ETSI TS 129 198-14 V5.2.0 (2003-06)

Technical Specification

Universal Mobile Telecommunications System (UMTS);
Open Service Access (OSA)
Application Programming Interface (API);
Part 14: Presence and Availability Management (PAM)
(3GPP TS 29.198-14 version 5.2.0 Release 5)

Reference

RTS/TSGN-0529198-14v520

Keywords

UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.org

Copyright Notification

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp .

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

The present document is part 14 of a multi-part TS covering the 3rd Generation Partnership Project: Technical Specification Group Core Network; Open Service Access (OSA); Application Programming Interface (API), as identified below. The **API specification** (3GPP TS 29.198) is structured in the following Parts:

Part 1:	Overview	
Part 2:	Common Data Definitions	
Part 3:	Framework	
Part 4:	Call Control SCF	
Part 5:	User Interaction SCF	
Part 6:	Mobility SCF	
Part 7:	Terminal Capabilities SCF	
Part 8:	Data Session Control SCF	
Part 9:	Generic Messaging SCF	(not part of 3GPP Release 5)
Part 10:	Connectivity Manager SCF	(not part of 3GPP Release 5)
Part 11:	Account Management SCF	
Part 12:	Charging SCF	
Part 13 :	Policy Management SCF	(new in 3GPP Release 5)
Part 14 :	**Presence and Availability Management SCF**	(new in 3GPP Release 5)

The **Mapping specification of the OSA APIs and network protocols** (3GPP TR 29.998) is also structured as above. A mapping to network protocols is however not applicable for all Parts, but the numbering of Parts is kept.
Also in case a Part is not supported in a Release, the numbering of the parts is maintained.

OSA API specifications 29.198-family					OSA API Mapping - 29.998-family	
29.198-01	Overview				29.998-01	Overview
29.198-02	Common Data Definitions				*29.998-02*	*Not Applicable*
29.198-03	Framework				*29.998-03*	*Not Applicable*
Call Control (CC) SCF	29.198-04-1 Common CC data definitions	29.198-04-2 Generic CC SCF	29.198-04-3 Multi-Party CC SCF	29.198-04-4 Multi-media CC SCF	29.998-04-1	Generic Call Control – CAP mapping
					29.998-04-2	*Generic Call Control – INAP mapping*
					29.998-04-3	*Generic Call Control – Megaco mapping*
					29.998-04-4	Multiparty Call Control –ISC mapping
29.198-05	User Interaction SCF				29.998-05-1	User Interaction – CAP mapping
					29.998-05-2	*User Interaction – INAP mapping*
					29.998-05-3	*User Interaction – Megaco mapping*
					29.998-05-4	User Interaction – SMS mapping
29.198-06	Mobility SCF				29.998-06	User Status and User Location – MAP mapping
29.198-07	Terminal Capabilities SCF				*29.998-07*	*Not Applicable*
29.198-08	Data Session Control SCF				29.998-08	Data Session Control – CAP mapping
29.198-09	*Generic Messaging SCF*				*29.998-09*	*Not Applicable*
29.198-10	*Connectivity Manager SCF*				*29.998-10*	*Not Applicable*
29.198-11	Account Management SCF				*29.998-11*	*Not Applicable*
29.198-12	Charging SCF				*29.998-12*	*Not Applicable*
29.198-13	Policy Management SCF				*29.998-13*	*Not Applicable*
29.198-14	**Presence & Availability Management SCF**				*29.998-14*	*Not Applicable*

1 Scope

The present document is part 14 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

The OSA specifications define an architecture that enables application developers to make use of network functionality through an open standardised interface, i.e. the OSA APIs. The concepts and the functional architecture for the OSA are contained in 3GPP TS 23.127 [3]. The requirements for OSA are contained in 3GPP TS 22.127 [2].

The present document specifies the Presence and Availability Management Service Capability Feature (SCF) aspects of the interface. All aspects of the Presence and Availability Management SCF are defined here, these being:

- Sequence Diagrams
- Class Diagrams
- Interface specification plus detailed method descriptions
- State Transition diagrams
- Data Definitions
- IDL Description of the interfaces

The process by which this task is accomplished is through the use of object modelling techniques described by the Unified Modelling Language (UML).

This specification has been defined jointly between 3GPP TSG CN WG5, ETSI SPAN 12 and the Parlay Consortium, in co-operation with a number of JAIN™ Community member companies.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TS 29.198-1: "Open Service Access; Application Programming Interface; Part 1: Overview".

[2] 3GPP TS 22.127: "Stage 1 Service Requirement for the Open Service Access (OSA) (Release 5)".

[3] 3GPP TS 23.127: "Virtual Home Environment (Release 5)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 29.198-1 [1] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TS 29.198-1 [1] apply.

4 Presence and Availability Management SCF

4.1 Introduction

The goal of these interfaces is to establish a standard for maintaining, retrieving and publishing information about

- Presence and Availability of entities for various forms of communication and the contexts in which they are available.

4.2 Motivation

Consider the following simple but desirable scenario for a communication service: An end-user wishes to receive instant messages from her management at any time on her mobile phone, from co-workers only on her desktop computer, and in certain cases for the messages to be forwarded to e-mail or even a fax machine/printer. The senders may know her availability for various forms of communication in the way she chooses to reveal it or alternatively the senders may never know how she will be receiving their messages. This scenario spans over multiple services and protocols and can only be solved currently by a proprietary solution that maintains the required information in an ad-hoc fashion within the application.

PAM is not a replacement for the protocols being standardized for various communication and network services. PAM attempts to standardize the management and sharing of presence and availability information across multiple services and networks.

The PAM specification is motivated by the observations that

- The notions of Identity, Presence and Availability are common to but independent of the various communication technologies, protocols and applications that provide services using these technologies.

- Presence does not necessarily imply availability. End-users or organizations require greater control over making themselves available through various communication devices.

- Presence based services need to address privacy concerns on who can access presence information and under what conditions.

Management of availability will span over multiple communication services and service providers.

4.3 Goals

The purpose of this document is to adopt the first release of a Presence and Availability Management interface specification created by an industry consortium, PAM forum, established for this purpose harmonized with the IETF model for presence (RFC 2778). This specification is also consistent with the ongoing work in 3GPP for defining the requirements and architecture for a standard presence service in the network.

With a desired goal of rapid acceptance and usage, the specification has been deliberately designed to be as simple as possible with an attempt to include a minimal set of functionality that is sufficient for use in non-trivial applications. Often, this has been at the cost of some useful features, which would have made the specification baroque and cumbersome if not controversial.

4.4 Concepts

This chapter briefly describes the various concepts involved in this specification to serve as the context for the rest of the document.

4.4.1 Identity

Identity, for purposes of the PAM specification, is a limited electronic representation of an entity (i.e., an individual or an organization) that participates in PAM-enabled applications and services. This concept corresponds to the concept of Presentity as described in the IETF Common Presence and Instant Messaging Model (RFC 2778).

The main characteristic of an entity that is central to PAM specifications is the name (or handle) by which entities are identified by applications and services. Entities may have multiple names, login ids, account names, etc., by which they are identified. As PAM attempts to abstract over multiple networks and services, it does not assume that a single name will necessarily identify entities across all application domains.

The generalized structure available in 3GPP for user names that may contain various formats for addressing has been adopted for these specifications.

For flexibility and extensibility, attribute lists are used to associate additional data with identities. Identities are typed to provide a way to manage such attribute lists. An identity type may be associated with a specific set of attributes and all identities of that type inherit instances of such attributes.

For consistency with IETF (RFC 2778) defined presence data models, PAM pre-defines an identity type Presentity with a list of presence attributes as defined in TS 22.141 based on the definitions in RFC 2778.

PAM implementations may map certain existing directory and database data to one or more types to allow access via PAM interfaces. PAM specifications do not specify how the data within the profiles are to be stored. They may be stored within the PAM implementation or mapped to data stored on external directories and databases.

4.4.2 Presence

The concept of presence has been used in several application areas, being most explicit in Instant Messaging. Starting from a simple notion of online/offline status, it has expanded to include other context information around the status such as disposition (out to lunch, away from the computer, etc.) and activity status (on the phone, idle, etc.). Location information, on the other hand, has largely been kept separate from what has been traditionally considered presence information. PAM specifications broaden the concepts of presence recognizing that all such information, including location, describes different contexts of an entity's existence. The unifying property is that the presence information is continually changing and that there is value in knowing the current information at different points in time for services and applications.

For the purposes of PAM specifications, presence is an extensible set of characteristics that captures the dynamic context in which an identity or an agent exists at any point in time. In contrast to the relatively static information about identities or agents (e.g., names, addresses, capabilities), presence refers to dynamic information such as location, status, disposition, etc. Registrations of presence and location information in existing applications are covered by this definition.

Presence information is differentiated from the more static information associated with identities and agents that are stored in attributes. The rationalization for this design is that the presence information is dynamic and has implications on the implementation. Some of the presence information is too dynamic to be maintained in static data stores such as directories and without this hint about the data characteristics, PAM implementers may make sub-optimal decisions on the way the data is stored. Second, presence information typically has expiration data that needs to be understood by the implementation.

The PAM specification recognizes that devices that provide presence information are not necessarily devices that communicate.

The PAM specification does not specify the methods by which the presence information is derived. For example, an instant messaging client on a desktop computer can register its status based on when a user is logged in. A mobile phone may do an explicit registration on a WAP server for instant messaging. The phone's presence for voice calls, on the other hand, may be inferred implicitly by querying the cellular network for the device being on when requested. The presence of an identity, on the other hand, may be computed using presence information from one or more devices owned by the identity.

Finally, the PAM specification does not require that the presence information be stored explicitly (i.e., in a materialized fashion) in a PAM implementation. An implementation may infer the presence information on demand from the underlying services or networks.

For compatibility with the presence model from IETF (RFC 2778), a type called Presentity is pre-defined with the attributes consistent with the IETF Presence model.

4.4.3 Availability

Availability is a property of an identity denoting its ability and willingness to share information about itself or to communicate with another identity based on factors such as the type of communication requested, the identity of the calling entity and the preferences and policies that are associated with the recipient. This is the primary means by which the current PAM specification enables controls for privacy. While presence is, in most applications, a necessity for availability, presence does not necessarily imply availability to all.

Availability is always with respect to a context. A context in PAM specifications is a set of attributes defining the state in which the availability is requested. For example, the query “Is Jane available for IM for Rob?” identifies the type of communication and the identity of the asker as the context. PAM allows for availability to be differentiated based on any attribute of a context. A context, “Communication” is pre-defined in PAM.

Most queries for presence in existing applications can be mapped into PAM availability queries to control the information being given out. Alternatively, queries can be mapped directly into PAM presence queries in situations where privacy controls and policies are not required or all presence data is open to the entity querying. This allows PAM specifications to be consistent with existing presence servers and to serve as the basis for presence services across multiple protocols while providing uniform and flexible privacy controls.

PAM specification does not specify whether the availability is computed on demand or stored explicitly. In some applications, the availability may be pre-computed and stored explicitly while in some, it may be computed at each request for availability.

While the PAM specification provides a mechanism to associate preferences with an Identity to control availability, it neither specifies the syntax and semantics of the preferences nor the process by which the availability is computed. These aspects are left to the implementation.

For example, a particular implementation may provide the facility to store preferences as rules such as “I prefer to receive my instant messages on my computer rather than my cell phone unless the message is from my boss or the computer is off, etc.”.

As an example, a computation of availability for communication may consist of the following algorithm:

1) Find all devices of the identity being called that are capable of the specified form of communication AND have registered their presence status as available.

2) Evaluate the rules associated with the identity being called to select the preferred device(s) from the set of present devices determined in Step 1.

3) If there are any devices available satisfying Step 2, indicate the availability of the identity being called via the available devices.

An implementation can chose to provide one or more means to specify preferences. It is expected that if there is industry standardization on the specification of preferences, the implementations will support such a standard. This is currently outside the scope of PAM.

4.4.4 Events

Events are representations of certain identified occurrences related to the concepts described above. The PAM specification provides for registering interest (i.e., callbacks) in being notified of such occurrences. Any entity that subscribes to the Event is a “watcher” in the IETF terminology (RFC 2778). An implementation is expected to provide such notifications.

Examples of events include,

- Change in the presence information of an identity
- Change in availability of an identity for a particular form of communication

PAM specifications contain a set of pre-defined events. Each event is defined by a name of the event, a set of input attribute value pairs that must be provided when an event is registered for and a set of attribute value pairs that are included in the notifications sent out when the event of interest occurs.

4.5 Scope of PAM information

Presence and Availability Management has the following types of information in its scope:

- Presence information, which consists of an identity's dynamic characteristics such as status and geographical location.
- Availability information, which consists of preferences associated with identities and computation of availability, based on the devices present and the current preferences.
- Notification of changes to the above pieces of information.
- Security issues for access to this information.

The PAM specification consists of interfaces to manage or access the above information.

The specification purposefully does not include

- Storage design or storage requirements for any of the presence and availability information.

These are to be decided by specific implementations of the PAM specification.

4.6 Security and privacy

As the Presence and Availability Management interface is designed to share information across administrative domains and to facilitate availability computation based on the identity of the entity desiring communication, security and privacy issues are addressed in the design. Two of the issues considered to be within the scope of PAM are:

- Access control to an implementation of the PAM specification.
- Use of an authenticated entity's credentials by methods in the specification.

To understand the distinction between the first two issues, consider, for example, an end-user that logs on to an Instant Messaging client and wishes to send a message. The client (or a gateway to which the client talks to) may access a PAM implementation to determine the availability of the destination for the message. The client (or the gateway) will need to be authorized for access to the PAM implementation independent of the user that logs in. A gateway may, in fact, do this access on behalf of a number of clients and, for performance reasons, wish to authenticate itself just once on start up rather than at each invocation. This authentication is handled by the authentication mechanisms in the OSA Framework common to all services within OSA.

Second, each invocation of a particular method will need to contain the credentials of the end-user that logged into the client so that the computation of the availability can take that into account when necessary for privacy issues.

