

A SYSTEM FOR COMMUNICATING BETWEEN A SERVICE SERVER AND SERVICE CLIENTS

The present invention relates to communications between an independently developed service server and one or more independently developed service clients in a public network, and in particular to a method, apparatus and system for defining and publishing services of a service server, for requesting and obtaining the results of a service request from a service client and for a service server to independently notify service clients of an event.

Background

A number of conventional host computers and terminal configurations are illustrated in Fig. 1A and Fig. 1B where each host computer communicates with 1 or more terminals. Online access to host computer application software began with the use of terminals 112 having keyboards (not shown) and display monitors. The terminals 112 are installed with predefined fixed logic for functioning of the terminal devices in tandem with messages to and from a host processing computer 110. The terminal computers 112 are used as input/output terminals of the host computer 110.

Examples of relevant host computers include UNIX-based computers and mainframes. Such systems require that messages between a host computer 120, 130 and the access or remote terminals 122, 132 conform to predefined content formats and flows (e.g., ASCII keystrokes and IBM 3270 datastream and flows, respectively). The host computer 120, 130 receives input from the keyboard of a terminal 122, 132 and sends output to the display monitor of the terminal 122, 132.

Some business environments also require terminals 142, 152 to provide use of other devices, e.g. cash dispensers, passbook printers, etc. For example, a bank mainframe 140 can be connected to an automatic teller machine (ATM) 142 that has non-standard devices such as a cash dispenser. Traditionally, such non-standard or special devices are directly controlled by software installed in the controller computer. The software is programmed in relation to the direct interface provided by the special device. In a network environment, the device controller software typically acts as a server to client software. In this case, communications between the mainframe 140 and the ATM 142 may be by way of IBM 3624, IBM 473X, Diebold 912, or NCR ATM message formats and flows, for example. Alternatively, a bank controller 150 may be connected to a branch-teller personal computer (PC) that has another non-standard device such as a passbook printer 152. This configuration also

requires software specific message formats and flows. Thus, special message content formats and flows are defined for these terminals 142, 152.

In such cases, in addition to input from terminal keyboards, the host computer 140, 150 may receive input from any of a number of non-standard devices such as a passbook printer, card reader, etc. Likewise, in addition to the display device of a terminal, the host computer 140, 150 sends output to non-standard devices such as a receipt printer, cash dispenser, etc. Thus, a large number of possible terminal devices are commonly utilised, each often requiring very specific datastreams and flows.

Still further, such configurations of host computers and terminals include WWW servers for the Internet and Intranets and browser computers, where the web server can be accessed by PCs using various browser applications. The browser PCs serve as input/output terminals for the web server via http messages.

In essence, the message content formats and flows of the exemplary systems of Fig. 1A and Fig. 1B are defined to enable a host computer application to receive input from and send output to the devices in the terminals 122, 132, 142, 152, and 162. The message formats and flows are therefore very specific to the particular selection of devices of each terminal and each device's characteristics. For example, a host computer may send a message that has a field containing magnetic track data that is to be encoded on a magnetic stripe of a bank card or the like, which had earlier been inserted into a non-standard device of an ATM.

Existing message formats and flows of such host computers and terminals often have to be extended to support new devices, when such new devices become available for use in terminals. Thus, the message formats and flows are not only varied and complex, but often must be adapted for new terminal devices.

Also, message formats may include fields to authenticate a human user and the source of messages. For example, the fields may require an encrypted personal identification number (PIN) and a message authentication digest, respectively. Such fields in the message format may be necessary where online access is sensitive so as to protect against illegal tapping and tampering of messages.

For sharing of terminal computers across organisations, messages can be defined for inter-organisational host computers such as those shown in Fig. 2. These messages are also specific to the defined functions that are distributed among the terminals in networks 204, 214 and 224, the host processing computer 202, 212, and 222 of the respective terminal network, and the other organisation's host computer (e.g., ANSI X9.2).



Inter-organisation host computer messages can also be defined for provision of services to other organisations (e.g., SWIFT for electronic fund transfer). An example of such a system is shown in Fig. 3 where host processing computers 302, 312 and 322 are configured for communication with one another. Such inter-organisational messages are not device related, but are specific for the purpose of inter-organisation data (e.g., fund) transfer. For example, a number of major manufacturers and retailers use Electronic Data Interchange (EDI), which is a structured data format used to exchange data.

As terminals become more powerful and have greater processing power, they also offer more functionality packaged as a server, thereby providing services to clients that may reside on a traditional mainframe computer or another computer. The evolution of client server technology is such that server functions need not be at the level of direct device operation, but may be at a higher level involving multiple devices or multiple operations per device. This is obvious in client and server computers that are host computers with significant processing power.

This situation is one where operations of the service requester (i.e., the "client") and the service server are again tightly coupled via very specific message formats and flows. The message formats and flows are specific to the service intended to be provided by the server. For example, the service requester or client may be a bank mainframe computer application 140 of Fig. 1A and the service server may be the built-in software of an ATM 142, where the bank mainframe application requests the service of the ATM's cash dispenser.

These message flows and formats between the service requester and the service server are defined separately from the communication layer. Further, the messages may also flow between a client and a server running in the same computer. For example, the messages may be via an application program interface (API) calls or local socket messages. Quite often, as the services offered change, software at both the client and server has to be upgraded concurrently to stay in synchronisation.

The client may be provided access to sets of services provided by the server only for as long as the client is authenticated to use those sets of services. The sequence of service calls may also be tightly coupled with the nature of the services and how the services are to be provided. For example, a transaction status message may be sent from a terminal to a host computer only after an earlier transaction authorisation has been received from the host computer to execute the transaction at the terminal.

