



Sip Advanced Messaging, Usability, Routing and Accounting Interface *

Specification Version 1.06

indigo networks GmbH

August 21, 2007

*SAMURAI; The word 'samurai' is derived from the Japanese verb 'sabur', meaning 'to serve'.

This document describes a system for transportation of configuration and auxiliary services data between an end user's CPE and the service provider's SIP system. The CPE may download configuration data for automatic setup for the demanded service as well as setting options on the provider's system to adapt the service to individual needs. Examples for auxiliary services data may be *unified messaging*¹, history listings, account statements, server-side phonebook, etc.

¹Unified Messaging (or UM) is the integration of different streams of messages (email, Fax, voice, video, etc.) into a single in-box, accessible from a variety of different devices. [1]

Contents

1	Introduction	5
1.1	Technical	5
1.2	Terminology	5
2	RPC Transport	6
2.1	HTTPS	6
3	RPC Methods	7
3.1	RPC Method Usage	7
3.1.1	Data Types	7
3.1.2	Other Requirements	7
3.1.3	comments to specification	8
3.2	Mandatory Server Methods	8
3.2.1	system.listMethods	8
3.2.2	system.methodHelp	8
3.2.3	system.methodSignature	9
3.2.4	system.serverInfo	10
3.3	Optional Server Methods	10
3.3.1	samurai.AccountStatementGet	10
3.3.2	samurai.BalanceGet	13
3.3.3	samurai.ClientIdentify	13
3.3.4	samurai.HistoryGetByDate	14
3.3.5	samurai.ItemizedEntriesGet	15
3.3.6	samurai.OwnUriListGet	18
3.3.7	samurai.PhonebookEntryGet	19
3.3.8	samurai.PhonebookListGet	20
3.3.9	samurai.RecommendedIntervalGet	21
3.3.10	samurai.ServerdataGet	22
3.3.11	samurai.SessionClose	23
3.3.12	samurai.SessionInitiate	24
3.3.13	samurai.SessionInitiateMulti	25
3.3.14	samurai.SessionStatusGet	26
3.3.15	samurai.TosListGet	26

3.3.16 samurai.UmSummaryGet	27
3.3.17 samurai.UserdataGreetingGet	28
3.3.18 samurai.UserdataSipGet	28
3.4 Vendor Specific Extensions	29
A Server Status Codes	30
B TOS (type of service) specification	32
C XMLRPC message examples	33
D SAMURAI specification change log	35

1 Introduction

1.1 Technical

There are two main groups of target devices the interface is intended to be used by: Customizable devices (like PCs, PDAs, etc) and highly embedded devices (SIP Phones). Communication is done via remote procedure calls (RPC) transported via session layer protocols as e.g. HTTP. Security, such as authentication and encryption, is delegated to those session layer protocols. Transmitted data includes sensitive private data, so a choice of appropriate security mechanisms is highly recommended.

1.2 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119[2].

2 RPC Transport

The service specified by this document may be used via, but is not limited to, the following underlying layers:

2.1 HTTPS

Implements the *XML-RPC specification*[3] where remote procedure calls are encoded in XML¹ and sent in the body of HTTP messages as *text/xml* mimetype. To ensure privacy of user data HTTP traffic is secured by using SSL/TLS[4]. Authentication to the service is accomplished by using *HTTP Basic Authentication*[5].

¹Examples of encoding are given in appendix C.

3 RPC Methods

3.1 RPC Method Usage

3.1.1 Data Types

Definition of data types used in this specification.

<i>Type</i>	<i>Description</i>
array	Single- or multidimensional list of data values.
boolean	Boolean value, where 1 = true and 0 = false.
dateTime	Date and time formatted corresponding to RFC 3339[6], e.g. "2005-12-30T14:32:57+01:00".
float	Floating-point data with a maximum precision of two decimal places. This specification doesn't provide for exponential notation.
int	Integer in the range -2147483648 to +2147483647, inclusive.
string	For strings listed in this specification, a maximum allowed length may be listed using the form string(N), where N is the maximum string length in characters and may be "-" to indicate that the size is not limited by this specification. For all strings a maximum length is either explicitly indicated or implied by the size of the elements composing the string. If a string does not have an explicitly indicated maximum length, the default maximum is 256 characters. Strings longer than the specified maximum MAY result in an "parameter exceeds maximum size" error response.