It should be noted that the PAM specification allows for the possibility that the authentication of the end-users is not necessarily done within the PAM implementation itself. As long as the authenticated credentials supplied by the client (or gateway) are acceptable for validation and the client (or the gateway) itself is authenticated by the implementation, the authentication of end-users can occur anywhere outside the PAM implementation. A deployment scenario for a particular application is that one or more authentication services are provided as external services over PAM implementations.

This design does not preclude the possibility that the client (or the gateway) cannot be authenticated. Therefore, the credentials supplied by the client (or the gateway) may be held to stronger authentication criterion than credentials supplied by a trusted client (or gateway).

Finally, the PAM specification does not mandate the use of authentication within an implementation if the environment in which it is used does not require it.

Clause 5.1 explains the mechanism for providing data about the asker to each of the methods with a sequence diagram.

Privacy issues are addressed primarily by providing a mechanism to control the information flowing out of a PAM implementation based on whatever criterion the end user may choose to specify in the availability preferences and independent of any particular application.

The following security issues were considered to be outside the scope of PAM:

- Authentication of the identity of the end-users or entities. As explained above, this authentication may be provided by a third-party authentication service or it may occur through an authentication service written over the PAM platform. The only requirement is that the type of credentials supplied by the authentication service be acceptable to the PAM platform implementation being accessed.

Encryption of the flow of information between a PAM platform implementation and clients of this implementation. This is dependent on the method of access to the interface which is outside the scope of the PAM specification and hence to be determined by the implementation.

5 Sequence Diagrams

Most of the methods in the PAM interfaces are independently used to query or update presence and availability related information with no transactions or state transitions involved. There are two use cases for which sequence diagrams are useful

- Acquiring and using authentication tokens

- Registering for PAM events and getting notifications on the occurrence of the event.

The sequence diagrams for these two cases are provided below. It is assumed that the authentication with the OSA framework has already occurred and the application has access to the PAM interfaces.

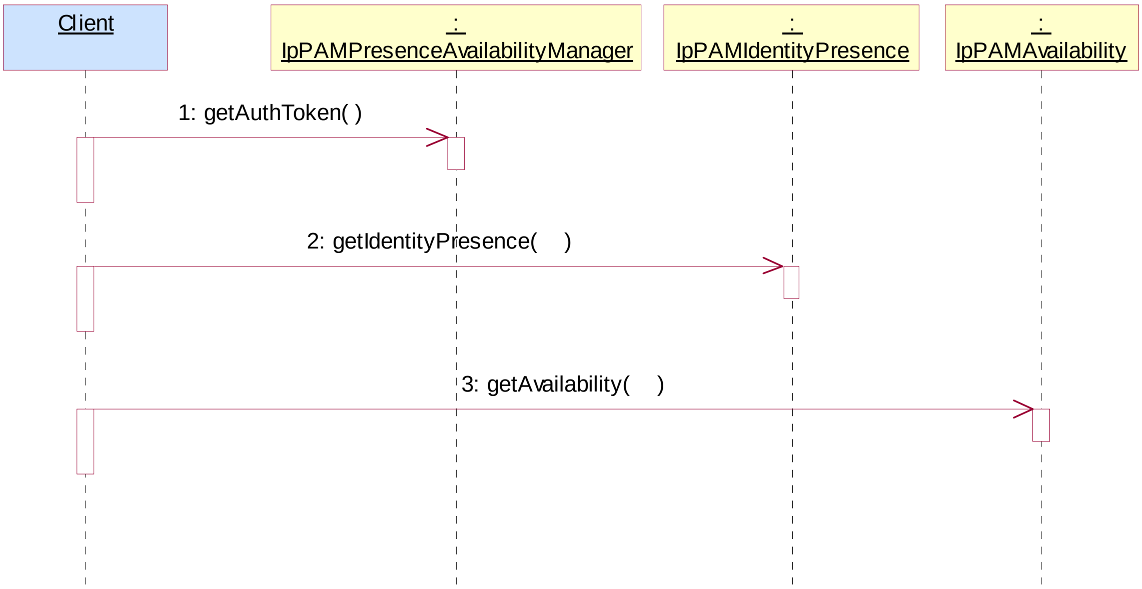
5.1 Use of authentication tokens

As an OSA Service, PAM uses the authentication features of the OSA Framework to provide access control to the PAM interfaces. In addition, PAM provides an optional mechanism for service/application level identification and authentication of the entity requesting the operation or alternatively on whose behalf the operation is being requested. To handle privacy requirements, the results of presence and availability data updates or queries are dependent on the entity requesting the operations.

In the simplest case, the entity authenticating to the OSA Framework to get access to the service interface is the entity requesting the operation. In general, however, a proxy or an application (such as a messaging server or a conferencing server) may authenticate with the OSA Framework once and then check for presence and availability on behalf of multiple client applications (such as instant messaging clients). The credential of these client applications if and when needed by the PAM service can then be provided via the credential parameter in each of the interface methods.

The mechanism to provide the asker data is via the optional parameter of type TpPAMCredential in each of the methods. Supplying the entire asker data in each of the methods is expensive for an implementation since it will need to parse and validate the data supplied in the asker data structure each time. An application may be accessing multiple methods for itself or for the benefit of end user(s) and will need to supply the relevant asker data in each case. To make the consideration of asker data more efficient, the application uses the getAuthToken() method in each of the managers in the SCFs once for each session per asker and gets a credential that can be reused as many times as necessary in the same session to represent the same asker.

The sequence diagram for an example usage is given below.



1: For any unique entity requesting the operation, the authenticated client of the OSA PAM service, requests for an authentication token using the getAuthToken() method in the PresenceAvailability Manager interface.

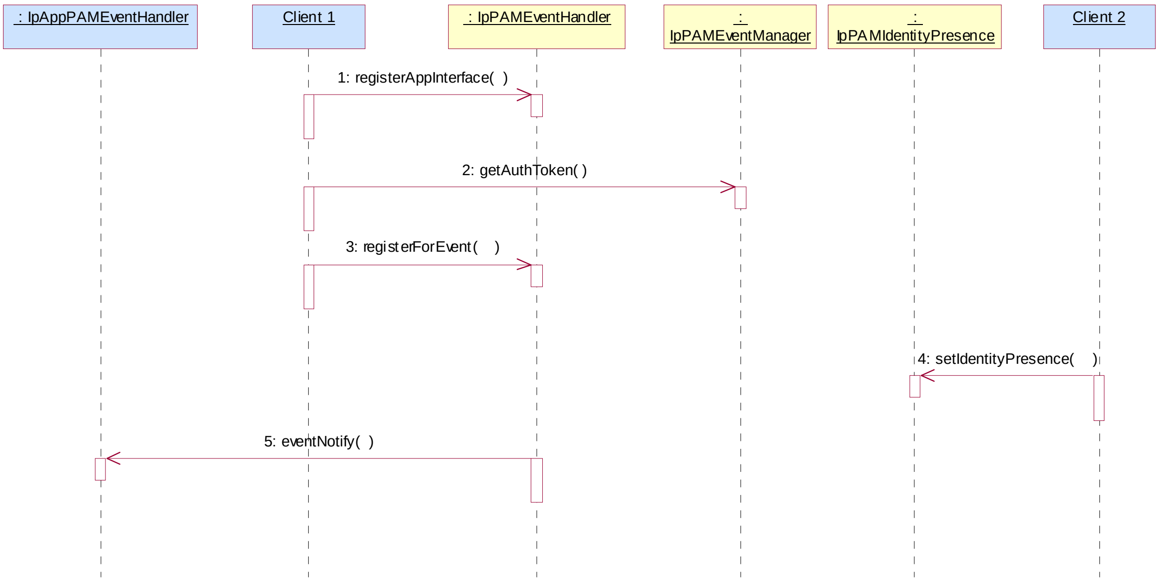
2: The token returned by the getAuthToken() method is used as the credential parameter of the getIdentityPresence() request.

3: The same token is used as the credential parameter of the getAvailability() request(). An authorization token can be used multiple times within the same session established with OSA framework.

5.2 Event registration and notification

An OSA client can register for certain events in the PAM service either for itself or on behalf of its own clients. The client will register one or more application interfaces with the event management service and then activate one or more events for each such registered interface.

The sequence diagram for an example is given below.



1: The client uses the registerAppInterface() method to register its notification interface. The getAuthToken() can be null since the client is doing this registration on its own behalf. The client gets a unique client ID back.

2: The client uses the getAuthToken() to get authentication credentials for its own application client on behalf of whom an event registration is required.

3: The client uses the registerForEvent() method to register for a change in availability event on behalf of its own client. The client gets a unique event ID back.

4: The presence information for an identity of interest in 3 above is changed by another client application acting on its behalf using the setIdentityPresence() method.

5: When the change in presence results in a change in the availability of the identity for the client that has registered for the availability change, a notification is sent out using the previously registered application interface.

6 Class Diagrams

PAM consists of the following SCFs:

- PAM AccessService consisting of interfaces to view and update presence and availability information and
- PAM EventManagement Service consisting of interfaces to subscribe to events in PAM and be notified of such events.

6.1 PAM Access SCF Class Diagrams

The PAM Access service consists of two packages, one for the application interfaces and one for the service interfaces. The application PAM Access package consists of 0 or more instances of the IpAppPAMPreferenceCheck interface and the PAM event management service package consists of a single instance of the following interfaces obtainable by applications using the service interface IpPAMPresenceAvailabilityManager.

IpPAMAgentPresence

- The purpose of this interface is to maintain the dynamic presence information of agents.

IpPAMIdentityPresence

- The purpose of this interface is to maintain the dynamic presence information of identities.

IpPAMAvailabilityManagement

- The purpose of this interface is to (i) Manage the preferences specified for the availability of an identity (ii) Query for the availability of identities for specific capabilities.

The interfaces and the relationships between them are shown in the figure below.

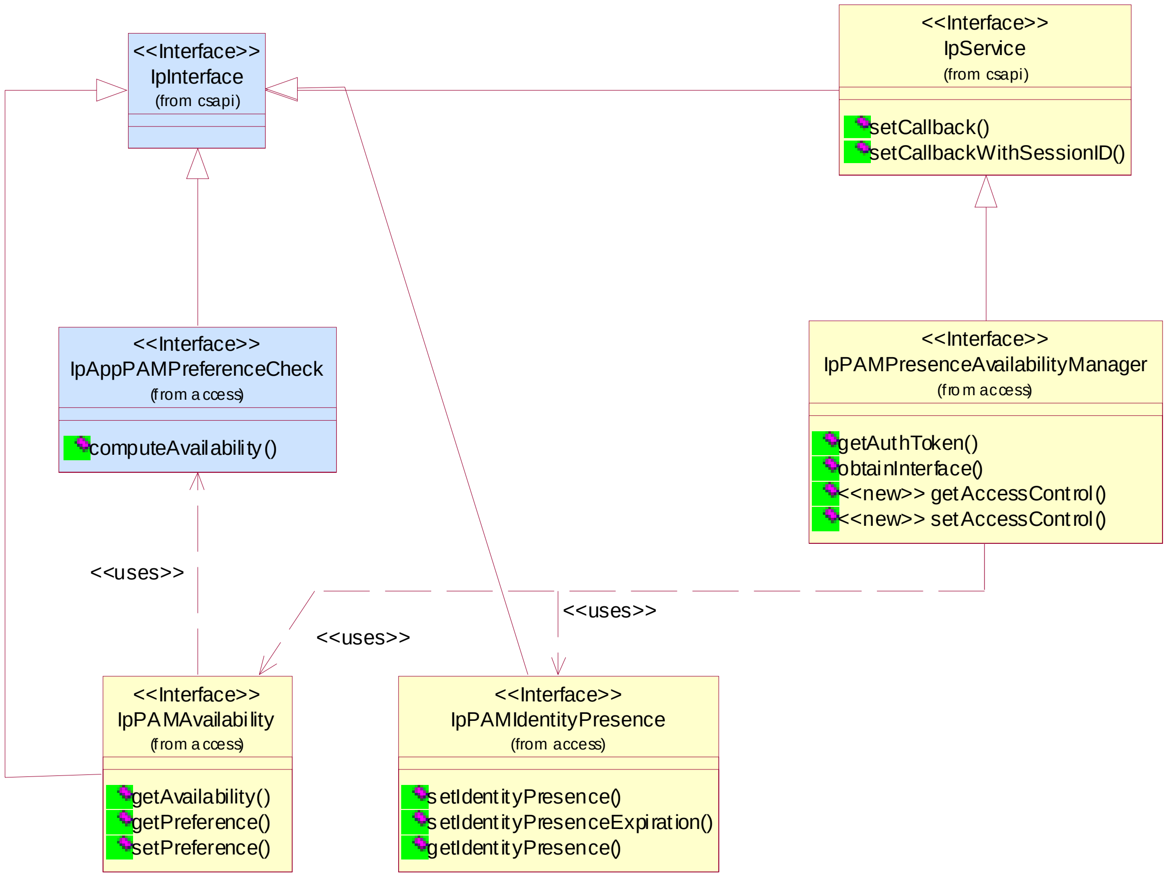


Figure: PAM Access Service

6.2 PAM Event SCF Class Diagrams

The PAM Event Management service consists of two packages, one for the application interfaces and one for the service interfaces. The application PAM event management package consists of 0 or more instances of the IpAppPAMEventHandler interface and the PAM event management service package consists of a single instance of the IpPAMEventHandler interface. This interface can be obtained by application using the service interface IpPAMEventManager.

The figure below shows the interfaces of PAM Event Management and the relationships between them.