Corporate interest in simple administration workstations has also lead to network computers (or netPCs/managed PCs) which load applications (e.g., Web

pages, Java applets) from an application server and execute the applications on the server. Such a network computer provides a remote user interface for Windows applications executed in the server. A well-defined PC configuration can run off an application server. However, this approach does not cater for non-standard devices that may be attached to a PC.

Summary of the Invention

In accordance with a first aspect of the invention, there is provided a method of communicating between a service server and at least one service client in a network, the method comprising the steps of:

publishing a service specification of the service server which uses a mix of none, or 1 or more standard or non-standard device(s), the service specification being defined in accordance with a service specification model for a service server in the network;

submitting a service request by said service client, said service request defined in accordance with said service specification model; and

generating a service server event by said service server, said service server event defined in accordance with the said service specification model.

Preferably, the service specification model defines a framework of semantics for modelling message formats and flows between the service server and the service client, both of which are independently developed with respect to each other.

Preferably, the method also comprises the step of incorporating a client context token in the service request. It may also comprise the step of either generating a solicited service server event, by the service server, in response to the service request, or generating an unsolicited service server event by said service server. Still further, the method may comprise the step of playing back the client context token in the solicited service server event.

Preferably, the method comprises the step of incorporating a redemption coupon in the solicited service server event or the unsolicited service server event. The redemption coupon contains information from the solicited service server event relating the service request to another service request. The redemption coupon may contain information from the service server to enable the service server to later determine if one or more prerequisite conditions for accepting another service request have been satisfied. Optionally, the one or more prerequisite conditions comprises at least one of the group consisting of: a set of service server events, where acceptance of the service request by the service server occurs with prior occurrence of one or more events of the set for the service client; and accumulated mileage points from

one or more prior service server events, where the accumulated points are reset at the discretion of the service server.

Preferably, the method further comprises the step of submitting another service request by the service client in response to the server event, the another service request including the redemption coupon. The service server may determine a client context for processing the another service request by using the redemption coupon in another service request by using the redemptions coupon by another service request.

Optionally, the method may further comprise at least one of the group of steps by the service server consisting of: validating the service request for a supported service and matching service specification of the service server; validating and/or authenticating the service client; and validating and/or authenticating the service server request message including its redemption coupon. Also, the method execution by the service client may optionally comprise at least one of the group of steps consisting of: validating the service server event by the service client; validating and/or authenticating the service server; and validating and/or authenticating a client context token.

Optionally, the service client is implemented in a computer system implementing the service server; this may be done using application software downloaded to and executed by the computer system.

In accordance with a second aspect of the invention, there is provided a system for communicating between a service server and at least one service client in a network, the system comprising:

means for publishing a service specification of the service server which uses a mix of none, or 1 or more standard and non-standard devices, the service specification being defined in accordance with a service specification model for a service server in the network;

means for submitting a service request by the service client, the service request defined in accordance with the service specification model;

means for generation of service server event by service server, the solicited service server event defined in accordance with the service specification model.

In accordance with a third aspect of the invention, there is provided a computer program product comprising a computer readable medium having a computer program recorded thereon for communicating between a service server and at least one service client in a network, the computer program product comprising:

means for publishing a service specification of the service server which uses a mix of none, or 1 or more of standard and non-standard devices, the service

specification being defined in accordance with a service specification model for a service server in the network;

means for submitting a service request by the service client, the service request defined in accordance with the service specification model; and

means for generating a service server event by said service server, said service server event defined in accordance with said specification model.

Brief Description of the Drawings

Embodiments of the invention are described hereinafter with reference to the drawings, in which:

Figs. 1A and 1B are block diagrams illustrating several conventional host computer and terminal configurations;

Fig. 2 is a block diagram illustrating communications between a number of host computers, each having an associated terminal network;

Fig. 3 is a block diagram illustrating inter-organisation communications between several host computers;

Fig. 4 is a block diagram illustrating a public network permitting communications between service servers and service clients in accordance with a preferred embodiment of the invention;

Fig. 5 is a block diagram illustrating a network service server and a remote client in accordance with the preferred embodiment;

Fig. 6A is a flow diagram illustrating the transmission of messages between the service server and service client of Fig. 5 in a first scenario;

Figs. 6B and 6C are flow diagrams illustrating message transmissions in second and third scenarios;

Figs. 7A and 7B are a flow diagram illustrating the processing of a service request in the service server of Fig. 5; and

Fig. 8 is a flow diagram illustrating the processing of a service server event in the service client of Fig. 5.

Detailed Description

The present invention relates to a method, apparatus and system for defining and publishing services that a service server in a networked environment offers, and for requesting and getting the results of a service request and for unsolicited asynchronous notification of service server events. The preferred embodiment of the invention provide a universal framework of simple specifications semantics upon

which message formats and flows between independently developed service clients and service servers in the networked environment can be modelled efficiently and comprehensively, with minimal coordination for synchronisation between developers of service client and service server software. This allows new services to be introduced without disrupting existing services of a service server, and may be used to form the basis for an open and flexible configuration of network computers where individual services can be either downloaded client software (e.g. Java applets, ActiveX objects etc.) or applications executing in remote computers and where the configuration of the network computers is not limited to standard devices like a keyboard, display, mouse and smart card reader. The approach developed in the preferred embodiment is based upon each service server independently defining and publishing its service specifications. These specifications may form the basis of data used to configure the operations of zero administration managed computers. For example, the specifications may be loaded from a local smart card chip or a remote computer. The system of the preferred embodiment does not define the message content formats and flows. Actual definition of message content formats and flows may either follow closely a service specification and its sequences of events, or map to them.