Table 3.1: Data Types

3.1.2 Other Requirements

All methods must be called using the exact number of arguments specified in this document. Methods called with either missing arguments or extra arguments will generate an error response. Argument order must be as specified in this document.

Future versions of this specification should not redefine the RPC methods defined in this document unless the changes are fully compatible to all prior specifications. Any changes needed in a future version should result in new RPC methods with distinct names being defined.

3.1.3 comments to specification

Argument names may be wrapped to increase readability of tables. In these cases the wrapping character ”-” is not part of the argument name.

3.2 Mandatory Server Methods

The methods listed in this section **MUST** be supported by the server. To allow for a maximum of flexibility, only a minimum set of methods is mandatory.

3.2.1 system.listMethods

Request a list of the RPC methods offered by the server with each entry being the name of a method stored as string value.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.2: system.listMethods calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
listMethods	array of string	List of local method names as strings.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.3: system.listMethods response arguments

3.2.2 system.methodHelp

Request a human readable description of the given method containing information about purpose and usage.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
MethodName	string	Name of the method (to which help is demanded) as string type value.

Table 3.4: system.methodHelp calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
methodHelp	string	Help text or description to the specified mehod.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.5: system.methodHelp response arguments

3.2.3 system.methodSignature

Request the signature of the given RPC method. The server will return a list of signatures valid for the specified method. The signature is defined as the promised return value's type followed by the types of any arguments. A method MAY be called in more than one way, which is what the purpose of defining signatures is meant to distinguish.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
MethodName	string	Name of the Method (to which a signature is demanded) as string type value.

Table 3.6: system.methodSignature calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
methodSignature	array of array of string	List of signatures valid for the method.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.7: system.methodSignature response arguments

3.2.4 system.serverInfo

The *system.serverInfo* method provides specific information on the server software.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.8: system.serverInfo calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SpecificationVersion	string	The version number of the implemented SAMURAI specification.
ServerName	string	The name of the server software.
ServerVersion	string	The version number of the server software.
ServerVendor	string	The name of the server software vendor (e.g. company or organization name).
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.9: system.serverInfo response arguments

3.3 Optional Server Methods

The methods listed in this section MAY be supported by server and client so the client MUST discover (eg. see 3.2.1) the methods available on the server after login.

3.3.1 samurai.AccountStatementGet

Request an account statement for a specified period. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 500, 511

<i>Argument</i>	<i>Type</i>	<i>Description</i>
PeriodStart	dateTime	Start date and time of demanded period.
PeriodEnd	dateTime	End date and time of demanded period.

Table 3.10: samurai.AccountStatementGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
PeriodStart	dateTime	Start date and time of demanded period.
PeriodEnd	dateTime	End date and time of demanded period.
AccountStatement-ChargedServices	array of struct	List of structures (see table 3.12) listing demanded service and corresponding charges.
BalanceStart	pricetype	Structure as defined in table 3.13, containing the account balance at the start of the given period.
BalanceEnd	pricetype	Structure as defined in table 3.13, containing the account balance at the end of the given period.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.11: samurai.AccountStatementGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
Timestamp	dateTime	Date and time the transaction was made. (This will not be set in cases where uses of a service are grouped.) <i>optional</i>
ServiceName	string	e.g. "local call"
Quantity	int	Quantity of taken services of type <i>ServiceName</i> .
TotalDuration	int	Total duration of sessions of type <i>ServiceName</i> in seconds. (This information may not be available in all cases.) <i>optional</i>
Price	pricetype	Structure as defined in table 3.13, containing all pricing information including values in- and excluding tax and tax information.

Table 3.12: *AccountStatementChargedServices* struct specification

<i>Argument</i>	<i>Type</i>	<i>Description</i>
TotalExcludingVat	float	Balance or total charge excluding VAT for the services demanded during the specified period. (This information may not be available in all cases, but MUST be if <i>TotalIncludingVat</i> is not available.) <i>optional</i>
TotalIncludingVat	float	Balance or total charge including VAT for the services demanded during the specified period. (This information may not be available in all cases, but MUST be if <i>TotalExcludingVat</i> is not available.) <i>optional</i>
VatAmount	float	Payable VAT as value in <i>currency</i> . (This information may not be available in all cases.) <i>optional</i>
VatPercent	float	Tax percentage payable (eg. 16% tax as "16.00").
Currency	string	Currency code as defined by the ISO 4217 standard[7] (e.g. "USD" or "EUR").