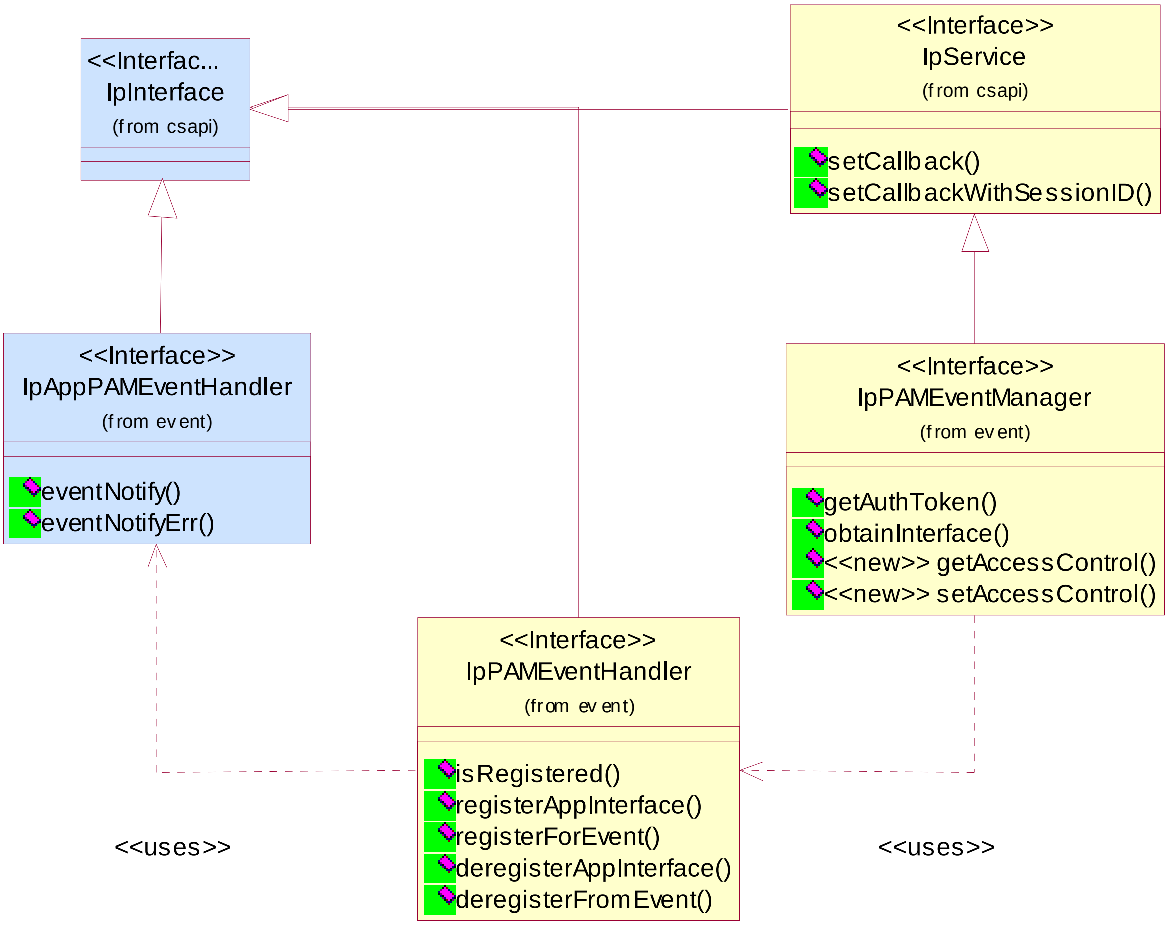


Figure: PAM Event Management Service

7 The Service Interface Specifications

7.1 Interface Specification Format

This clause defines the interfaces, methods and parameters that form a part of the API specification. The Unified Modelling Language (UML) is used to specify the interface classes. The general format of an interface specification is described below.

7.1.1 Interface Class

This shows a UML interface class description of the methods supported by that interface, and the relevant parameters and types. The Service and Framework interfaces for enterprise-based client applications are denoted by classes with name `Ip<name>`. The callback interfaces to the applications are denoted by classes with name `IpApp<name>`. For the interfaces between a Service and the Framework, the Service interfaces are typically denoted by classes with name IpSvc<name>, while the Framework interfaces are denoted by classes with name IpFw<name>

7.1.2 Method descriptions

Each method (API method “call”) is described. Both synchronous and asynchronous methods are used in the API. Asynchronous methods are identified by a '`Req`' suffix for a method request, and, if applicable, are served by asynchronous methods identified by either a '`Res`' or '`Err`' suffix for method results and errors, respectively. To handle responses and reports, the application or service developer must implement the relevant `IpApp<name> or IpSvc<name>` interfaces to provide the callback mechanism.

7.1.3 Parameter descriptions

Each method parameter and its possible values are described. Parameters described as 'in' represent those that must have a value when the method is called. Those described as 'out' are those that contain the return result of the method when the method returns.

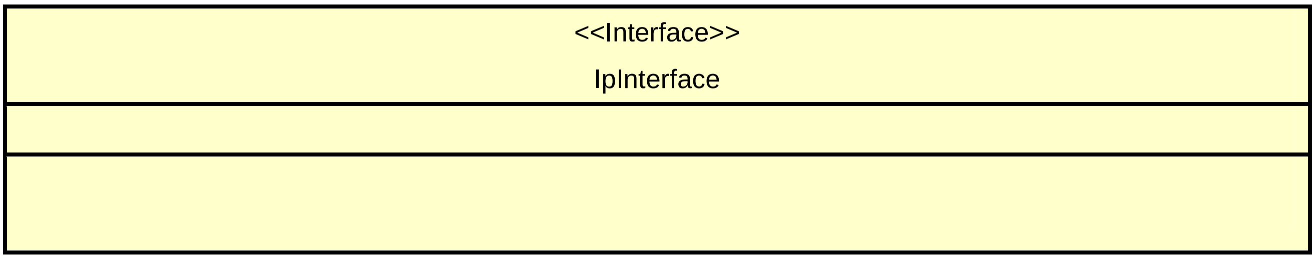
7.1.4 State Model

If relevant, a state model is shown to illustrate the states of the objects that implement the described interface.

7.2 Base Interface

7.2.1 Interface Class IpInterface

All application, framework and service interfaces inherit from the following interface. This API Base Interface does not provide any additional methods.



7.3 Service Interfaces

7.3.1 Overview

The Service Interfaces provide the interfaces into the capabilities of the underlying network - such as call control, user interaction, messaging, mobility and connectivity management.

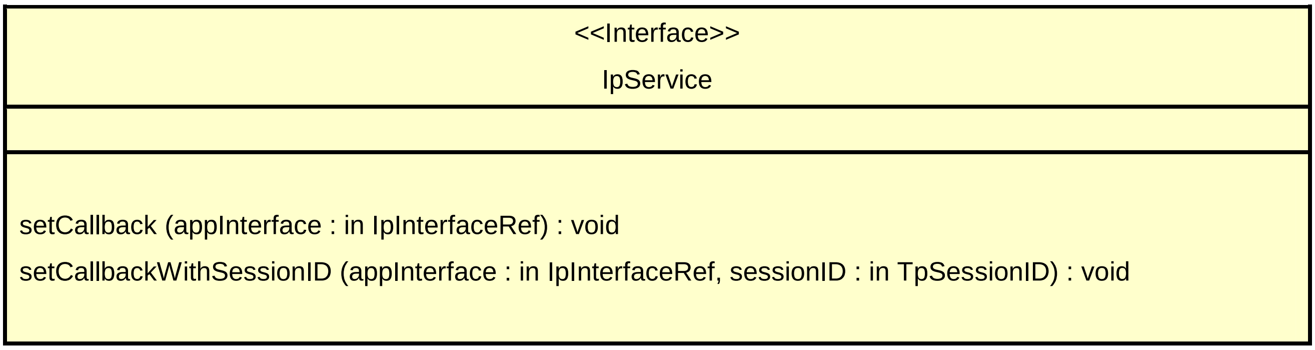
The interfaces that are implemented by the services are denoted as 'Service Interface'. The corresponding interfaces that must be implemented by the application (e.g. for API callbacks) are denoted as 'Application Interface'.

7.4 Generic Service Interface

7.4.1 Interface Class IpService

Inherits from: IpInterface

All service interfaces inherit from the following interface.



<<Interface>>

IpService

setCallback (appInterface : in IpInterfaceRef) : void

setCallbackWithSessionID (appInterface : in IpInterfaceRef, sessionID : in TpSessionID) : void

7.4.1.1 Method setCallback()

This method specifies the reference address of the callback interface that a service uses to invoke methods on the application. It is not allowed to invoke this method on an interface that uses SessionIDs.

Parameters

appInterface : in IpInterfaceRef

Specifies a reference to the application interface, which is used for callbacks.

Raises

TpCommonExceptions, P_INVALID_INTERFACE_TYPE

7.4.1.2 Method setCallbackWithSessionID()

This method specifies the reference address of the application's callback interface that a service uses for interactions associated with a specific session ID: e.g. a specific call, or call leg. It is not allowed to invoke this method on an interface that does not use SessionIDs.

Parameters

appInterface : in IpInterfaceRef

Specifies a reference to the application interface, which is used for callbacks.

sessionID : in TpSessionID

Specifies the session for which the service can invoke the application's callback interface.

Raises

TpCommonExceptions, P_INVALID_SESSION_ID, P_INVALID_INTERFACE_TYPE

8 Presence and Availability Management Interface Classes

PAM consists of the following SCFs

- PAM Provisioning Service (not included in the 3GPP release 5 specifications)
- PAM Access Service

- PAM Event Service

The PAM Access service consists of the identity presence and availability interfaces.

The Event service consists of the Event Management interfaces.

An implementation of this API which supports or implements a method described in the present document, shall support or implement the functionality described for that method, for at least one valid set of values for the parameters of that method. Where a method is not supported by an implementation of a Service interface, the exception P_METHOD_NOT_SUPPORTED shall be returned to any call of that method.

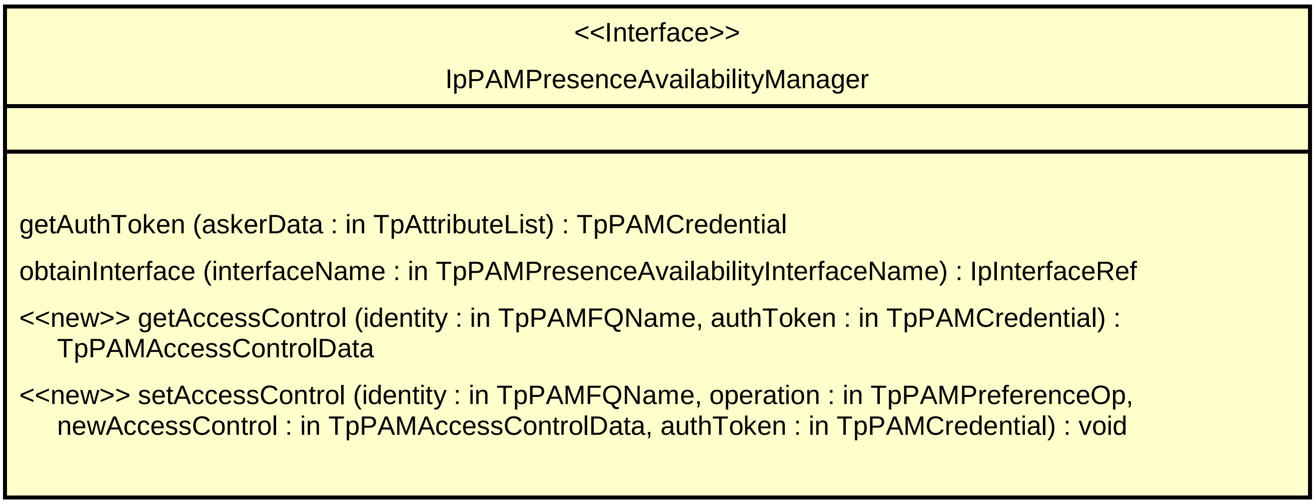
8.1 PAM Access SCF Interface Classes

This service consists of the presence and availability query and update interfaces.

8.1.1 Interface Class IpPAMPresenceAvailabilityManager

Inherits from: IpService.

The purpose of this interface is to supply the various interfaces available in this service to the application and to provide the authentication credentials. This interface is the only discoverable interface from the framework.
All PAM methods optionally use an authentication token as a parameter since the outcome of the operations may depend on the entity requesting the operation. To enable this, the getAuthToken() method is used to obtain an implementation dependent token. An application that has authenticated itself with the OSA framework, can get an authentication token for itself. Alternatively, if the application is requesting PAM operations on behalf of multiple entities, authentication tokens may be requested for each such entity after providing any available data about the asker. These tokens can then be used repeatedly for operations within a session without further need to identify the asker.



<<Interface>>

IpPAMPresenceAvailabilityManager

getAuthToken (askerData : in TpAttributeList) : TpPAMCredential

obtainInterface (interfaceName : in TpPAMPresenceAvailabilityInterfaceName) : IpInterfaceRef

<<new>> getAccessControl (identity : in TpPAMFQName, authToken : in TpPAMCredential) : TpPAMAccessControlData

<<new>> setAccessControl (identity : in TpPAMFQName, operation : in TpPAMPreferenceOp, newAccessControl : in TpPAMAccessControlData, authToken : in TpPAMCredential) : void

8.1.1.1 Method getAuthToken()

Get an authentication token for access to the interface methods.

Returns an implementation-dependent authentication credential that can be verified.

Parameters

askerData : in TpAttributeList

Specifies information about the asker. Can be an empty array. The exact attributes in this list are dependent on the application. PAM reserves the attribute "name" with type TpPAMFQName to contain the identity of the asker if known.

Returns

TpPAMCredential

Raises

TpCommonExceptions, P_PAM_INVALID_CREDENTIAL

8.1.1.2 Method obtainInterface()

Obtain available interfaces from the service. The valid parameters for this method can be obtained from the service property P_OBTAINABLE_INTERFACES.

Returns the requested interface.

Parameters

interfaceName : in TpPAMPresenceAvailabilityInterfaceName

Specifies the name of the required interface.

Returns

IpInterfaceRef

Raises

TpCommonExceptions, P_PAM_UNAVAILABLE_INTERFACE

8.1.1.3 Method <<new>> getAccessControl()

Get the access control associated with the data belonging to an identity. The data associated with an identity includes the static and dynamic attributes of an identity as well as data about agents associated with an identity.