Overview of Networked Environment

Fig. 4 represents an environment where any network-attached computer can act as a service server in relation to any other computer (ie, a service client) in a public network in accordance with the preferred embodiment of the invention. In this exemplary networked environment, an ATM 420 and kiosks 422 are installed and operated by a service bureau 418 via the public network 400, which is preferably the Internet or an Intranet. The kiosk service bureau 418 offers services in its kiosks to consumers of a number of different subscriber organisations. The subscriber organisations independently develop and maintain their own independently created application software for use with the kiosk. The application software may be WWW applications that are provided via a subscriber's web site. The service bureau 418 in this example is responsible for topping up cash in the ATMs and repairing the ATMs whenever necessary. Customers of a subscriber organisation such as a bank 428 can obtain cash from the ATM 420. Further, customers of another subscriber organisation such as the travel agency 432 can also use the kiosk 422 to buy travel tickets.

The kiosk service bureau 418 itself may have call centre agents, who in the process of assisting a consumer to operate the kiosk submit service requests to

operate kiosk devices on behalf of the customer. In this scenario, a kiosk network manager implemented by a computer operated by the terminal service bureau 418 is a service requester (client) to the ATMs 420 and kiosks 422. The kiosk network manager sends commands to these terminals 420, 422 and obtains common processing status responses (e.g., key synchronisation commands). The network manager also gets unsolicited terminal status change notifications from the terminals.

A bank 410 owns the branch local area network (LAN) 424, its teller PCs, and attached teller equipment (e.g., passbook printers, etc.). However, the branch LAN 424, teller PCs and attached teller equipment offer services to customers of both bank 410 and bank 428, which may for example have just been acquired by bank 410.

The statutory board 426 may act as a service server also providing a customs processing service to its service client, the import/export company 412, via the public network 400.

The home PC 430 can be a service client obtaining a purchase coupon based on the PC operator's account balance from bank 410, which is a service server to the home PC 430. The purchase coupon is then passed to a travel agency 432, acting as a service server, via the public network 400 to purchase a holiday package. The coupon is subsequently submitted by the travel agency 432, now acting as a service client, to the bank 410 to extract the appropriate funds.

The kiosk 422 may also retrieve a web page from the web site of the bank 428. The web page may have a Java applet that can be executed locally on the kiosk 422 on behalf of the bank 428, as part of servicing a customer of the bank 428.

A network computer 416 can load and execute application software (e.g., WWW pages and Java applets) from an application server 414 and provide a remote user interface for Windows applications executed by the server 414. In addition to standard PC devices like a keyboard, a mouse, a display monitor, and possibly a smart card reader, for example, other non standard devices including passbook printers can also be attached to the network computer 416.

As will be understood from the foregoing exemplary networked environment large numbers of various subscriber organisations, computer systems, and non-standard devices may be available via a public network. The components connected to the network may each be able to function both as a service server and a service client to other components and to themselves. Because each of the components may have its own message content formats and flows, a significant challenge is posed to the developers of client and server software for using such components, in view of the myriad of possible combinations.

Thus, the public network environment presents numerous difficulties, which the Applicant has identified, for independent developers of client software developed by independent organisations attempting to utilise service servers.

One difficulty posed by the networked environment described above is the issue of how independently developed software for service clients is to request services from any particular service server. A clear, simple and unambiguous service specification model is required for independent developers of client software to develop and test against. The definition of the service must also be independent of the network transport and network protocols to work across heterogeneous networks. The usual approach to address this challenge is to publish the services, and the specific message formats and sequences required to use each service. For example, suppliers of ATMs may provide the specific message formats and flows of their devices. However, this requires that the client software be catered for the various, specific message formats and flows of each device. The service server software could also participate in service capabilities discovery by responding to service query messages and could also generate service capabilities advertisement messages, in accordance to agreed standards.

Another difficulty posed by the networked environment is that a service server must be able to verify that independently developed software for service clients conforms to specifications used by the service server.

Still another difficulty is how to prevent client software from "illegal" use of services (e.g., solicit PIN entry before a consumer inserts a card or request for illegal update to the database in the service server). While a service may be defined to be as legal as possible (e.g, not to offer service to operations that cannot be executed by a remote application), certain "sensitive" services must still be offered, but only to authenticated service clients. A service request may also be accepted by the service server if it just follows a set of service server events generated previously by the service server, that has not yet "expired". Examples of this include the following:

- (a) a cash dispense service request is only to be allowed from a service client that is not only authenticated and that previously, successfully requested the encrypted PIN entry service, but also requires that the kiosk has not timed out and cancelled the transaction before the cash dispense service request comes in; and
- (b) a service request to solicit PIN entry can be accepted from a host computer acting as a service client only if a card was

inserted earlier in a kiosk and a service request was sent to the host computer to validate the card.

Still further, there is a need for the ability to ensure that a service server operates in an approved and consistent manner. For example, as part of returning a consumer's card at a kiosk, the same prompt is displayed, and upon the same timeout, the consumer is prompted a second time before the card is retained, using the same consumer interaction regardless of which service client is being serviced. It is also necessary to ensure that a client cannot eject a card (e.g., to cancel a transaction) and still have money dispensed afterwards from the kiosk.

Yet another difficulty is that each client may have multiple threads of service requests concurrently being serviced by a service server. Service requests in each thread are related to each other. Thus, in accordance with the preferred embodiment, the service server must be able to associate a new service request with prior service requests as being for the same context. A new context begins with a service server event. Service requests that are executed in response to the service server event are part of the context that an event belongs to. Subsequent service server events may also be part of the same context.