Table 3.13: *pricetype* struct specification

3.3.2 samurai.BalanceGet

Request the current balance of the user's account.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.14: samurai.BalanceGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
BalanceTime	dateTime	Date and time the demanded balance is valid at.
CurrentBalance	pricetype	Amount of money in <i>Currency</i> that is available at user's account at <i>BalanceTime</i> (structure <i>pricetype</i> is defined in table 3.13).
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.15: samurai.BalanceGet response arguments

3.3.3 samurai.ClientIdentify

The client SHOULD identify itself to the server including version and vendor information to allow for better support in cases of problems. It is recommended to call this method once during startup of the client.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
ClientName	string	Name of the client. <i>optional</i>
ClientVersion	string	Version of the client. <i>optional</i>
ClientVendor	string	Vendor of the client. <i>optional</i>

Table 3.16: samurai.ClientIdentify calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.17: samurai.ClientIdentify response arguments

3.3.4 samurai.HistoryGetByDate

Request a history listing for a given list of local SIP URI. If no local SIP URI is specified, a listing of all sessions of all local SIP URIs is returned. The listing may also be filtered by given status (e.g. "outgoing" to get a history listing of all outgoing sessions). If no status is specified, the history listing will include sessions of all status types. Standard session status values are "outgoing", "accepted" and "missed". Vendor specific status values MAY be defined but MUST have a "X-" prefix to prevent collisions with future extensions of this specification. Additionally the listing MAY be limited by specifying start and / or end date and time. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 501, 506, 509

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LocalUriList	array of string	List of local SIP URI that the history listing is demanded for. (May be empty for matching all local SIP URIs.) <i>optional</i>
StatusList	array of string	List of status that the history listing is to be filtered by (e.g. "outgoing"; may be empty for complete listing). <i>optional</i>
PeriodStart	dateTime	Start date and time for demanded listing. <i>optional</i>
PeriodEnd	dateTime	End date and time for demanded listing. <i>optional</i>

Table 3.18: samurai.HistoryGetByDate calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
History	array of struct	List of structures of type <i>HistoryEntry</i> (see table 3.20) that each carry information of a session.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.19: samurai.HistoryGetByDate response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
EntryID	string	Unique identifier for the history entry.
Timestamp	dateTime	Date and Time of session start.
TOS	string	Describing the type of service used. Types of service are defined in appendix B.
LocalUri	string	Local SIP URI of the session.
RemoteUri	string	Remote SIP URI of the session.
Status	string	Describing if the session was either "accepted", "missed" or "outgoing". Vendor specific status values may be defined, but MUST have "X-" as prefix.

Table 3.20: *HistoryEntry* struct specification

3.3.5 samurai.ItemizedEntriesGet

Request a listing of itemized entries of a specified period of time and specified local SIP URI(s). If no local SIP URI is specified, the listing will include data of all local SIP URIs. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 500, 501, 506, 511

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LocalUriList	array of string	List of local SIP URI that the listing is demanded for. <i>optional</i>
PeriodStart	dateTime	Start date and time of demanded period.
PeriodEnd	dateTime	End date and time of demanded period.

Table 3.21: samurai.ItemizedEntriesGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
PeriodStart	dateTime	Start date and time of demanded period.
PeriodEnd	dateTime	End date and time of demanded period.
ItemizedEntries	array of struct	List of structures of type <i>ItemizedEntry</i> (see table 3.23) that each carry information of a session.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.22: samurai.ItemizedEntriesGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
Timestamp	dateTime	Date and Time of session start.
SourceUri	string	SIP URI the session was originating from.
TargetUri	string	SIP URI the session was made to. <i>optional</i>
Price	pricetype	Structure as defined in table 3.13, containing the pricing information on costs incurred including values in- and excluding tax and tax information.
SetupFee	pricetype	Structure as defined in table 3.13, containing pricing information on setup fee including values in- and excluding tax and tax information. (This information may not be available if no setup fee was incurred.) <i>optional</i>
PricePerUnit	pricetype	Structure as defined in table 3.13, containing pricing information per unit including values in- and excluding tax and tax information. (This information may not be available in all cases.) <i>optional</i>
TicksA	int	Duration of the first unit in seconds. (This information may not be available in all cases.) <i>optional</i>
TicksB	int	Duration of units after the first unit in seconds. (This information may not be available in all cases.) <i>optional</i>
UnitsCharged	int	Seconds/pages/messages (depending on type of service) charged. (This information may not be available in all cases.) <i>optional</i>
TariffName	string	e.g. "local call"
Duration	int	Duration of the session in seconds. (This information may not be available in all cases.) <i>optional</i>
TOS	string	Describing the service used. Types of service are defined in appendix B.

