cryptography*: the data could be stored in a coded way, in order to improve security.

In the case of the usage of User Profile Service Interface for a specific business case, additional requirements are:

9. *specialized operation*: some generic operation to access data could be specialised for the access of some critical value (e.g., get_Nickname);
10. *handling of notifications of changes in data*: this requirement becomes mandatory;
11. *data model definition and modification*: these operations must be reserved to the Network Operator; versions must guarantee backward compatibility;
12. *Restriction: an application could have restrictions/constraints on the operations it can invoke and on the data it can access.*

8.1.4 Solution Elements

The interface for "application user profile", to provide data housing capabilities to applications, could be defined by starting from the definition of existing protocols to access directory records, such as LDAP, or XML-based Repository.

The interface could be separated into the following parts (some of them optional):

- Functions to manage the data model (definition, modification, data migration, etc.);
- Functions to handle the stored data (creation, deletion, read/write access, search, etc.);
- Functions to handle events (e.g., subscription, notification, etc.).

9 Conclusions

The project team intends to work on the development of a preliminary solution to these issues up until the end of 2001. All issues shall be considered within the scope of the EURESCOM project with the exception of the OLO Environment issue, which is provided as input to the Parlay/OSA requirements exercise.

[End of document]