A further difficulty is that a service client must be able to associate a solicited service server event from a service server as being generated in response to a service request that was sent out earlier to the service server.

Another difficulty is that a new service request may be accepted from the same service client before completion of an earlier service request for a given context. For example, this may include a service request from a service client to cancel processing of an earlier service request from the same client.

Still further, in an insecure network, there is a need for the service server to be able to authenticate the service client identity, for the service client to be able to authenticate the service server identity, for the service server to be able to authenticate the service request message including the redemption coupon that is part of a service request message, for the service client to be able to authenticate the service server event message including the client context token that is returned as part of a service server event message and for the service client and the service server to be able to encrypt selected portions of data transferred between the service client and the service server. The owner of the service server and the owner of the service client may decide that all or some of the above security mechanisms are not required, depending on the application security requirements and depending on whether the required security protections are already available via other means (e.g., middleware, intranet firewalls, etc.).

Finally, there is a need for a service server to validate that assumptions by a service client of how the service server will service a service request must match what is actually supported by the service server.

The preferred embodiment of the invention provides a specification for service that is able to overcome or at least ameliorate these difficulties.

Service Server and Service Client of Preferred Embodiment

In accordance with the preferred embodiment, a service server may offer services that involve any of a number of device operations (e.g., cash dispenser, card reader, keyboard, etc). Also, the service server may optionally require human interaction and involve the use of guidance messages while providing its services. For example, an application running on a mainframe computer can be one form of a service server that does not require human interaction while in the process of providing the service. In contrast, a kiosk (e.g., 420 of Fig. 4) acting as a service server may offer services requested by a backend mainframe application acting as a service client that requires interaction with a consumer (e.g., asking a consumer to remove cash from a cash dispenser of the kiosk). In this circumstance where the service provided by the kiosk involves human interaction at the service server, the service operations involving kiosk devices must work in tandem with guidance screens or messages.

Such a service server in accordance with the preferred embodiment responds to a service request from a service client with one or more service server events. This caters for the asynchronous operation of devices and the asynchronous nature of response messages provided over a network. The execution or performance of a service may generate more than one event. Thus, events may be single, multiple or recurring events. For example, the processing of a service request may involve more than one step, each of which the client needs to be notified of by an event. The execution of the service may also generate recurring events. For example, a service request enabling a kiosk to accept bank cards, credit cards or the like for a service client generates one event for each new card insert by a customer of the client.

Still further, a service client that has been pre-registered (i.e., offline) with a service server may also receive unsolicited service server events from a service server. For example, a browser PC acting as a service server can send an unsolicited service server event (e.g., HTTP GET) causing a WWW site acting as a service client to send a web page to the browser PC. The HTTP GET response represents a service request to the browser to present the web page and solicit input. The web site or server can also view itself as a service server that receives a service request (e.g.,

HTTP GET) from the browser PC acting as a server client. This is one instance of many possibilities where a pair of service servers interact and independently view the other as a service client. Client software including Java applets may be downloaded to execute in the service server computer. Messages in this case are local, and are possibly inter-process or inter-thread ones.

Delivered together with the service request and the service server event may be data elements that provide variable and context dependent information. These data elements may carry executable scripts e.g. Java applets, JavaScript, HTML that may need to be signed.

Fig. 5 is a block diagram illustrating a service server and one or more service clients in accordance with the preferred embodiment. The service server 506 may be implemented using a terminal 500 such as a computer or ATM. The terminal 500 may also include middleware 504 for network communications and optionally a local client 502. The service server 506 provides operation guidance messages 508 to devices of the terminal and controls operation of terminal device(s) 510. The service server directly manage and control the terminal devices. In this example, a remote (service) client 520 communicates for services with the service server 506 via messages transmitted over the network 530, which may be the Internet, an Intranet, a LAN, a wide area network (WAN), etc. The local (service) client 502 may be a locally installed client application or can be one (e.g., Java applets) downloaded from a remote computer (e.g., 520). Such a local client 502 communicates with the service server 506 via messages or application program interface (API) calls. The service specification standard according to the preferred embodiment is used as semantics for client-server communications and is network centric. It does not specifically define the message formats and flow between the service client and the service server. The foregoing provides a block representation of the service client, service server and networked environment to which the service specification model of the preferred embodiment is applied.

Service Specification Model

The following service specification model provides specific details of a preferred embodiment. However, the invention is not limited only to the specifics presented. Instead modifications and/or changes obvious to one skilled in the art can be made thereto without departing from the scope and spirit of the invention. The service specifications of a network service server are described hereinafter and are for each service that a service server provides.

Tables A, B, C and D provide the details of how the service specifications of a service server can be defined and published.

TABLE A: Service Server Specifications

- A) Server Server Id: Unicode name string.
- B) Server description: this enables discovery by clients and advertisement to potential clients.
- C) Algorithm types that are supported by the service server for generating digital signatures for both its service request message type and its service server event message type. This specification relates to various possible algorithms that are defined and published by the service server. These algorithms may use various combinations of random seeding using message content in conjunction with various industry standard algorithms, to sign the message. Message content used for generating the digital signature in the service request message should include the client ID and other sensitive message elements in the service request message, e.g. its redemption coupon. Message content used for generating the digital signature in the service server event message should include the server ID, the client context token that was copied from a service request message (in the case of solicited service server event) and other sensitive message elements in the service server event message. Examples of industry standard algorithms are the data encryption standard (DES) using common secret symmetric keys, and the Rivest, Shamar and Adleman (RSA) public key algorithm using digital certificates.
- D) Algorithm types that are supported by the service server for encryption of selected data elements in both its service request message type and its service server event message type. This specification relates to various possible algorithms that are defined and published by the service server. These algorithms may use random data that is transmitted as part of the service request message to seed the key that is used to encrypt selected data elements in the service request message and use random data that is transmitted as part of the service server event message to encrypt selected data elements in the service server event message. Examples of industry standard algorithms are the date encryption standard (DES) using common secret symmetric keys, and the Rivest, Shamar and Adleman (RSA) public key algorithm using digital certificates.