Table 3.23: *ItemizedEntry* struct specification

3.3.6 samurai.OwnUriListGet

Request the list of SIP URIs belonging to the user's account.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.24: samurai.OwnUriListGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
OwnUriList	array of struct	One or more structures of type <i>OwnUri</i> (see table 3.26) that each describe a SIP URI associated to the user's account.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.25: samurai.OwnUriListGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SipUri	string	SIP URI associated to the user's account.
E164Out	string	<i>E.164</i> , <i>The international public telecommunication numbering plan</i> conforming number associated with <i>SipUri</i> set for outgoing sessions. (This information may not be available in all cases.) <i>optional</i>
E164In	array of string	List of <i>E.164</i> , <i>The international public telecommunication numbering plan</i> conforming numbers associated with <i>SipUri</i> for incoming sessions. (This information may not be available in all cases.) <i>optional</i>
TOS	array of string	List of service types for <i>SipUri</i> . Types of service are defined in appendix B.
DefaultUri	boolean	MUST be set once (but only once) per "OwnUriList" and (!) "TypeOfService"; <i>DefaultUri</i> will be used for CLIP.
UriAlias	string	Alias name defined for <i>SipUri</i> . (This information may not be available in all cases.) <i>optional</i>

Table 3.26: *OwnUri* struct specification

3.3.7 samurai.PhonebookEntryGet

Request one or more specified entries from the phonebook stored on the server. A hash on the phonebook-entries is provided by the server to allow the client to keep track of changes. Since hash-computation is done serverside, it is up to the server implementor to choose an adequate hash function. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 510

<i>Argument</i>	<i>Type</i>	<i>Description</i>
EntryIDList	array of string	List of unique identifiers of the phonebook entries that are demanded.

Table 3.27: samurai.PhonebookEntryGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
EntryList	array of struct	One or more structures of type <i>PhonebookEntry</i> (see table 3.29) that each describe a phonebook entry stored on the server.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.28: samurai.PhonebookEntryGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
EntryID	string	Unique identifier of the phonebook entry that is demanded.
EntryHash	string	Hashvalue on the phonebookentry (cp. 3.3.7 <i>samurai.PhonebookEntryGet</i>).
Entry	string(-)	"vCard - The Electronic Business Card - Version 2.1" [8]. Note that this document does not specify a maximum size of the string containing vCARD data.

Table 3.29: *PhonebookEntry* struct specification

3.3.8 samurai.PhonebookListGet

Request the list of all EntryIDs and corresponding hash-values of the serverside phonebook.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.30: samurai.PhonebookListGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
PhonebookList	array of struct	List of structures of type <i>PhonebookListEntry</i> (see table 3.32) that each stores ID and hash-value of one phonebook entry.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.31: samurai.PhonebookListGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
EntryID	string	Unique identifier referring to an entry of the user's phonebook on the server.
EntryHash	string	Hashvalue on the phonebookentry (cp. 3.3.7 <i>samurai.PhonebookEntryGet</i>).

Table 3.32: *PhonebookListEntry* struct specification

3.3.9 samurai.RecommendedIntervalGet

Request the time in seconds the service provider recommends as minimum interval for calling a specific method. One or more method names may be specified in a list. The client SHOULD make use of this method because calling a method more often than recommended MAY cause the server to block the request and return statuscode 412 (cp. table A.2 in appendix A).

<i>Argument</i>	<i>Type</i>	<i>Description</i>
MethodList	array of string	List of method names to request the minimum recommended calling interval for.

Table 3.33: samurai.RecommendedIntervalGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
IntervalList	array of struct	Structure as defined in table 3.35, containing method name and recommended minimum interval.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.34: samurai.RecommendedIntervalGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
MethodName	string	Name of server method.
RecommendedInterval	int	Minimum interval in seconds the provider recommends for calling method <i>MethodName</i> .