This method should be used in conjunction with the setAccessControl method.

Returns the access control if previously specified for the identity. Is null if there is no access control associated.

Parameters

identity : in TpPAMFQName

Specifies the identity of interest.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Returns

TpPAMAccessControlData

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL

8.1.1.4 Method <<new>> setAccessControl()

Set the access controls for the data associated with the specified identity. If the identity is Null, the access control is set for all identities (if authorized to do so). The data associated with an identity includes the static and dynamic attributes of an identity as well as data about agents associated with an identity.

Any existing access control will be modified based on the operation.

If the new access control is specified as Null for replace operation , an existing access control will be removed.

Parameters

identity : in TpPAMFQName

Specifies the identity of interest.

operation : in TpPAMPreferenceOp

Specifies the operation to be performed with the specified preference.

newAccessControl : in TpPAMAccessControlData

Specifies the access controls to add.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL

8.1.2 Interface Class IpPAMIdentityPresence

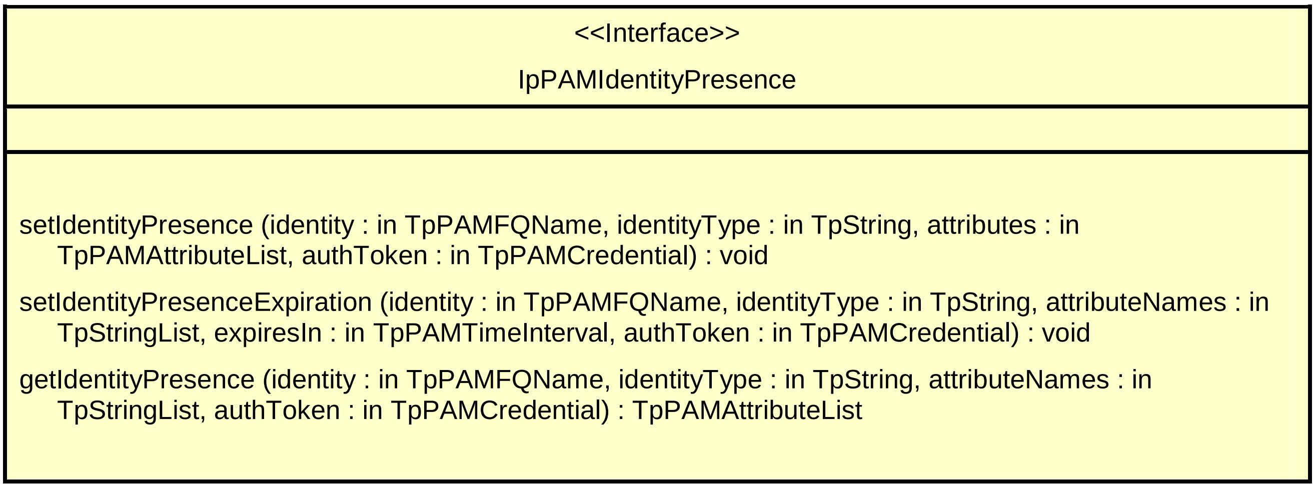
Inherits from: IpInterface.

The purpose of this interface is to maintain the dynamic presence information of identity.

The underlying implementations may optimize the storage for this dynamic data rather than rely on a general-purpose directory or database when performance is an issue. Presence information for identities may be explicitly registered are may be implicitly derived from the underlying networks or presence information from agents associated with the identity.

This interface is meant for use by applications that register and/or maintain dynamic presence information associated with identities and accessible without the privacy or other controls established by availability preferences. These applications may not be aware of the name and the types of agents associated with the identity.

The presence information can be explicitly registered using the interface or the presence may come from information implicitly derived (e.g., using presence information of agents associated with the identity).



<<Interface>>

IpPAMIdentityPresence

setIdentityPresence (identity : in TpPAMFQName, identityType : in TpString, attributes : in TpPAMAttributeList, authToken : in TpPAMCredential) : void

setIdentityPresenceExpiration (identity : in TpPAMFQName, identityType : in TpString, attributeNames : in TpStringList, expiresIn : in TpPAMTimeInterval, authToken : in TpPAMCredential) : void

getIdentityPresence (identity : in TpPAMFQName, identityType : in TpString, attributeNames : in TpStringList, authToken : in TpPAMCredential) : TpPAMAttributeList

8.1.2.1 Method setIdentityPresence()

Set identity's dynamic attributes.

Parameters

identity : in TpPAMFQName

Specifies the identity.

identityType : in TpString

Specifies the type of the identity.

attributes : in TpPAMAttributeList

Specifies the attributes to set.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_UNKNOWN_TYPE, P_PAM_UNKNOWN_ATTRIBUTE, P_PAM_INVALID_CREDENTIAL

8.1.2.2 Method setIdentityPresenceExpiration()

Set or reset the expiration of an identity's named presence attributes. If the attributeNames parameter is an empty list, the expiration time of all attributes defined for the identity will have their expiration time changed.

Parameters

identity : in TpPAMFQName

Specifies the identity.

identityType : in TpString

Specifies the type of the identity.

attributeNames : in TpStringList

Specifies the names of the attributes. Can be an empty list.

expiresIn : in TpPAMTimeInterval

Specifies the number of seconds until the attributes expire. A value of -1 indicates no expiration.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_UNKNOWN_TYPE, P_PAM_UNKNOWN_ATTRIBUTE, P_PAM_INVALID_CREDENTIAL

8.1.2.3 Method getIdentityPresence()

Retrieve presence attributes associated with an identity.

Return value contains the requested attributes of the named capability. If the attributes parameter is an empty array, all attributes of the named profile are included.

Parameters

identity : in TpPAMFQName

Specifies the identity.

identityType : in TpString

Specifies the type of the identity.

attributeNames : in TpStringList

Specifies the attributes of interest. Can be an empty list.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Returns

TpPAMAttributeList

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_UNKNOWN_TYPE, P_PAM_UNKNOWN_ATTRIBUTE, P_PAM_INVALID_CREDENTIAL

8.1.3 Interface Class IpPAMAvailability

Inherits from: IpInterface.

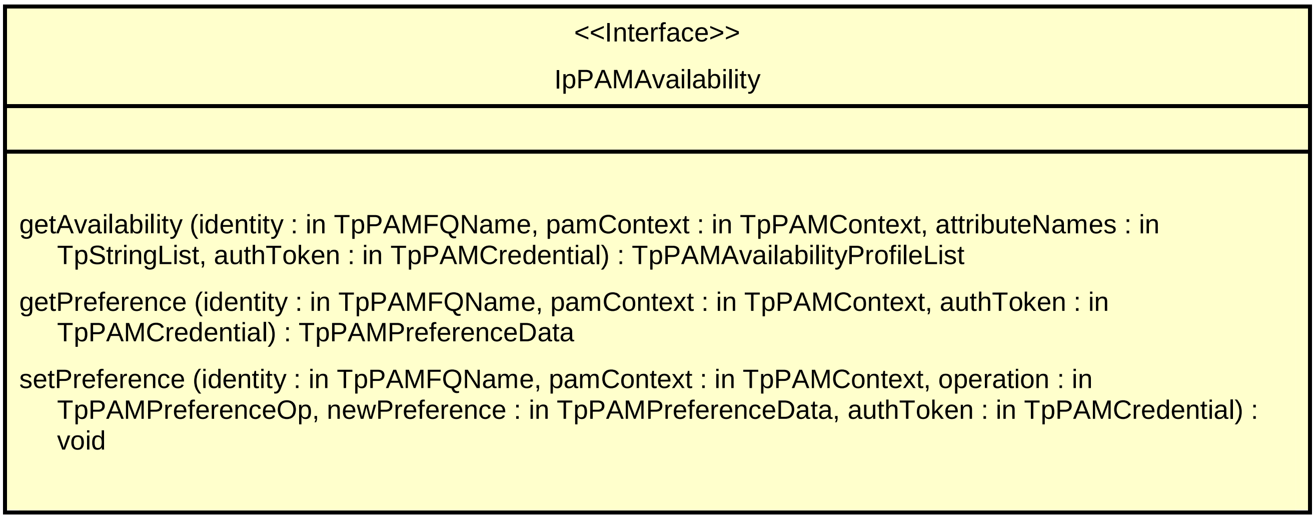
The purpose of the interface is to
- Manage the preferences specified for the availability of an identity and, to
- Query for the availability of identities for specific capabilities.

- Query for attributes of interest from an identity.

Simple implementations may equate the availability of identities to presence of their agents with available status. More complex implementations may consider, in addition, the preferences specified for availability as well as the attributes of the entity asking for availability.

The queries for availability are done for a specified context. A context is a set of attributes describing the situation for which availability is requested. PAM specifies one pre-defined context - Communication. The Communication context is used for availability for a specific mode of communication. Applications and PAM implementations may extend and provide additional contexts such as availability at a particular location, availability for a specific mode of communication at a given location, etc. The context information also includes any information about the asker as may be provided by the asker.

The specification defines two types of preference mechanisms although implementations may support additional mechanisms. The first mechanism consists of access control lists that specify identities that are allowed/denied to access information about the identity whose preference is being set. The second mechanism allow for an external application interface to be specified to check for access control as well as to compute availability.



<<Interface>>

IpPAMAvailability

getAvailability (identity : in TpPAMFQName, pamContext : in TpPAMContext, attributeNames : in TpStringList, authToken : in TpPAMCredential) : TpPAMAvailabilityProfileList

getPreference (identity : in TpPAMFQName, pamContext : in TpPAMContext, authToken : in TpPAMCredential) : TpPAMPreferenceData

setPreference (identity : in TpPAMFQName, pamContext : in TpPAMContext, operation : in TpPAMPreferenceOp, newPreference : in TpPAMPreferenceData, authToken : in TpPAMCredential) : void

8.1.3.1 Method getAvailability()

Get the availability for an identity for a given context.

All contexts may optionally include an asker profile. Although PAM applications may decide what attributes to include in an asker profile, PAM implementations should not require such attributes to be present. The implementations should leave it to the availability computations to decide the availability based on the (partial) information provided.

It is also up to the availability computation to decide on the trustworthiness of the asker profile information based on the application, the credentials of the entity asking for availability and/or the credentials, if any, of the entity accessing the interface.

Returns a value containing a list of attributes as available to the asker in the requested context. If no information is available to the asker an empty list is returned.

Parameters

identity : in TpPAMFQName

Specifies the identity for which the availability is being requested.

pamContext : in TpPAMContext

Specifies the context for which the availability is requested.

attributeNames : in TpStringList

Specifies the attributes of interest. Can be an empty list to indicate all attributes.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Returns

TpPAMAvailabilityProfileList

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL

8.1.3.2 Method getPreference()

Get the availability preferences of an identity for the specified communication mode.

This method should be used in conjunction with the setPreference method.

Returns the preference for the named context if previously specified for the identity. Is null if there are no preferences associated.

Parameters

identity : in TpPAMFQName

Specifies the identity of interest.

pamContext : in TpPAMContext

Specifies the context for which the preferences are requested.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Returns

TpPAMPreferenceData

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL

8.1.3.3 Method setPreference()

Set the availability preferences for the specified identity for the specified context. If the identity is Null, the preference is set for all identities (if authorized to do so).

The existing preference will be modified based on the operation.

If the new preference is specified as Null for replace operation , any existing preferences for the specified context will be removed.

Parameters

identity : in TpPAMFQName

Specifies the identity with which the preference will be associated.

pamContext : in TpPAMContext

Specifies the capability to which this preference applies.

operation : in TpPAMPreferenceOp

Specifies the operation to be performed with the specified preference

newPreference : in TpPAMPreferenceData

Specifies the availability preference to add.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

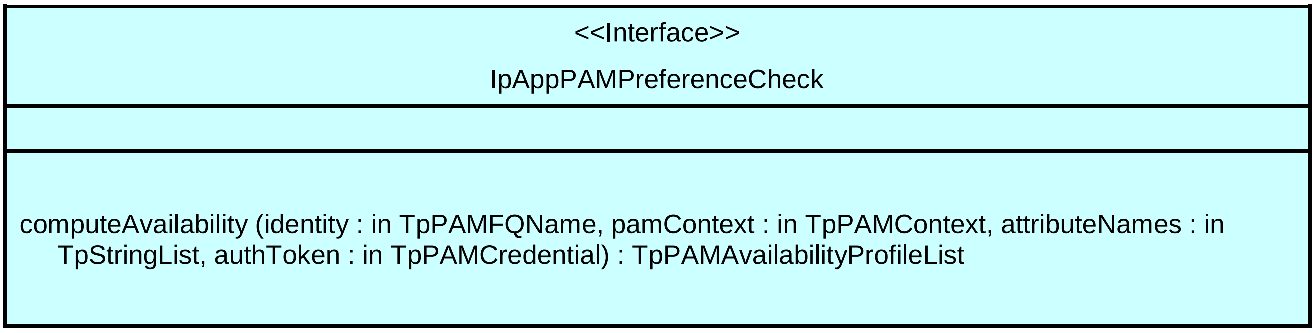
Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL

8.1.4 Interface Class IpAppPAMPreferenceCheck

Inherits from: IpInterface.

The purpose of this interface is to provide methods to be called by the PAM service to check for access control or to compute availability using an implementation provided by an application. Instances of this interface are registered using the setPreference() method in the availability management interface.



<<Interface>>

IpAppPAMPreferenceCheck

computeAvailability (identity : in TpPAMFQName, pamContext : in TpPAMContext, attributeNames : in TpStringList, authToken : in TpPAMCredential) : TpPAMAvailabilityProfileList

8.1.4.1 Method computeAvailability()

Compute the availability for an identity for a given context. The data provided is the same as the data provided for the getAvailability call. The application implementing this interface uses the identity presence interface to get the current presence data and maintains its own user preferences to compute the availability.

Returns a value containing a list of attributes as available to the asker in the requested context. If no information is available to the asker an empty list is returned.