TABLE B: Service Specifications For Each Supported Service Request

- A) Service request identity specification: a unicode service name string that is unique within the network service server.
- B) Service request category identity specification: a unicode service name string that is unique within the network service server. This enables classification of service requests.
- C) Prerequisite service request(s) acceptance mode and conditions, including either:
 - i) Prerequisite set(s) that need to be satisfied: this enables multiple paths to the service request. Within a set, a prerequisite server event may be required multiple times. An example of a prerequisite set(s) is "(A, B), (A2, D, E), (C3)", where A, B, C, D and E are server event ID/Category combinations. If second set is used, the server event A is required twice. If the third set is used, three occurrences of the server event C is required. A redemption coupon is only returned with the last C event.
 - ii) Minimum accumulated event points: these are required for the same context for the same client from prior server events. A redemption coupon is returned only with the last event which just accumulated sufficient or more points.
- D) Boolean as to whether service request can be submitted to the service server independently of a service server event from the service server. Service requests that are not part of a context (i.e., not in response to a service server event) are specified as true here.
If this specification is true, the following list applies.
 - i) List of concurrent service request ID/Category combinations: this list is provided for identifying service requests that can be accepted from the same client while the service server is still in midst of processing this service request. For example, the list may include the ID/Category of a service request that can cancel this service request while in progress.
- E) Boolean value indicating whether or not a redemption coupon is required.
- F) List of data arguments: each argument provides specific information required by the service and is identified by "data ID = value". The specification also indicates default values for any arguments that are not

provided. These are free format specifications that are not tied to the client or server configuration. For each data element, a Boolean value is provided as to whether or not a data element is encrypted. For each data element, a text description of each data element if provided, is good for discovery.

- G) Solicited server event information: contains a list of events that may be generated as a result of processing of this service request. Even if the service is synchronous, its completion is indicated by at least one event. The events listed below may also be specified as events that may be generated as a result of other service request.
- i) Each event is uniquely identified. It is recommended that the service request ID/Category string combination be part of the event ID/Category string to enable easy association of an event with the service request that results in that event. This is relevant if that event results only from one specific service request.
 - a) Server event ID unicode string: unique within each server event category
 - b) Server event category ID unicode string: unique within server for both solicited and unsolicited server event categories
 - ii) Detailed information about each event:
 - a) Descriptive text: useful for discovery
 - b) Boolean value indicating whether or not a redemption coupon is available with this event message
 - c) Mileage points earned, if any, with this event
 - d) Descriptive text description: useful for server discovery.
 - e) A list of service server event data items. Each data item provides specific information about the event and may be identified by 'data ID = value'. The specification does not specify the format of the data item. For each data item, a Boolean value indicates whether or not the data item is to be encrypted. For each data element, a text description of each data element if provided, is good for discovery.
 - f) Flag from one of the group that includes the following four values:
 - start new context,
 - middle of context,
 - end of context, and

- independent one off event, not part of a context.
- g) List of set(s) of service requests that may be issued by service client on receipt of this event. ((A, B), (C, D, E), (F, F)) means that one of the following service request A, C or F can be accepted from the service client that receives this event. While service request A is being processed, service request B can be issued to the service server, while service request C is in progress, either service request D or E can be issued to the service server, while service request F is in progress, another service request F can be issued to the service server.
- iii) Standard events having ID strings and meanings that are predefined
 - a) Service not found
 - b) Service specifications mismatch
 - c) Context cancelled by service server (e.g., consumer press CANCEL key)
 - d) Context cancelled by accepted service request from same client
 - e) Required resources (e.g. devices) not available
 - f) Redemption coupon already redeemed
 - g) Redemption coupon invalid
 - h) Redemption coupon expired
 - i) Client authentication failure
 - j) Redemption coupon authentication failure
- iv) Information about other non standard event types including rejection for undefined reasons
 - a) Boolean as to whether it is recurring
 - b) Boolean as to whether the event indicates the end of processing of a service request.

**TABLE C: Service Specifications For Each Supported Unsolicited Service
Server Event**

-
- i) Server event ID unique within each server event category
 - ii) Server event category identity specification unique within the server for both solicited and unsolicited service server event categories
 - iii) Descriptive text: useful for discovery
 - iv) Detailed information about the event:
 - * Boolean value indicating whether or not a redemption coupon is available with this event message.
 - * Mileage points earned, if any, with this event
 - * A list of service server event data items. Each data item provides specific information about the event and may be identified by 'data ID = value'. The specification does not specify the format of the data item. For each data item, a Boolean value indicates whether or not the data item is to be encrypted.
 - * Flag from one of the group that includes the following four values : start new context, middle of context, end of context and independent one off event - not part of a context.
 - * List of set(s) of service requests that may be issued by service client on receipt of this event. ((A,B), (C,D,E), (F,F)) means that one of the following service request A, C or F can be accepted from the service client that receives this event. While service request A is being processed, service request B can be issued to the service server, while service request C is in progress, either service request D or E can be issued to the service server, while service request F is in progress, another service request F can be issued.