Table 3.35: *RecommendedInterval* struct specification

3.3.10 samurai.ServerdataGet

Request serverdata required for automatical setup of a client for SIP communication including SIP registrar server, SIP outbound proxy, STUN¹ server, NTP² server, HTTP server, SAMURAI server, SIMPLE³ server and XMPP⁴ server. Fields that do not apply to the service provider's system MAY be left blank.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.36: samurai.ServerdataGet calling arguments

¹Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs); RFC 3489[9]

²Network Time Protocol

³Sip for Instant Messaging and Presence Leveraging Extensions (IETF working group)

⁴Extensible Messaging and Presence Protocol; see <http://www.xmpp.org/>

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SipRegistrar	string	Hostname or IP of service provider's SIP registrar server.
SipOutboundProxy	string	Hostname or IP of service provider's SIP outbound proxy server.
StunServer	string	Hostname or IP of service provider's STUN server.
NtpServer	string	Hostname or IP of service provider's NTP server.
HttpServer	string	Hostname or IP of service provider's HTTP server.
SamuraiServer	string	Hostname or IP of service provider's SAMURAI server.
SimpleServer	string	Hostname or IP of service provider's SIMPLE server.
XmppServer	string	Hostname or IP of service provider's XMPP server.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.37: samurai.ServerdataGet response arguments

3.3.11 samurai.SessionClose

Close a specified session created by *samurai.SessionInitiate* (see 3.3.12). This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 510, 512

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SessionID	string	Unique identifier associated to the session that is to be closed.

Table 3.38: samurai.SessionClose calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.39: samurai.SessionClose response arguments

3.3.12 samurai.SessionInitiate

Setup a connection between the user's own sip-client (e.g. IP-Phone) and a given remote SIP URI; commonly known as "click2dial". This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 501, 502, 506, 520, 525

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LocalUri	string	Local SIP URI that the session originates from.
RemoteUri	string	Remote SIP URI that is to be called.
TOS	string	Type of service for the demanded session. Types of service are defined in appendix B.
Content	string(-)	Content to be transmitted. This field only applies to specific types of content, e.g. "text" and is empty otherwise (e.g. for "voice").
Schedule	dateTime	Date and time the session is to be initiated. If <i>Schedule</i> is not set the session is initiated immediately. <i>optional</i>

Table 3.40: samurai.SessionInitiate calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SessionID	string	Unique identifier associated to the created session.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.41: samurai.SessionInitiate response arguments

3.3.13 samurai.SessionInitiateMulti

Setup a connection between the user's own sip-client (e.g. IP-Phone) and a list of remote SIP URI. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 501, 502, 506, 520, 525

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LocalUri	string	Local SIP URI that the session originates from.
RemoteUri	array of string	List of remote SIP URI that sessions are to be established to.
TOS	string	Type of service for the demanded sessions. Types of service are defined in appendix B.
Content	string(-)	Content to be transmitted. This field only applies to specific types of content, e.g. "text" and is empty otherwise (e.g. for "voice").
Schedule	dateTime	Date and time the sessions are to be initiated. If <i>Schedule</i> is not set the sessions are initiated immediately. <i>optional</i>

Table 3.42: samurai.SessionInitiateMulti calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SessionList	array of struct	Array of structures (see table 3.44) listing the IDs linked to the corresponding remote URI.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.43: samurai.SessionInitiateMulti response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SessionID	string	Unique identifier associated to the created session.
RemoteUri	string	Remote SIP URI of the initiated session.

Table 3.44: *SessionListEntry* struct specification

3.3.14 samurai.SessionStatusGet

Request status information for a specified session created by *samurai.SessionInitiate* (see 3.3.12). This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 510, 512

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SessionID	string	Unique identifier associated to the session that status information is requested for.

Table 3.45: samurai.SessionStatusGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SessionStatus	string	Human readable status information for the referenced session.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.46: samurai.SessionStatusGet response arguments

3.3.15 samurai.TosListGet

Request a listing of all available types of service (TOS). TOS are defined in appendix B.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.47: samurai.TosListGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
TosList	array of string	List of available TOS.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.48: samurai.TosListGet response arguments

3.3.16 samurai.UmSummaryGet

Request a summary of read and unread messages stored in the user's unified messaging (UM) box. A list of labels and / or TOS and / or local URIs may be given for filtering. If no parameters are specified, a single summary, including all messages stored in the user's UM box is returned. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A): 519, 526

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LabelName	array of string	None, one or more label names. <i>optional</i>
TOS	array of string	None, one or more TOS. <i>optional</i>
LocalUriList	array of string	None, one or more local URIs. <i>optional</i>

Table 3.49: samurai.UmSummaryGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
UmSummary	array of struct	List of one or more structures as specified in table 3.51.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.50: samurai.UmSummaryGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
Read	int	Sum of all messages with status "read" that are stored in the user's UM-box and apply to the set filter parameters (see method description).
Unread	int	Sum of all messages with status "unread" that are stored in the user's UM-box and apply to the set filter parameters (see method description).
LabelName	string	Name of the label read/unread-information is valid for. <i>optional</i>
TOS	string	TOS read/unread-information is valid for. <i>optional</i>

Table 3.51: UmSummary struct specification

3.3.17 samurai.UserdataGreetingGet

Request data from the server to be used for personalized greetings comprehending gender, first and last name.