Parameters

identity : in TpPAMFQName

Specifies the identity for which the availability is being requested.

pamContext : in TpPAMContext

Specifies the context for which the availability is requested.

attributeNames : in TpStringList

Specifies the attributes of interest. Can be an empty list to indicate all attributes.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Returns

TpPAMAvailabilityProfileList

8.2 PAM Event SCF Interface Classes

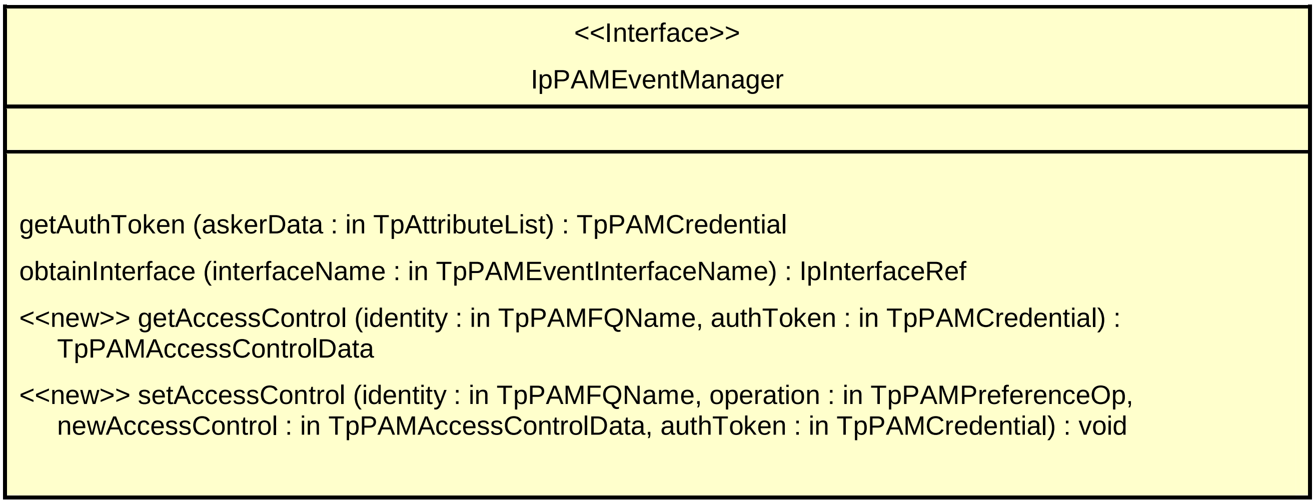
This service contains an interface for registering for notifications for events that occur within the PAM service.

8.2.1 Interface Class IpPAMEventManager

Inherits from: IpService.

The purpose of this interface is to supply the various interfaces available in this service to the application and to provide the authentication credentials. This interface is the only discoverable interface from the framework.

All PAM methods use an authentication token as a parameter since the outcome of the operations may depend on the entity requesting the operation. To enable this, the getAuthToken() method is used to obtain an implementation dependent token. An application that has authenticated itself with the OSA framework, can get an authentication token for itself. Alternatively, if the application is requesting PAM operations on behalf of multiple entities, authentication tokens may be requested for each such entity after providing any available data about the asker. These tokens can then be used repeatedly for operations within a session without further need to identify the asker.



<<Interface>>

IpPAMEventManager

getAuthToken (askerData : in TpAttributeList) : TpPAMCredential

obtainInterface (interfaceName : in TpPAMEventInterfaceName) : IpInterfaceRef

<<new>> getAccessControl (identity : in TpPAMFQName, authToken : in TpPAMCredential) : TpPAMAccessControlData

<<new>> setAccessControl (identity : in TpPAMFQName, operation : in TpPAMPreferenceOp, newAccessControl : in TpPAMAccessControlData, authToken : in TpPAMCredential) : void

8.2.1.1 Method getAuthToken()

Get an authentication token for access to the interface methods.

Returns an implementation-dependent authentication credential that can be verified.

Parameters

askerData : in TpAttributeList

Specifies information about the asker. Can be an empty array. The exact attributes in this list are dependent on the application. PAM reserves the attribute "name" with type TpPAMFQName to contain the identity of the asker if known.

Returns

TpPAMCredential

Raises

TpCommonExceptions, P_PAM_INVALID_CREDENTIAL

8.2.1.2 Method obtainInterface()

Obtain available interfaces from the service. The valid parameters for this method can be obtained from the service property P_OBTAINABLE_INTERFACES.

Returns the requested interface.

Parameters

interfaceName : in TpPAMEventInterfaceName

Specifies the name of the required interface.

Returns

IpInterfaceRef

Raises

TpCommonExceptions, P_PAM_UNAVAILABLE_INTERFACE

8.2.1.3 Method <<new>> getAccessControl()

Get the access control associated with the data belonging to an identity. The data associated with an identity includes the static and dynamic attributes of an identity as well as data about agents associated with an identity.

This method should be used in conjunction with the setAccessControl method.

Returns the access control if previously specified for the identity. Is null if there is no access control associated.

Parameters

identity : in TpPAMFQName

Specifies the identity of interest.

authToken : in TpPAMCredential

Of the entity who wishes to do this operation.

Returns

TpPAMAccessControlData

Raises

TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL

8.2.1.4 Method <<new>> setAccessControl()

Set the access controls for the data associated with the specified identity. If the identity is Null, the access control is set for all identities (if authorized to do so). The data associated with an identity includes the static and dynamic attributes of an identity as well as data about agents associated with an identity.

Any existing access control will be modified based on the operation.

If the new access control is specified as Null for replace operation , an existing access control will be removed.

Parameters

`identity : in TpPAMFQName`

Specifies the identity of interest.

`operation : in TpPAMPreferenceOp`

Specifies the operation to be performed with the specified preference.

`newAccessControl : in TpPAMAccessControlData`

Specifies the access controls to add.

`authToken : in TpPAMCredential`

Of the entity who wishes to do this operation.

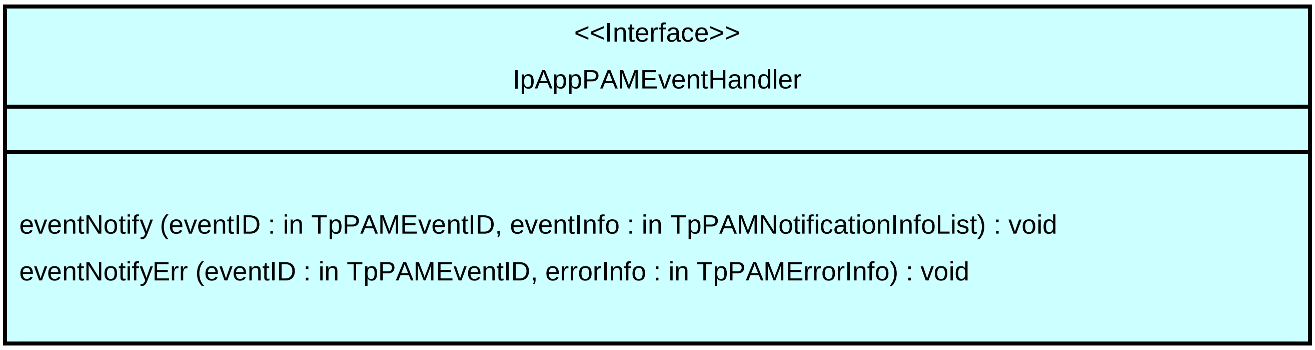
Raises

`TpCommonExceptions, P_PAM_UNKNOWN_IDENTITY, P_PAM_INVALID_CREDENTIAL`

8.2.2 Interface Class IpAppPAMEventHandler

Inherits from: IpInterface.

This is the interface that a client application must implement and register with the Event Service in order to be notified of events.



<<Interface>>

IpAppPAMEventHandler

eventNotify (eventID : in TpPAMEventID, eventInfo : in TpPAMNotificationInfoList) : void

eventNotifyErr (eventID : in TpPAMEventID, errorInfo : in TpPAMErrorInfo) : void

8.2.2.1 Method eventNotify()

Notify the occurrence of an event. The implementations will not attempt to re-notify on failure.

Parameters

`eventID : in TpPAMEventID`

Specifies a prior event registration ID.

eventInfo : in TpPAMNotificationInfoList

Contains the data about the events that occurred.

8.2.2.2 Method eventNotifyErr()

Notify an error in the event reporting. The error may concern all assignments, one whole assignment or a part of it. An eventNotify is sent after the error condition has passed away unless the event has been subsequently deregistered. Re-registration may only be needed in fatal system error cases. Note that in normal operation unavailable or protected pieces of presence information are delivered by the normal reporting methods instead of an error method.

Parameters

eventID : in TpPAMEventID

Specifies a prior event registration ID.

errorInfo : in TpPAMErrorInfo

Contains the data relating to the error.

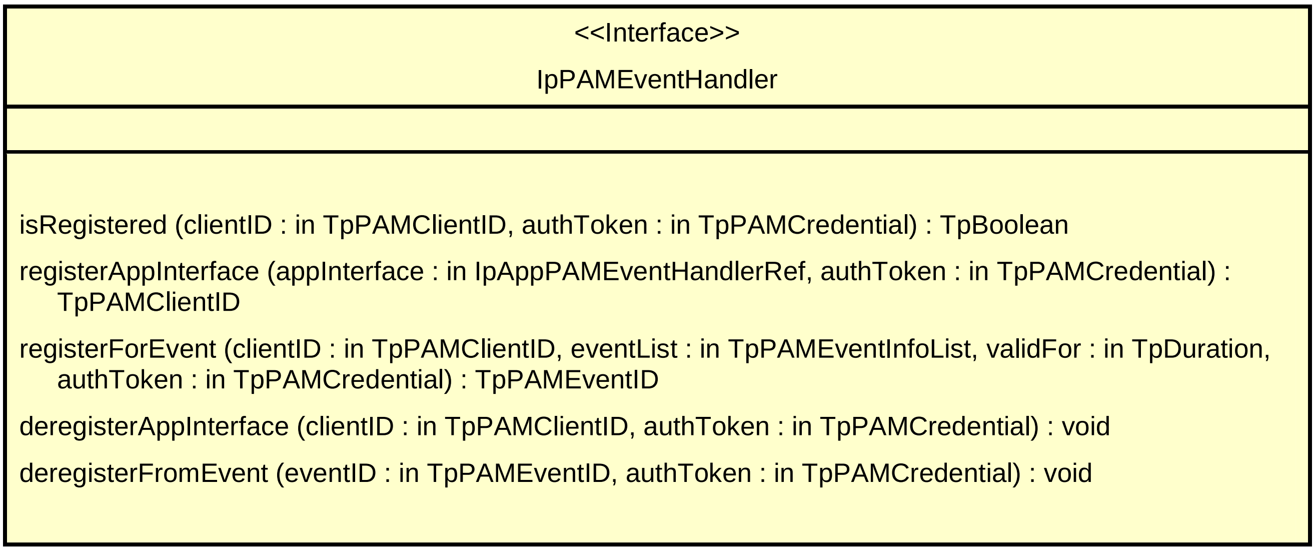
8.2.3 Interface Class IpPAMEventHandler

Inherits from: IpInterface.

The purpose of this interface is to manage the registrations of interest in events and the registration of client interfaces for subsequent notification. All notifications in the present document are to be sent after the corresponding event has occurred and are asynchronous. An application must first register a notification interface with the service. It can then register interest in one or more events for this interface.

A failure or a reset of a PAM implementation may result in a loss of all prior event and interface registrations. The client application may need to confirm the continued registration of the notification interface and re-register if necessary.

For security and privacy purposes, a registration for an event is allowed if and only if the supplied credentials during registration is sufficient to have allowed access to the information related to the event through one or more of the PAM interface methods.



<<Interface>>

IpPAMEventHandler

isRegistered (clientID : in TpPAMClientID, authToken : in TpPAMCredential) : TpBoolean

registerAppInterface (appInterface : in IpAppPAMEventHandlerRef, authToken : in TpPAMCredential) : TpPAMClientID

registerForEvent (clientID : in TpPAMClientID, eventList : in TpPAMEventInfoList, validFor : in TpDuration, authToken : in TpPAMCredential) : TpPAMEventID

deregisterAppInterface (clientID : in TpPAMClientID, authToken : in TpPAMCredential) : void

deregisterFromEvent (eventID : in TpPAMEventID, authToken : in TpPAMCredential) : void

8.2.3.1 Method isRegistered()

Check if a client application interface is registered.

Returns True if the registration ID is still valid, False otherwise.

Parameters

`clientID : in TpPAMClientID`

Specifies the registration ID provided at registration.

`authToken : in TpPAMCredential`

Credential of the entity who wishes to do this operation.

Returns

`TpBoolean`

Raises

`TpCommonExceptions, P_PAM_INVALID_CREDENTIAL`

8.2.3.2 Method registerAppInterface()

Register a client application's notification interface.

Returns an ID returned by the service that uniquely identifies this registration.

Parameters

`appInterface : in IpAppPAMEventHandlerRef`

Specifies the client notification interface.

`authToken : in TpPAMCredential`

Credential of the entity who wishes to do this operation.

Returns

`TpPAMClientID`

Raises

`TpCommonExceptions, P_PAM_INVALID_CREDENTIAL`

8.2.3.3 Method registerForEvent()

Register a client application's interest in one or more events.

Returns an ID returned by the service that uniquely identifies this registration for the event.

Parameters

`clientID : in TpPAMClientID`

Specifies the registration ID provided at registration.

eventList : in TpPAMEventInfoList

Specifies the events of interest.

validFor : in TpDuration

Specifies the interval in milliseconds until which the subscription is held and notifications provided. A time interval of 0 or negative values indicate a subscription that never expires until explicitly canceled.

authToken : in TpPAMCredential

Credential of the entity who wishes to do this operation.

Returns

TpPAMEventID

Raises

TpCommonExceptions, P_PAM_NOT_REGISTERED, P_PAM_INVALID_CREDENTIAL

8.2.3.4 Method deregisterAppInterface()

Unregister a client application's notification interface.

All registrations for events for this client registration are also removed.