TABLE D: Service Specifications For Message types and Elements

-
- i) Service request message specification
 - a) Service request identity
 - b) Service request category identity

- c) List of data arguments: each of which provides a specific information required by the service. Each argument is identified by "data ID = value".
- d) Service specifications digest: this allows the service server software to validate that the service specification used by the client is same as what the service server supports.
- e) Client ID
- f) Client context token (to be played back by server in generated server event(s))
- g) Redemption coupon to be used by this service request for the client. The redemption coupon may be used by the service server to validate that the current service request is processed after the service server has reached a desired state (e.g. generation of one or more earlier service server events in response to earlier service requests). For example, successful completion of an earlier key synchronisation request, successful acceptance of consumer card and solicitation of PIN entry. The service server also validates that the redemption coupon is still valid. For example, the service may have timeout and cancel the last transaction, or the service server may have rebooted. The redemption coupon may be from one of two sources:
 - * Service server event(s) generated in processing earlier service requests from the service client. The redemption coupon is returned with a server event that was generated in response to an earlier service request from the client. This redemption coupon may be null either because the service request is issued independently of a service server event and hence no redemption coupon is available, or because the service server event that leads to the service request does not provide a redemption coupon.
 - * An earlier unsolicited service server event to client. For example, a card insert occurrence at a kiosk results in a service server event being sent to the host to validate the card. This service server event may be delivered with a redemption coupon that the host can use in a service request to the kiosk to solicit entry of PIN.

h) Algorithm type used in generating secure digital signature to enable the service server to authenticate a service request message (i.e. above items). The service client indicates here the authentication algorithm used. The service server may be configured as to whether use of this field is mandatory, possibly for each service client. Each service client may or may not be part of an insecure network.

i) Algorithm type used in encrypting selected data elements.

j) Secure digital signature that authenticates the service request message itself. This enables the service server to:

- * Authenticate the source (i.e. client ID)
- * Detect tampering with the service request (e.g. changing data elements)
- * Detect illegal recording/playback of redemption coupon by unauthorised clients

The specifics of the digital signature are not defined and is left to the service server owner to specify. The signature is optional. Well controlled Intranets may not require this. The required authentication may also be provided by the middleware 504 that connects the service server 506 and the service client 502, 520. The service server may be configured as whether use of this field is mandatory, possibly for each service client. Each service client may or may not be part of an insecure network.

k) Random seed for encryption key used for encryption. This may be a random key encrypted under a fixed secret key that is known only to both the service client and service server.

ii) Service server event message specification

- a) Boolean value indicating whether or not the event message is solicited
- b) Identity of service request that activates the event (ignore if unsolicited)
- c) Identity of service request category that activates the event
- d) Server ID
- e) Service server event ID
- f) Service server event category
- g) Copy of client context coupon (ignore if unsolicited)

- h) A list of service server event data items: each of which provides a specific information provided by the service. Each argument is identified by "data ID = value".
- i) Algorithm type used by server to generate digital signature. Server needs to indicate the authentication algorithm used.
- j) Secure digital signature that authenticates the service server event message content listed(s) just above. This is optional. Well controlled Intranets may not need require this. The required authentication may also be provided by the middleware that connects the service and the client. This is an option that is configured as part of business arrangement contract between a service client and service server.
- k) Algorithm type used for encryption of selected data elements.
 - * If solicited, specified by client in service request message.
 - * If unsolicited, configured in service server.
- l) Service server event redemption coupon. The specification does not require publication of the format of the coupon. The specifics listed here only serve as an example. It is not part of the service specification and need not be published. All the client needs to know is that it needs to include this coupon with the next service request. Suggested content at discretion of server:
 - * Context ID for reference to information like:
 - ID of the last service request
 - ID of service client for which the coupon is intended for
 - prerequisite service request processing information for the same context for the same client. This can be, for example, either mileage points accumulated so far by the server for the client for the current context, or a list of server events that have occurred to earn mileage for the client for the current context.
 - * Numbers randomly generated by the service server together with date/time stamps, if necessary to prevent reuse of redemption coupons generated earlier. The randomness can

be used by the service server as random changing password to be used in subsequent service request.

- * May be partially or fully encrypted at discretion of service server.

- m) Random seed for encryption key use for encryption of selected data elements. This may be a random key encrypted under a fixed secret key that is known only to both the service client and service server.

In accordance with the preferred embodiment, a service provides all operation guidance and interaction that the relevant service server 506, shown in Fig. 5, requires during the course of processing a service request and at the end of such processing. While requesting a service, service client software 502, 520 does not concurrently provide operation guidance and interaction 508. This enables the service server 506 to ensure consistent operation guidance and interaction 508 for human operators. The following service specification specifies that all operation guidance and interaction 508 required during the course of providing a service are to be provided within the service. The operation guidance and interaction 508 used in the course of and at the end of providing service are themselves not part of the service specification. The operation guidance and interaction 508 may be in terms of LED lights, and need not be in terms of display monitor presentations only. This is optional and can be left up to the service server 506. Thus, the service server owner can change operation guidance and interaction 508 without impacting on service client software 502, 520. This helps to ensure that the service server 506 operates and provides operation guidance and interaction 508 in a consistent and approved manner. The service server 508 may offer services that include display guidance and interaction objectives. Actual presentation and interaction is at the discretion of the service server.

Further, the service server 506 must be capable of taking over human guidance, whenever necessary, while the service is being processed. For example, a service server must be able to take over a display screen from a client's downloaded Java applet or web page, while dispensing cash and prompting the consumer to remove the issued cash. This may be achieved by the service server minimising or hiding a browser window.