<i>Argument</i>	<i>Type</i>	<i>Description</i>
none	void	This method has no calling arguments.

Table 3.52: samurai.UserdataGreetingGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
Gender	string	Gender stored in the user's account.
FirstName	string	First name stored in the user's account.
LastName	string	Last name stored in the user's account.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.53: samurai.UserdataGreetingGet response arguments

3.3.18 samurai.UserdataSipGet

Request the SIP userdata for authentication to the SIP server. This method MAY return the following *method specific server status codes* (cp. table A.2 in appendix A):

501, 506, 527

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LocalUriList	array of string	List of local SIP URI that SIP userdata is demanded for.

Table 3.54: samurai.UserdataSipGet calling arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
SipDataList	array of struct	List of one or more structures as specified in table 3.56.
StatusCode	int	Numeric status code as defined in appendix A.
StatusString	string	Human readable description of <i>StatusCode</i> .

Table 3.55: samurai.UserdataSipGet response arguments

<i>Argument</i>	<i>Type</i>	<i>Description</i>
LocalUri	string	Local SIP URI that SIP userdata is demanded for.
SipUserID	string	Unique identifier for authentication to the SIP-server.
SipPassword	string	Authentication password for authentication to the SIP-server.

Table 3.56: *UserdataSip* struct specification

3.4 Vendor Specific Extensions

Vendor specific methods MAY be added, but MUST have "X-" as name prefix to prevent collisions with future extensions of this specification.

A Server Status Codes

The *general server status codes* listed in table A.1 may be returned by any method called and thus MUST be understood by any client. *Method specific server status codes* shown in table A.2 MUST be understood by any client that implements the appropriate methods. Applying *method specific server status codes* are referenced to in the respective method descriptions above.

<i>Status Code</i>	<i>Description</i>
200	Method success
400	Method not supported
401	Request denied (no reason specified)
402	Internal error
403	Invalid arguments
404	Resources exceeded (this MUST not be used to indicate parameters in error)
405	Invalid parameter name
406	Invalid parameter type
407	Invalid parameter value
408	Attempt to set a non-writable parameter
409	Notification request rejected.
410	Parameter exceeds maximum size.
411	Missing parameter.
412	Too many requests.

Table A.1: general server status codes

<i>Status Code</i>	<i>Description</i>
500	Date out of range.
501	Uri does not belong to user.
502	Unknown type of service.
503	Selected payment method failed.
504	Selected currency not supported.
505	Amount exceeds limit.
506	Malformed SIP URI.
507	URI not in list.
508	Format is not valid E.164.
509	Unknown status.
510	Unknown ID.
511	Invalid timevalue.
512	Referenced session not found.
513	Only single default per TOS allowed.
514	Malformed VCARD format.
515	Malformed PID format.
516	Presence information not available.
517	Invalid label name.
518	Label not assigned.
519	Label doesn't exist.
520	Parameter includes invalid characters.
521	Bad password. (Rejected due to security concerns.)
522	Malformed timezone format.
523	Delay exceeds limit.
524	Requested VPN type not available.
525	Requested TOS not available.
526	Unified messaging not available.
527	URI not available for registration.

Table A.2: method specific server status codes

<i>Status Code</i>	<i>Description</i>
900 - 999	Vendor defined status codes

Table A.3: server status codes of vendor specific extensions

B TOS (type of service) specification

The provider may offer different services as for example voice- and video-communication to it's customers. The different services demand for different treatment in terms of configuration and billing, as well as in terms of presentation to the customer. To take these constraints into account, a *type of service* (TOS) identifier is defined as table B.1 shows. Additional to specifying the types of service, the measuring unit for messages of each type is defined, e.g. the "interpreted size" of a fax-message is measured in "pages" and the "interpreted size" of a voice-message is measured in "seconds". Vendor specific types of service may be used, but MUST have "X-" as the identifier's prefix as well as a defined measuring unit for belonging messages.