Parameters

clientID : in TpPAMClientID

Specifies the registration ID provided at registration.

authToken : in TpPAMCredential

Credential of the entity who wishes to do this operation.

Raises

TpCommonExceptions, P_PAM_NOT_REGISTERED, P_PAM_INVALID_CREDENTIAL

8.2.3.5 Method deregisterFromEvent()

Unregister a client application's interest in an event.

Parameters

eventID : in TpPAMEventID

Specifies a prior event registration ID.

authToken : in TpPAMCredential

Credential of the entity who wishes to do this operation.

Raises

TpCommonExceptions, P_PAM_NOT_REGISTERED, P_PAM_INVALID_CREDENTIAL

9 State Transition Diagrams

There are no State Transition Diagrams for the Presence and Availability Management SCFs.

10 PAM Service Properties

The following table lists properties relevant to all the PAM SCFs

Property	Type	Description
P_OBTAINABLE_INTERFACES	STRING_SET	The interfaces obtainable from the service

10.1 PAM Provisioning service properties

Implementations of the PAM Provisioning APIs for 3GPPshall have the Service Properties set to the indicated values:

```
P_OBTAINABLE_INTERFACES = {}
```

10.2 PAM Access Service

Implementations of the PAM Access APIs for 3GPPshall have the Service Properties set to the indicated values:

```
P_OBTAINABLE_INTERFACES = {
P_PAM_IDENTITY_PRESENCE,
P_PAM_AVAILABILITY
}
```

10.3 PAM Event Service

PAM Event service has the following property in addition to the above.

Property	Type	Description
P_EVENT_TYPES	INTEGER_SET	The pre-defined event types that can be registered for

Implementations of the PAM Event APIs for 3GPP shall have the Service Properties set to the indicated values:

```
P_OBTAINABLE_INTERFACES = {
P_PAM_EVENT_HANDLER
}
P_EVENT_TYPES = {
PAM_CE_IDENTITY_PRESENCE_SET,
PAM_CE_AVAILABILITY_CHANGED,
PAM_CE_WATCHERS_CHANGED
}
```

11 PAM Data Definitions

All data types referenced in this document but not defined in this clause are common data definitions which may be found in 3GPP TS 29.198-2.

11.1 Entity Address Definitions

11.1.1 TpPAMFQName

This is the same as TpURN and is used to name entities in PAM Access service.

11.1.2 TpPAMFQNameList

This is a `Numbered List of Data Elements` of type TpPAMFQName.

11.2 Attribute Data Definitions

11.2.1 TpPAMAttribute

This is a `Sequence of Data Elements` containing the attribute name, type, expiration time and value. This is derived from the common attribute type TpAttribute to add the expiration value for dynamic attributes.

Sequence Element Name	Sequence Element Type	Notes
`AttributeName`	`TpString`	The name of the attribute.
`AttributeType`	`TpAttributeType`	The type of the attirbute. Valid values for Type must include at least TpString, TpInt32 and TpFloat.
`AttributeValue`	`TpAny`	The values for the attribute. This model allows multi-valued attributes. Cannot be an empty list.
`ExpiresIn`	`TpPAMTimeInterval`	The interval in milliseconds in which the attribute values are valid. A time interval of PAM_MAX_LONGINT indicates static attribute values that never expire. A time interval of 0 or negative values indicate an expired value and the time for which it has expired.

11.2.2 TpPAMAttributeList

This is a `Numbered List of Data Elements` of type TpPAMAttribute.

11.2.3 TpPAMAttributeDef

This is a Sequence of Data Elements containing the definition of an attribute. This definition constitutes the “schema” for an attribute and contains fields to define the type and behavior of a dynamic attribute. Each definition using these fields results in a TpPAMAttribute with the corresponding name and type and dynamic behavior as defined by the remaining fields. In 3GPP Release 5, no methods exist to create PAM attributes at runtime and hence this type is not used in any method. However, certain pre-defined attributes are defined for identity presence in Section 11.10 using the following fields. This type is included in this document to specify the semantics of the fields in the pre-defined attributes.

Sequence Element Name	Sequence Element Type	Notes
Name	TpString	Name of attribute
Type	TpString	Type of attribute. Valid values for Type must include at least TpString, TpInt32 and TpFloat
IsStatic	TpBoolean	True indicates that the attributes is always static and its values never expire. False indicates that the attribute can be dynamic and may contain values that expire.
IsRevertOnExpiration	TpBoolean	True indicates that the attribute reverts to the default value on expiration. False indicates that the attribute will not revert to the default value.
DefaultValues	TpAny	An attribute is always initialized with this value. If the *isRevertOnExpiration* attribute is set to true, a dynamic attribute that has expired while stored in a PAM implementation is reset to this value with the *expiresIn* interval set to PAM_MAX_LONGINT. The default attribute value is interpreted based on the value of the attribute Type.

11.2.4 TpPAMAttributeDefList

This is a Numbered List of Data Elements of type TpPAMAttributeDef.

11.3 Presence Data Definitions

11.3.1 TpPAMCapability

This defines the extensible communication capabilities. This data type is identical to a TpString, and is defined as a string of characters that specify the communication capabilities. The following strings are pre-defined.

Character String Value	Description
P_PAM_VOICE	Capability for voice calls
P_PAM_SMS	Capability for SMS
P_PAM_IM	Capability for Instant Messaging
P_PAM_MMS	Capability for Multi-media messaging

11.3.2 TpPAMCapabilityList

This is a Numbered List of Data Elements of type TpPAMCapability.

11.3.3 TpPAMPresenceData

This is a Sequence of Data Elements for a specific identity type Presentity pre-defined in PAM. Since multiple presence data records can be associated with an identity, each distinct record is uniquely named.

Sequence Element Name	Sequence Element Type	Notes
Name	TpString	Name of presence data
PresenceAttributes	TpPAMAttributeList	Presence Attributes

11.3.4 TpPAMPresenceDataList

This is a Numbered List of Data Elements of type TpPAMPresenceData.

11.4 Pre-defined Presence type

11.4.1 Presentity

An identity type Presentity is pre-defined for all identities associated with the attribute PresenceProfile defined as:

Attribute Definition Field	Value
Name	PresenceProfile
Type	TpPAMPresenceData
IsStatic	False
IsRevertOnExpiration	False
DefaultValues	Null

11.5 Availability Data Definitions

11.5.1 TpPAMAvailabilityProfile

This is a Sequence of Data Elements containing the list of attribute values as determined by the definition of the context for which the availability is provided.

Sequence Element Name	Sequence Element Type	Notes
PrivacyCode	TpPAMPrivacyCode	Contains the privacy codes
AvailabilityData	TpPAMPresenceData	Contains a list of presence attributes

11.5.2 TpPAMAvailabilityProfileList

This is a Numbered List of Data Elements of type TpPAMAvailabilityProfile.

11.5.3 TpPAMPrivacyCode

This data type is identical to a TpString, and is defined as a string of characters that specify the privacy code for availability profiles. These codes are just indications of the privacy expected by the service and not are meant to be enforced by the service. Other Network operator specific codes may also be used, but should be preceded by the string "S_". The following values are defined.

Character String Value	Description
PAM_CP_ASKER_ONLY	The profile is available to the asker only and should not be further transmitted
PAM_CP_AUTHORIZED	The profile can be provided by the asker to authorized entities
PAM_CP_UNLIMITED	The profile can be distributed without limits

11.6 Availability Context Data Definitions

Availability is always queried for in a specific context on behalf of an asker. There is one context for communication pre-defined in this version of the specification.

11.6.1 TpPAMContext

This is a `Sequence of Data Elements` containing the data which defines the context in which an availability is queried and information about the asker that is requesting the data.

Sequence Element Name	Sequence Element Type	Notes
ContextData	TpPAMContextData	Contains the context name and the list of attributes that define the context. The attributes to be included for a given context are specified by the definition of the context.
AskerData	TpAttributeList	Contains information about the asker of availability. The exact attributes in this list are dependent on the application. PAM reserves the attribute “name” with type TpPAMFQName to contain the identity of the asker if known.

.

11.6.2 TpPAMContextName

This specifies the availability contexts.

Name	Value	Description
PAM_CONTEXT_ANY	0	Denotes any known context
PAM_CONTEXT_COMMUNICATION	1	Denotes a communication context

11.6.3 TpPAMContextData

This is a `tagged choice of data elements` that specifies the optional data that may be required to define a particular context

	Tag Element Type	
	TpPAMContextName	

Tag Element Value	Choice Element Type	Choice Element Name
PAM_CONTEXT_ANY	None	Undefined
PAM_CONTEXT_COMMUNICATION	TpPAMCommunicationContext	CommunicationContext

11.6.4 TpPAMCommunicationContext

This is a `Sequence of Data Elements` containing the list of attribute values for defining a communication context.

Sequence Element Name	Sequence Element Type	Notes
`CommunicationCapability`	`TpPAMCapability`	Specifies the communication type for which the availability is requested

11.6.5 TpPAMContextList

This is a `Numbered List of Data Elements` of type TpPAMContext.

11.7 Credential data definitions

11.7.1 TpPAMCredential

This is the same as TpOctetSet. This data is opaque to the application and is implementation dependent. As this data is valid only in the context of a single session with the service and hence cannot be used across multiple services, there are no inter-operability issues here. The application simply uses the credential returned from the getAuthToken() method in all other methods that require the credentials.

11.8 Availability and Access Control Preference Data Definitions

PAM allows several types of preferences to be specified. It includes an access control list specifying who is allowed to check for presence or subscribe to presence data for each identity. It also includes an interface for an application to register an interface to do access control checks and availability computations outside of the presence service.

11.8.1 IpAppPAMPreferenceCheckRef

Defines a Reference to type IpAppPAMPreferenceCheck.

11.8.2 TpPAMAccessControlData

This is a `Sequence of Data Elements` for access control data.

Sequence Element Name	Sequence Element Type	Notes
`DefaultPolicy`	`TpPAMACLDefault`	Specifies whether the default policy is to allow or deny access for names not mentioned in the list
`AllowList`	`TpPAMFQNameList`	Specifies a list of identities to be allowed access
`DenyList`	`TpPAMFQNameList`	Specifies a list of identities to be denied access

11.8.3 TpPAMACLDefault

Defines the two possible default policies for access control.

Name	Value	Description
`PAM_ACCESS_ALLOW`	`0`	Allow access by default
`PAM_ACCESS_DENY`	`1`	Deny access by default

11.8.4 TpPAMPreferenceOp

This data type is identical to a TpString, and is defined as a string of characters that specify the operations to be performed with a preference. The following values are defined.

Character String Value	Description
PAM_ACCESS_ADD	Add the specified preference to the current preferences
PAM_ACCESS_DELETE	Delete the specified preference from the current preferences
PAM_ACCESS_REPLACE	Replace the current preferences with the specified preference

11.8.5 TpPAMPreferenceType

This specifies the names of privacy and access control mechanisms supported by the service.

Name	Value	Description
PAM_ACCESS_LIST	0	The control data contains additions or modifications to access control list of who is authorized to access the presence information or subscribe to it.
PAM_EXTERNAL_CONTROL	1	The access control and availability computations are done external to the presence service

11.8.6 TpPAMPreferenceData

This is a tagged choice of data elements that specifies the preference data. The data depends on the type of preference being specified.

	Tag Element Type	
	TpPAMPreferenceType	

Tag Element Value	Choice Element Type	Choice Element Name
PAM_EXTERNAL_CONTROL	IpAppPAMPreferenceCheckRef	ExternalControlInterface

11.9 Time data definitions

11.9.1 TpPAMTimeInterval

This is identical to TpInt64.

11.10 Pre-defined Entity Types and Attributes

This version of the specification pre-defines one identity type called “Presentity”. The following constant can be used to refer to this Identity Type. All identities in the PAM service are associated with this identity type. For example, the identityType parameter in IpIdentityPresence and IpEventHandler methods take this as the value. This is also used in the event registration data structure (e.g., TpPAMAVCEventData) in the IdentityType field.

Character String Value	Description
P_PAM_PRESENTITY_TYPE	The pre-defined identity type called Presentity.

Every identity type in PAM can be defined with a set of attributes that are associated with all identities of that type. The following dynamic attributes are pre-defined as attributes of type TpPAMAttribute for the “Presentity” identity type and shall be supported as attributes of all identities in implementations of this service. These attributes are defined using TpPAMAttributeDef fields as follows:

AttributeName	AttributeType	IsStatic	IsRevertOn Expiration	DefaultValue	Description
P_SUBSCRIBER_STATUS	P_STRING	False	False	None	Specifies the status of the subscriber
P_NETWORK_STATUS	P_STRING	False	False	None	Specifies the status of the network
P_COMMUNICATION_MEANS	P_PAM_CAPABILITY	False	False	None	Specifies the means of communication. The type is TpPAMCapability
P_CONTACT_ADDRESS	P_ADDRESS	False	False	None	Address for communication
P_SUBSCRIBER_PROVIDED_LOCATION	P_STRING	False	False	None	Location nformation provided by subscriber. Is optional.
P_NETWORK_PROVIDED_LOCATION	P_STRING	False	False	None	Location information provided by subscriber. Is optional.
P_PRIORITY	P_INT32	False	False	None	Priority for communication
P_OTHER_INFO	P_STRING	False	False	None	Additional information

11.11 Interface name definitions

This section defines the names to be used for obtaining interfaces from the corresponding service interfaces in each PAM SCF.

11.11.1 TpPAMProvisioningInterfaceName

This data type is identical to a TpString, and is defined as a string of characters that identify the names of the PAM Provisioning interfaces that are to be supported by the OSA API.

Character String Value	Description
P_PAM_IDENTITY_MANAGEMENT	The name for the PAM Identity Management interface.
P_PAM_AGENT_MANAGEMENT	The name for the PAM Agent Management interface
P_PAM_AGENT_ASSIGNMENT	The name for the PAM Agent Assignment interface
P_PAM_IDENTITY_TYPE_MANAGEMENT	The name for the PAM Identity Type Management interface
P_PAM_AGENT_TYPE_MANAGEMENT	The name for the PAM Agent Type Management interface
P_PAM_CAPABILITY_TYPE_MANAGEMENT	The name for the PAM Capability Type Management interface

11.11.2 TpPAMPresenceAvailabilityInterfaceName

This data type is identical to a TpString, and is defined as a string of characters that identify the names of the PAM Access interfaces that are to be supported by the OSA API.