In the service specification, a client identifier or ID is also needed by the service server 506 to identify the service client 502, 520. Likewise, the client 502,

520 requires a server ID to identify the service server 506. The client ID and server ID may be determined and assigned by the service server 506 providing the relevant service. The server ID is unique within some subnet that the client 502, 520 can identify (e.g., a unique Internet host IP address). Alternatively, unique client IDs and server IDs may be assigned by a network authority. All clients 502, 520 that need to get services from a service server have to get a respective client ID, the service server ID and keys required for authenticating and/or encrypting, as part of a business arrangement contract. The contract may be obtained online in any of a number of ways. Likewise, the client/server IDs may be mapped to networking transport address in any of a number of ways. Both of these aspects are the domain of the networking middleware 504 of Fig. 5.

Also in the service specification, a client context token may be submitted by the service client 502, 520 as part of a service request (e.g., see service request 604 of Fig. 6 described hereinafter) to the service server 506. The service server 506 is expected to play back the client context token in service server event(s) generated by processing the service request. This is to enable the service client 502, 520 to associate service events to service requests that have been submitted earlier. The service specification does not define the specific format of the token. The token format is for proprietary definition by the client and need not be published by the client. All that the service server 506 needs to do is to pass it back in server event(s) generated as a result of processing the client's service request. Authentication of the client context token is purely at the discretion of the service client 502, 520 that issued it. For example, the service client 502, 520 may verify the integrity of its client context coupon, described below, using cryptography technology (e.g., encrypted by private key known only to the client XOR with dynamic token content data). Not defining the format of the client context token enables flexibility in how a service client may validate the correctness and integrity of a context token. The token may be expired by the issuer (e.g. client gives up waiting for the response service server event).

A redemption coupon serves two specific purposes. The first is to permit the service server 506 to place, directly or indirectly via reference, data in the coupon that enables the service server 506 to relate a service request to other related service requests (e.g., same transaction, same client). The second is so that the service server 506 can place, directly or indirectly via reference, data that enables the service server 506 to determine if prerequisite conditions for accepting a service request are in place (e.g., recent successful completion of earlier service requests). The redemption coupon is to be passed to the server client 502, 520 in a service server

event from the service server 506. The redemption coupon is then submitted as part of the next related service request to the service server 506. The context to be used for processing a request is identified by the service server 506 by examining the redemption coupon.

The specification does not define the format of the redemption coupon. It is for proprietary definition by the service server 506 and need not be published by the service server 506. All that the service client 502, 520 needs to do is to include it in its next service request to the service server 506. Authentication of the redemption coupon is purely at the discretion of the service server 506 that issued it. For example, the service server may verify the integrity of its coupon token using cryptography technology (e.g., encrypted by a private key known only to the service server XOR with dynamic content data). Not defining the format of the redemption coupon enables flexibility in how a service server 506 validates the correctness and integrity of the redemption coupon. The redemption coupon may be time expired by the issuer (e.g., service server gives up waiting for the next service request from the coupon recipient and cancels the transaction).

Each service request specifies prerequisite conditions for acceptance, which when reached are reflected in the redemption coupon. A redemption coupon is therefore also cumulative, with each subsequently generated redemption coupon earning more mileage, if the previous one is not used. Service request prerequisite condition check processing can be defined and published in two modes. Firstly, the service server 506 can generate for the client, one of a set(s) of selected service server event(s). This permits restriction of acceptance of a service request only with prior occurrences of selected events for the service client 502, 520. These events may only be the result of appropriate processing of prior selected service request(s) from the client 502, 520. Secondly, the client may have accumulated sufficient points from prior events. Cumulated points are tracked by the service server 506 via the returned redemption coupons. The redemption coupon format may be published by the service server 506 to provide instructions to developers of the client application. The above "mileage earned by coupon recipient" can be represented in the redemption coupon by the service server 506. The mileage earned by the service client 502, 520 is reset at the discretion of the service server 506.

In the service specification, each service request is defined with its own set of data elements. Also, each service server event (event ID) is defined with its own set of data elements. The data elements enable data transfer, beyond what is in the basic common basic definition, to and from the service server 506 and its clients 502, 520. If the service client 502, 520 needs to get data (e.g., a purchasing coupon) from

one service server and pass it in a service request to another service server, the data elements provide the vehicle for doing so. This case however requires a special arrangement between the two service servers as to the format and content of the data elements (e.g., private security schemes between the two servers). Encryption of any data element, if required, is specified by the service server in the service specification. Unsolicited service server events may be published by the service server 506 for posting new keys (encrypted in the predefined keys) to service clients. The data elements may be used to contain executable scripts. Such data elements are likely to be included in the generation of the digital signature.

In the service specification, digital signatures are preferably used to verify authenticity of service request and server event messages. They are used to authenticate the source of a message, detect tampering of a message, and detect unauthorised/illegal recording or playback of redemption coupon by unauthorised service clients.

As mentioned hereinbefore, a service request may also result in recurring events. Therefore, recurring service server event messages are used (e.g., one per card insert in response to an open-for-card-transactions service request from a service client). As part of processing of each service request, the service specification assumed by the client is also compared with what is actually supported by the service server to ensure synchronisation of service specification assumptions.

In the service specification, a service client 502, 520 is expected to timeout while waiting for a service server event, if necessary.

Also, there is not any direct reference to any devices or their functions in the service specification. A display service indirectly uses the display device 510 in a manner allowed by the service server 506. This offers the service server 506 flexibility in choice of configuration of devices 510 to be used to perform a service request.

In the service specification, a service server event may occur only after the occurrence of a physical event (e.g., insertion or removal of a consumer card).

The service specification does not define the network message formats and flows for requesting a service and getting the results of processing a service request. Thus, it is left open to the middleware 504 to provide the communication, handshaking, guaranteed delivery, etc, between the service server 506 and the service client 502, 520. For example, a service server event may be communicated via an asynchronous message from the service server 506, using optimisation strategies selected by the middleware.