<i>identifier</i>	<i>units</i>	<i>comment</i>
fax	pages	fax transmission
text	characters	text message (e.g. "SMS")
video	seconds	video communication
voice	seconds	voice communication

Table B.1: *type of service* specification

C XMLRPC message examples

```
<?xml version="1.0" encoding="UTF-8"?>
<methodCall>
  <methodName>samurai.AccountStatementGet</methodName>
  <params>
    <param>
      <value><struct>
        <member>
          <name>PeriodEnd</name>
          <value><string>
            2007-07-31T00:00:00
          </string></value>
        </member>
        <member>
          <name>PeriodStart</name>
          <value><string>
            2007-06-01T00:00:00
          </string></value>
        </member>
      </struct></value>
    </param>
  </params>
</methodCall>
```

Listing C.1: XMLRPC method call to *samurai.AccountStatementGet*

```

<?xml version="1.0"?>
<methodResponse>
  <params><param>
    <value><struct>
      <member><name>StatusCode</name><value><i4>200</i4></value></member>
      <member>
        <name>PeriodEnd</name><value><string>2007-07-31T00:00:00+0200</string></value>
      </member>
      <member>
        <name>BalanceEnd</name>
        <value><struct>
          <member><name>Currency</name><value><string>EUR</string></value></member>
          <member><name>TotalIncludingVat</name><value><double>2.7204</double></value></member>
        </struct></value>
      </member>
      <member>
        <name>AccountStatementChargedServices</name>
        <value><array><data>
          <value><struct>
            <member>
              <name>ServiceName</name><value><string>Festnetz Deutschland</string></value>
            </member>
            <member>
              <name>Price</name>
              <value><struct>
                <member>
                  <name>Currency</name><value><string>EUR</string></value>
                </member>
                <member>
                  <name>TotalIncludingVat</name><value><double>0.0179</double></value>
                </member>
                <member>
                  <name>VatPercent</name><value><double>0.19</double></value>
                </member>
              </struct></value>
            </member>
            <member><name>Quantity</name><value><i4>1</i4></value></member>
          </struct></value>
          <value><struct>
            <member>
              <name>ServiceName</name><value><string>germany 01801</string></value>
            </member>
            <member>
              <name>Price</name>
              <value><struct>
                <member>
                  <name>Currency</name><value><string>EUR</string></value>
                </member>
                <member>
                  <name>TotalIncludingVat</name><value><double>0.0390</double></value>
                </member>
                <member>
                  <name>VatPercent</name><value><double>0.19</double></value>
                </member>
              </struct></value>
            </member>
            <member><name>Quantity</name><value><i4>1</i4></value></member>
          </struct></value>
        </data></array></value>
      </member>
      <member>
        <name>PeriodStart</name><value><string>2007-06-01T00:00:00+0200</string></value>
      </member>
      <member>
        <name>StatusString</name><value><string>Method success</string></value>
      </member>
      <member>
        <name>BalanceStart</name>
        <value><struct>
          <member><name>Currency</name><value><string>EUR</string></value></member>
          <member><name>TotalIncludingVat</name><value><double>2.7773</double></value></member>
        </struct></value>
      </member>
    </struct></value>
  </param></params>
</methodResponse>

```

Listing C.2: XMLRPC method response of *samurai.AccountStatementGet*

D SAMURAI specification change log

Version 1.06 - released August, 2007 *official release*

Bibliography

- [1] Wikipedia. Unified messaging — wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Unified_messaging&oldid=48957058, 2006. [21-May-2006].
- [2] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. RFC 2119 (Best Current Practice), March 1997.
- [3] D. Winer. Xml-rpc specification. <http://www.xmlrpc.com/spec>, June 1999.
- [4] E. Rescorla. HTTP Over TLS. RFC 2818 (Informational), May 2000.
- [5] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. HTTP Authentication: Basic and Digest Access Authentication. RFC 2617 (Draft Standard), June 1999.
- [6] G. Klyne and C. Newman. Date and Time on the Internet: Timestamps. RFC 3339 (Proposed Standard), July 2002.
- [7] International Organization for Standardization. Codes for the representation of currencies and funds. ISO 4217, 1995.
- [8] versit Consortium Specification. vcard - the electronic business card - version 2.1, sep 1996.
- [9] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy. STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). RFC 3489 (Proposed Standard), March 2003.