Character String Value	Description
P_PAM_IDENTITY_PRESENCE	The name for the PAM Identity Presence interface.
P_PAM_AGENT_PRESENCE	The name for the PAM Agent Presence interface
P_PAM_AVAILABILITY	The name for the PAM Availability interface

11.11.3 TpPAMEventInterfaceName

This data type is identical to a TpString, and is defined as a string of characters that identify the names of the PAM Event management interfaces that are to be supported by the OSA API.

Character String Value	Description
P_PAM_EVENT_HANDLER	The name for the Event Handler interface.

11.12 Event data definitions

There are two sets of data structures used for events. One set is used by applications to provide information when registering for an event and the second set is used to supply information to the applications in the notifications when the events occur.

11.12.1 IpAppPAMEventHandlerRef

Defines a Reference to type IpAppPAMEventHandler.

11.12.2 TpPAMClientID

This is the same is TpInt32 and is used to identify, uniquely within an implementation, registration of an application interface for notification of events.

11.12.3 TpPAMEventID

This is the same as TpAssignmentID and is used to identify, uniquely within an implementation, a registration for a specific event.

11.12.4 TpPAMEventName

This data type identifies the values that specify the event names.

Name	Value	Description
PAM_CE_IDENTITY_PRESENCE_SET	0	Notify if the value of presence attributes of an identity is explicitly set
PAM_CE_AVAILABILITY_CHANGED	1	Notify if the availability of an identity changes
PAM_CE_WATCHERS_CHANGED	2	Notify if the current set of watchers change
PAM_CE_IDENTITY_CREATED	3	Notify if a new identity has been created
PAM_CE_IDENTITY_DELETED	4	Notify if an identity has been deleted
PAM_CE_GROUP_MEMBERSHIP_CHANGED	5	Notify if the membership of a group changes.
PAM_CE_AGENT_CREATED	6	Notify if a new agent has been created
PAM_CE_AGENT_DELETED	7	Notify if an agent has been deleted
PAM_CE_AGENT_ASSIGNED	8	Notify if an agent is assigned to an identity
PAM_CE_AGENT_UNASSIGNED	9	Notify if an agent has been unassigned from an identity
PAM_CE_CAPABILITY_CHANGED	10	Notify if the capability of an identity changes
PAM_CE_AGENT_CAPABILITY_PRESENCE_SET	11	Notify if the value of presence attributes of an agent is explicitly set
PAM_CE_AGENT_PRESENCE_SET	12	Notify if the value of presence attributes of an agent is explicitly set

11.12.5 TpPAMEventNameList

This is a `Numbered List of Data Elements` of type TpPAMEventName.

Each event is defined by the data that applications must provide during registration using TpPAMEventInfo and data that is provided to the application during notification of such events using TpPAMNotificationInfo.

11.12.6 TpPAMEventInfo

This is a tagged choice of data elements that specifies the event data provided by applications while registering.

	Tag Element Type	
	TpPAMEventName	

Tag Element Value	Choice Element Type	Choice Element Name
PAM_CE_IDENTITY_PRESENCE_SET	TpPAMIPSEventData	IdentityPresenceSet
PAM_CE_AVAILABILITY_CHANGED	TpPAMAVCEventData	AvailabilityChanged
PAM_CE_WATCHERS_CHANGED	TpPAMWCEventData	WatchersChanged
PAM_CE_IDENTITY_CREATED	TpPAMICEventData	IdentityCreated
PAM_CE_IDENTITY_DELETED	TpPAMIDEventData	IdentityDeleted
PAM_CE_GROUP_MEMBERSHIP_CHANGED	TpPAMGMCEventData	GroupMembershipChanged
PAM_CE_AGENT_CREATED	TpPAMACEventData	AgentCreated
PAM_CE_AGENT_DELETED	TpPAMADEventData	AgentDeleted
PAM_CE_AGENT_ASSIGNED	TpPAMAAEventData	AgentAssigned
PAM_CE_AGENT_UNASSIGNED	TpPAMAUEventData	AgentUnassigned
PAM_CE_CAPABILITY_CHANGED	TpPAMCCEventData	CapabilityChanged
PAM_CE_AGENT_CAPABILITY_PRESENCE_SET	TpPAMACPSEventData	AgentCapabilityPresenceSet
PAM_CE_AGENT_PRESENCE_SET	TpPAMAPSEventData	AgentPresenceSet

11.12.7 TpPAMEventInfoList

This is a Numbered List of Data Elements of type TpPAMEventInfo.

11.12.8 TpPAMNotificationInfo

This is a tagged choice of data elements that specifies the notification data provided to the applications for each event.

	Tag Element Type	
	TpPAMEventName	

Tag Element Value	Choice Element Type	Choice Element Name
PAM_CE_IDENTITY_PRESENCE_SET	TpPAMIPSNotificationData	IdentityPresenceSetNotify
PAM_CE_AVAILABILITY_CHANGED	TpPAMAVCNotificationData	AvailabilityChangedNotify
PAM_CE_WATCHERS_CHANGED	TpPAMWCNotificationData	WatchersChangedNotify
PAM_CE_IDENTITY_CREATED	TpPAMICNotificationData	IdentityCreatedNotify
PAM_CE_IDENTITY_DELETED	TpPAMIDNotificationData	IdentityDeletedNotify
PAM_CE_GROUP_MEMBERSHIP_CHANGED	TpPAMGMCNotificationData	GroupMembershipChangedNotify
PAM_CE_AGENT_CREATED	TpPAMACNotificationData	AgentCreatedNotify
PAM_CE_AGENT_DELETED	TpPAMADNotificationData	AgentDeletedNotify
PAM_CE_AGENT_ASSIGNED	TpPAMAANotificationData	AgentAssignedNotify
PAM_CE_AGENT_UNASSIGNED	TpPAMAUNotificationData	AgentUnassignedNotify
PAM_CE_CAPABILITY_CHANGED	TpPAMCCNotificationData	CapabilityChangedNotify
PAM_CE_AGENT_CAPABILITY_PRESENCE_SET	TpPAMACPSNotificationData	AgentCapabilityPresenceSetNotify
PAM_CE_AGENT_PRESENCE_SET	TpPAMAPSNotificationData	AgentPresenceSetNotify

11.12.9 TpPAMNotificationInfoList

This is a Numbered List of Data Elements of type TpPAMNotificationInfo.

11.12.10 PAM_CE_IDENTITY_CREATED

Notify if a new identity has been created. Notifications for creation of multiple identities are bunched into a single notification whenever possible. A notification of this event is NOT sent for new association of types with an existing identity.

11.12.10.1TpPAMICEventData

This is a Sequence of Data Elements to specify the input data for subscribing to identity creations. The event is registered for changes in any agents of the named type. If no identity types are named, then the event is registered for all identity types.

Sequence Element Name	Sequence Element Type	Notes
IdentityType	TpStringList	Specifies the type of the identities for which this notification is requested. Can be an empty array if notification required for identities of any type

11.12.10.2TpPAMICNotificationData

This is a Sequence of Data Elements to specify the data that is provided in the notifications for identity creation events.

Sequence Element Name	Sequence Element Type	Notes
Identities	TpPAMFQNameList	Contains the names of the identities that have been created.

11.12.11 PAM_CE_IDENTITY_DELETED

Notify if an identity has been deleted. Notifications for deletion of multiple identities are bunched into a single notification whenever possible. A notification of this event is NOT sent for removing association of types with an existing identity.

11.12.11.1TpPAMIDEventData

This is a Sequence of Data Elements to specify the input data for subscribing to identity deletions. The event is registered for changes in any of the named identities. If no identities are named, then the event is registered for all agents.

Sequence Element Name	Sequence Element Type	Notes
IdentityName	TpPAMFQNameList	Specifies the name of the identity whose deletion is to be notified. Can be an empty array
IdentityType	TpStringList	Specifies the type of the identity for which this notification is requested if identityName is an empty array. Can be an empty array if notification required for identities of any type

11.12.11.2TpPAMIDNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for identity deletion events.

Sequence Element Name	Sequence Element Type	Notes
Identities	TpPAMFQNameList	Contains the names of the identities that have been deleted.

11.12.12 PAM_CE_GROUP_MEMBERSHIP_CHANGED

Notify if the membership of a group changes. Notifications for changes to multiple groups are bunched into a single notification whenever possible.

11.12.12.1TpPAMGMCEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to group membership changes. The event is registered for changes in any of the named groups. If no groups are named, then the event is registered for all groups.

Sequence Element Name	Sequence Element Type	Notes
GroupName	TpPAMFQNameList	Specifies the name of the group for which the change is to be notified. Can be an empty array if notifications are required for any group.
GroupType	TpStringList	Specifies the type of the group for which this notification is requested if the groupName is specified as an empty array. Can be an empty array if notification required for groups of any type.

11.12.12.2TpPAMGMCNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for group membership changes.

Sequence Element Name	Sequence Element Type	Notes
Groups	TpPAMFQNameList	Contains the names of the groups that have been changed.

11.12.13 PAM_CE_AGENT_CREATED

Notify if a new agent has been created. Notifications for creation of multiple agents are bunched into a single notification whenever possible. The notification for this event is NOT sent for new associations of types with agents.

11.12.13.1TpPAMACEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to agent creations. The event is registered for changes in any agents of the named type. If no agent types are named, then the event is registered for all agent types.

Sequence Element Name	Sequence Element Type	Notes
AgentType	TpStringList	Specifies the type of the agents for which this notification is requested. Can be an empty array if notification required for agents of any type

11.12.13.2TpPAMACNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for agent creation events.

Sequence Element Name	Sequence Element Type	Notes
Agents	TpPAMFQNameList	Contains the names of the agents that have been created.

11.12.14 PAM_CE_AGENT_DELETED

Notify if an agent has been deleted. Notifications for deletion of multiple agents are bunched into a single notification whenever possible. This event notification is NOT sent for disassociating a type from an agent.

11.12.14.1TpPAMADEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to agent deletions. The event is registered for changes in any of the named agents. If no agents are named, then the event is registered for all agents.

Sequence Element Name	Sequence Element Type	Notes
AgentName	TpPAMFQNameList	Specifies the name of the agent whose deletion is to be notified. Can be an empty array
AgentType	TpStringList	Specifies the type of the agent for which this notification is requested if agentName is an empty array. Can be an empty array if notification required for agents of any type

11.12.14.2TpPAMADNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for agent deletion events.

Sequence Element Name	Sequence Element Type	Notes
Agents	TpPAMFQNameList	Contains the names of the agents that have been deleted.

11.12.15 PAM_CE_AGENT_ASSIGNED

Notify if an agent is assigned to an identity.

11.12.15.1TpPAMAAEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to agent assignments from an identity. The event is registered for changes in any of the named agents. If no agents are named, then the event is registered for any agent.

Sequence Element Name	Sequence Element Type	Notes
IdentityName	TpPAMFQNameList	Specifies the name of the identity for which the assignment is to be notified. Can be an empty array if notification is required for any identity instance.
IdentityType	TpStringList	Specifies the type of the identity for which this notification is requested if the identityName is specified as an empty array. Can be an empty array if notification required for identities of any type.
AgentName	TpPAMFQNameList	Specifies the name of the agent whose assignment is to be notified. Can be an empty array
AgentType	TpStringList	Specifies the type of the agent for which this notification is requested if agentName is an empty array. Can be an empty array if notification required for agents of any type

11.12.15.2TpPAMAANotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for agent assignment events.

Sequence Element Name	Sequence Element Type	Notes
Identity	TpPAMFQName	Contains the name of the identity to whom an agent has been assigned.
Agent	TpPAMFQName	Contains the name of the agent that has been assigned.

11.12.16 PAM_CE_AGENT_UNASSIGNED

Notify if an agent has been unassigned from an identity.

11.12.16.1TpPAMAUEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to agent unassignments from an identity. The event is registered for changes in any of the named agents. If no agents are named, then the event is registered for all assigned agents.

Sequence Element Name	Sequence Element Type	Notes
IdentityName	TpPAMFQNameList	Specifies the name of the identity for which the unassignment is to be notified. Can be an empty array if notification is required for any identity instance.
IdentityType	TpStringList	Specifies the type of the identity for which this notification is requested if the identityName is specified as an empty array. Can be an empty array if notification required for identities of any type.
AgentName	TpPAMFQNameList	Specifies the name of the agent whose unassignment is to be notified. Can be an empty array
AgentType	TpStringList	Specifies the type of the agent for which this notification is requested if agentName is an empty array. Can be an empty array if notification required for agents of any type

11.12.16.2TpPAMAUNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for agent unassignment events.

Sequence Element Name	Sequence Element Type	Notes
Identity	TpPAMFQName	Contains the name of the identity from whom an agent has been unassigned.
Agent	TpPAMFQName	Contains the name of the agent that has been unassigned.

11.12.17 PAM_CE_CAPABILITY_CHANGED

Notify if the capability of an identity changes.

11.12.17.1TpPAMCCEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to identity capability changed events. The event is registered for changes in any of the named capabilities. If no capabilities are named, then the event is registered for all capabilities.

Sequence Element Name	Sequence Element Type	Notes
IdentityName	TpPAMFQNameList	Specifies the name of the identity for which the capability change is to be notified. Can be an empty array if notification is required for any identity instance
IdentityType	TpStringList	Specifies the type of the identity for which this notification is requested if the identityName is specified as an empty array. Can be an empty array if notification required for identities of any type.
Capabilities	TpPAMCapabilityList	Specifies the capabilities of interest. Can be an empty array if notifications are required for any capability.

11.12.17.2TpPAMCCNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for capability change events.

Sequence Element Name	Sequence Element Type	Notes
Identity	TpPAMFQName	Contains the name of the identity whose capability has changed.
Capabilities	TpPAMCapabilityList	Contains the capabilities that have changed (i.e., added or removed).

11.12.18 PAM_CE_AGENT_CAPABILITY_PRESENCE_SET

Notify if the value of capability presence attributes of an agent is set. Expiration of the dynamic attributes does not trigger this notification.

11.12.18.1TpPAMACPSEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to agent capability presence set events. The event is registered for changes in any of the named attributes. If no attributes are named, then the event is registered for all attributes in the presence information.

Sequence Element Name	Sequence Element Type	Notes
AgentName	TpPAMFQNameList	Specifies the name of the agent for which the capability presence change is to be notified. Can be an empty array if notification is required for any agent instance.
AgentType	TpStringList	Specifies the type of the agent for which this notification is requested if the agentName is specified as an empty array. Can be an empty array if notification required for agents of any type.
Capabilities	TpPAMCapabilityList	Specifies the capabilities of interest. Can be an empty array if notifications are required for any capability.
AttributeNames	TpStringList	Specifies attributes of interest. Can be an empty array
ReportingPeriod	TpPAMTimeInterval	Specifies the interval for periodic reporting (regardless of change). If -1, the event notification happens only on a change. If 0, there is a single immediate notification.

11.12.18.2TpPAMACPSNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for capability presence set events.

Sequence Element Name	Sequence Element Type	Notes
Agent	TpPAMFQName	Contains the name of the agent whose capability presence has changed
Capability	TpPAMCapability	Specifies the capability for which the presence has changed.
AttributeNames	TpStringList	Contains the attribute names that have changed in value

11.12.19 PAM_CE_AGENT_PRESENCE_SET

Notify if the value of presence attributes of an agent is set. Expiration of the dynamic attributes does not trigger this notification.

11.12.19.1TpPAMAPSEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to agent presence set events. The event is registered for changes in any of the named attributes. If no attributes are named, then the event is registered for all attributes in the presence information.

Sequence Element Name	Sequence Element Type	Notes
AgentName	TpPAMFQNameList	Specifies the name of the agent for which the assignment is to be notified. Can be an empty array if notification is required for any agent instance.
AgentType	TpStringList	specifies the type of the agent for which this notification is requested if the agentName is specified as an empty array. Can be an empty array if notification required for agents of any type.
AttributeNames	TpStringList	Specifies attributes of interest. Can be an empty array
ReportingPeriod	TpPAMTimeInterval	Specifies the interval for periodic reporting (regardless of change). If -1, the event notification happens only on a change. If 0, there is a single immediate notification.

11.12.19.2TpPAMAPSNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for agent presence set events.

Sequence Element Name	Sequence Element Type	Notes
Agent	TpPAMFQName	Contains the name of the agent whose capability has changed
AttributeNames	TpStringList	Contains the attribute names that have changed in value

11.12.20 PAM_CE_IDENTITY_PRESENCE_SET

Notify if the value of presence attributes of an identity is set. Expiration of the dynamic attributes do not trigger this notification.

11.12.20.1TpPAMIPSEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to identity presence set events. The event is registered for changes in any of the named attributes. If no attributes are named, then the event is registered for all attributes in the presence information.

Sequence Element Name	Sequence Element Type	Notes
IdentityName	TpPAMFQNameList	Specifies the name of the identity for which the assignment is to be notified. Can be an empty array if notification is required for any identity instance.
IdentityType	TpStringList	specifies the type of the identity for which this notification is requested if the identityName is specified as an empty array. Can be an empty array if notification required for identities of any type.
AttributeNames	TpStringList	Specifies attributes of interest. Can be an empty array
ReportingPeriod	TpPAMTimeInterval	Specifies the interval for periodic reporting (regardless of change). If -1, the event notification happens only on a change. If 0, there is a single immediate notification even if there is no change. For all other values, there is a periodic notification at the specified time interval regardless of change.

11.12.20.2TpPAMIPSNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for identity presence set events.

Sequence Element Name	Sequence Element Type	Notes
Identity	TpPAMFQName	Contains the name of the identity whose capability has changed
Attributes	TpPAMPresenceDataList	Contains the attributes that have changed in value

11.12.21 PAM_CE_AVAILABILITY_CHANGED

Notify if the availability of an identity changes. The event is registered for changes in any of the named attributes. If no attributes are named, then the event is registered for all attributes in the presence information.

11.12.21.1TpPAMAVCEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to availability changed events.

Sequence Element Name	Sequence Element Type	Notes
IdentityName	TpPAMFQNameList	Specifies the name of the identity for which the assignment is to be notified. Can be an empty array if notification is required for any identity instance.
IdentityType	TpStringList	specifies the type of the identity for which this notification is requested if the identityName is specified as an empty array. Can be an empty array if notification required for identities of any type.
PAMContext	TpPAMContextList	Specifies the context in which the availability is to be monitored. Cannot be an empty array
AttributeNames	TpStringList	Specifies attributes of interest. Can be an empty array
ReportingPeriod	TpPAMTimeInterval	Specifies the interval for periodic reporting (regardless of change). If -1, the event notification happens only on a change. If 0, there is a single immediate notification even if there is no change. For all other values, there is a periodic notification at the specified time interval regardless of change.

11.12.21.2TpPAMAVCNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for availability changed events.

Sequence Element Name	Sequence Element Type	Notes
Identity	TpPAMFQName	Contains the name of the identity whose capability has changed
Availability	TpPAMAvailabilityProfileList	Contains the availability information that has changed

11.12.22 PAM_CE_WATCHERS_CHANGED

Notify if list of watchers for any event changed.

11.12.22.1TpPAMWCEventData

This is a `Sequence of Data Elements` to specify the input data for subscribing to watchers changed events.

Sequence Element Name	Sequence Element Type	Notes
Events	TpPAMEventNameList	Specifies the list of events for which the watchers are to be monitored. Can be an empty array if notification is required for watchers to any event
IdentityName	TpPAMFQNameList	Specifies the name of an identity for whom the change in watchers is to be notified. Can be an empty array if notification is required for any identity instance.
IdentityType	TpStringList	specifies the type of the identity for which this notification is requested if the identityName is specified as an empty array. Can be an empty array if notification required for identities of any type.
ReportingPeriod	TpPAMTimeInterval	Specifies the interval for periodic reporting (regardless of change). If -1, the event notification happens only on a change. If 0, there is a single immediate notification even if there is no change. For all other values, there is a periodic notification at the specified time interval regardless of change.

11.12.22.2TpPAMWCNotificationData

This is a `Sequence of Data Elements` to specify the data that is provided in the notifications for watchers changed events.

Sequence Element Name	Sequence Element Type	Notes
Event	TpPAMEventName	Contains the name of the event for which the watchers changed
ChangeType	TpPAMwatcherChangeType	Specifies whether the listed watchers were added or deleted
Identity	TpPAMFQName	Contains the name of the identity whose capability has changed
Watchers	TpPAMFQNameList	Contains the list of watchers involved in the change

11.12.22.3TpPAMwatcherChangeType

This specifies the values representing the type of change that occurred to the list of watchers.

Name	Value	Description
PAM_WATCHERS_PERIODIC	0	Periodic reporting, not necessarily a change.
PAM_WATCHERS_ADDED	1	Watchers added to the list
PAM_WATCHERS_DELETED	2	Watchers deleted from the list

11.13 Error Types

11.13.1 TpPAMErrorCause

This defines the types of errors reported by PAM.

Name	Value	Description
P_PAM_CAUSE_UNDEFINED	0	Undefined.
P_PAM_CAUSE_INVALID_ADDRESS	1	The request cannot be handled because the address specified is not valid.
P_PAM_CAUSE_SYSTEM_FAILURE	2	System failure. The request cannot be handled because of a general problem in the service or in the underlying network.
P_PAM_CAUSE_INFO_UNAVAILABLE	3	The information is currently not available.
P_PAM_CAUSE_EVENT_REGISTRATION_CANCELLED	4	The registration for the event has been cancelled by the service.

11.13.2 TpPAMErrorInfo

This is a `Sequence of Data Elements` to specify the error notification data.

Sequence Element Name	Sequence Element Type	Description
Cause	TpPAMErrorCause	Contains information about the reason for the error
ErrorData	TpPAMNotificationInfo	Contains information relevant to each error such as the identity for which the error exists and/or the attributes for which the error exists

12 Presence and Availability Management Exception Classes

The following are the list of exception classes which are used in this interface of the API.

Name	Description
P_PAM_AGENT_EXISTS	indicates that an Agent with the *agentName* already exists
P_PAM_ALIAS_EXISTS	indicates that the specified alias is already associated to the Identity
P_PAM_ALIAS_NOT_UNIQUE	indicates that the alias has already been assigned to another identity
P_PAM_ATTRIBUTE_EXISTS	indicates that at least one of the named attributes already exists
P_PAM_DISASSOCIATED_TYPE	indicates that one of the specified types is not associated with the named identity/agent
P_PAM_IDENTITY_EXISTS	indicates that the specified Identity already exists
P_PAM_INVALID_CREDENTIAL	indicates that the credential presented is not recognized or insufficient for the operation
P_PAM_IS_CYCLIC	indicates that the requested operation will create cyclic relationship
P_PAM_MEMBER_EXISTS	indicates that the specified member is already in the group
P_PAM_NO_CAPABILITY	indicates that a supplied capability is not a capability of the requested agent. No attributes are affected
P_PAM_NOT_MEMBER	indicates that the specified member is not member of the group
P_PAM_NOT_REGISTERED	indicates that the interface was not previously registered
P_PAM_NOT_SUPPORTED	implementation dependent status that indicates that this method is not supported by the implementation
P_PAM_TYPE_ASSOCIATED	indicates that a named type has already been associated with the identity/agent
P_PAM_TYPE_EXISTS	indicates that the named type already exists
P_PAM_UNASSIGNED_ALIAS	indicates that the specified alias was not an alias of the named identity
P_PAM_UNAVAILABLE_INTERFACE	indicates that the specified interface does not exist or is unavailable
P_PAM_UNKNOWN_AGENT	indicates that the Agent with the specified name does not exist
P_PAM_UNKNOWN_ALIAS	indicates that the Alias with the specified name does not exist
P_PAM_UNKNOWN_ASSIGNMENT	indicates that no assignment exists for this identity and agent
P_PAM_UNKNOWN_ATTRIBUTE	indicates that at least one of the specified attributes has not been defined or has not been associated with the specified object
P_PAM_UNKNOWN_ATTRIBUTES	indicates that the specified attribute list contains attributes not part of the named object
P_PAM_UNKNOWN_CAPABILITY	indicates that a supplied capability is not a capability of the requested agent, or has not been defined. No attributes are affected
P_PAM_UNKNOWN_GROUP	indicates that the specified group identity does not exist
P_PAM_UNKNOWN_IDENTITY	indicates that the specified identity does not exist
P_PAM_UNKNOWN_MEMBER	indicates that the specified member identity does not exist
P_PAM_UNKNOWN_TYPE	indicates that the named type does not exist / indicates that the named identity/agent has not been associated with the named type / indicates that a specified type name has not been defined as an agent type

Each exception class contains the following structure:

Structure Element Name	Structure Element Type	Structure Element Description
ExtraInformation	TpString	Carries extra information to help identify the source of the exception, e.g. a parameter name

Annex A (normative): OMG IDL Description of Presence and Availability Management SCF

The OMG IDL representation of this interface specification is contained in text files (pam_interfaces, pam_data.idl contained in archive 2019814IDL.ZIP) which accompanies the present document.

Annex B (informative): Java API Description of the Presence and Availability Management SCFs

The Java API representation of this specification can be obtained from the following URL:

- JAIN Presence and Availability Management (http://jcp.org/jsr/detail/123.jsp)

Each JSR webpage contains a table identifying the relationships between the different versions of the Parlay, ETSI/OSA, 3GPP/OSA and JAIN SPA specifications. In addition, each JAIN SPA specification version indicates to which Parlay, ETSI/OSA and 3GPP/OSA specification versions it corresponds to.

Annex C (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
April 2002	--	--	--	--	Draft v100 submitted to TSG CN email list for Information		1.0.0
June 2002	CN_16	NP-020196	--	--	Draft v200 submitted to TSG CN#16 for Approval	2.0.0	5.0.0
Sep 2002	CN_17	NP-020440	001	--	Add text to clarify requirements on support of methods	5.0.0	5.1.0
Sep 2002	CN_17	NP-020440	002	--	Remove declaration of unused datatype TpPAMTime	5.0.0	5.1.0
Sep 2002	CN_17	NP-020395	003	--	Add text to clarify relationship between 3GPP and ETSI/Parlay OSA specifications	5.0.0	5.1.0
Jun 2003	CN_20	NP-030245	004	--	Make TpPAMCapability extensible by changing its type to TpString	5.1.0	5.2.0
Jun 2003	CN_20	NP-030240	005	--	Change the type of TpPAMFQName to TpURN	5.1.0	5.2.0
Jun 2003	CN_20	NP-030245	006	--	Clarifiy use of askerData parameter to getAuthToken method in each PAM SCF	5.1.0	5.2.0
Jun 2003	CN_20	NP-030245	007	--	Add authToken parameter to computeAvailability method	5.1.0	5.2.0
Jun 2003	CN_20	NP-030245	008	--	Replace use of IpInterfaceRef in PAM with actual application interfaces	5.1.0	5.2.0
Jun 2003	CN_20	NP-030245	009	--	Add expiration time for PAM event registrations	5.1.0	5.2.0
Jun 2003	CN_20	NP-030245	010	--	Send subscription notification cancellation to watchers	5.1.0	5.2.0
Jun 2003	CN_20	NP-030241	011	--	Change PAM Presence and Availability SCF name to PAM Access	5.1.0	5.2.0
		NP-030245	012	--	Move Access Control Mechanism to Manager Interface		

History

Document history		
V5.0.0	June 2002	Publication
V5.1.0	September 2002	Publication
V5.2.0	June 2003	Publication