As mentioned above, the service client 502 (e.g., a downloaded Java applet) may execute locally in the same terminal or computer as the service server 506.

Thus, the service specifications cover a full spectrum of messaging requirements. However, other implementations may optionally practice only a subset of the specifications. For example, the digital signature/encryption of data elements may be skipped if the network is secured, or the client ID and the server ID may be skipped if the middleware 504 takes care of client and server routing and authentication.

Exemplary Communication Scenarios

Fig. 6A is a flow diagram illustrating one scenario of communications between a service client 502, 520 and a service server 506. Again, the service client 502 may be a downloaded applet executing on the terminal. Processing commences in step 602 by the service client 502, 520 sending a service request 604 to the service server 506. The service request 604 includes a client context token 1. The service server 506 responds to the service request 604 in step 606. In step 606, the service request 604 is validated for supported service and matching service specifications. Also, the client is validated/authenticated, the service request message including its redemption coupon is validated/authenticated. If the service request 604 is rejected, a service server reject event 608 is transmitted by the service server 506 to the service client 502, 520. The service server reject event 608 plays back the client context token 1, and ends processing of the service server 506. Otherwise, if the service request is accepted by the service server 506, processing continues at step 610. In step 610, the context is retrieved from the service request 604. In this step, operation guidance and interaction are provided, if necessary, and multiple asynchronous operations occur in relation to the terminal devices. The service server 506 then sends a service server event (solicited) 612 to the service client 502, 520. The service server event 612 includes a redemption coupon A, and plays back the client context token 1. The service server event (solicited) 612 may be a single, multiple, or recurring event. Upon receipt of the service server event 612 by the service client 502, 520, the service client 502, 520 validates/authenticates the service server event message including its server ID and its client context token, then sends a further service request in step 614 to the service server 506. This further service request 616 includes a client context coupon 2 and plays back the redemption coupon A.

The process of communications then continues in accordance with steps 606 and 610 with reference to the first service request 604.

Fig. 6B illustrates a second exemplary message scenario between the service server 506 and the service client 502, 520. Processing commences in step 652 where the service server 506 sends a service server event 654 to the service client 502, 520. The service server event (unsolicited) 654 includes a redemption coupon B. Upon receipt of the service server event 654 by the service client 502, 520, a service request is sent to the service server 506 in step 656. The service request 658 includes a client context token 3, and plays back redemption coupon B. In step 660, the service server 506 validates the service request for supported service and matching service specification, and also validates/authenticates the client and the service request message including its redemption coupon. If the service request is rejected by the service server 506, a service server reject event 662 is sent to the service client 502, 520, which plays back the client context token 3. Otherwise, the processing of the service server 506 continues at step 664. In step 664, the context is retrieved, and operation guidance and interaction, if required and asynchronous operations may occur. The service server 506 then sends a service server event (solicited) 666 to the service client 502, 520. The service server event 666 includes a redemption coupon C, and plays back the client context token 3. The service client 502, 520 validates/authenticates the service server event message including its server ID and its client context token, then sends a service request in step 668 to the service server 506. This further service request 670 includes a client context token 4, and plays back the redemption coupon C.

Fig. 6C illustrates a third exemplary message scenario between the service server 506 and the service clients 502, 520. The service server 506 sends an unsolicited service server event in step 672 to the service client. The service server event (unsolicited) 674 may be a server status message, key posting message, etc., and may have a redemption coupon.

The middleware 504 may build additional messages on top of those in the scenarios of Figs. 6A to 6C for message integrity, synchronisation, guaranteed message delivery, etc. It may also present the same information in different formats.

Processing of A Service Request in Service Server

Fig. 7 is a flow diagram illustrating an example of processing of a service request in the server 506. Processing commences in step 700. In step 702, the next service request is obtained from the client or an internal job queue. In the latter event, internally generated service requests from the internal queue generate unsolicited server events. In decision block 704, a check is made to determine if the client is valid. If decision block 704 returns false (no), a rejected event is posted at

step 706. Otherwise, processing continues at decision block 708. In decision block 708, a check is made to determine if the service request including the redemption coupon is valid and has not been tampered with (i.e., the source is valid etc). The digital signature if available would be used here. If decision block returns false (no), a rejected event is posted at step 710. Otherwise, processing continues at decision block 712. In decision block 712, a check is made to determine if the coupon for redemption is valid. This may involve matching the context ID, checking if another service request with the same context is in progress. If so, a check is made to determine if it is allowed for the service request. A check may also be made to determine if prerequisite conditions have been met. If decision block 712 returns false (no), processing continues at step 714 and a rejected event is posted. Otherwise, if decision block 712 returns true (yes), processing continues at step 716. In step 716, the rest of the service request message is disassembled. This may involve extracting service request data elements, decrypting selected ones, and saving a client context token, if any. Processing then continues at decision block 718.

In decision block 718, a check is made to determine if the service request is part of an existing context. If decision block 718 returns true (yes), processing continues at step 724. In step 724, the context control block is retrieved. Processing then continues at step 726. Otherwise, if decision block 718 returns false (no), processing continues at decision block 720. In decision block 720, a check is made to determine if required resource(s) are available. If decision block 720 return false (no), processing continues at step 722 and an event is posted. Otherwise, if decision block 720 returns true (yes), processing continues at step 726.

In step 726, processing specific to the service request is carried out with respect to the service request state. Operation guidance and interaction may be provided, where required. Processing then continues at decision block 728. In decision block 728, a check is made to determine if this is a new context. If decision block 728 returns true (yes), processing continues at step 730. In step 730, a new context ID is assigned for the coupon, and a new context control block is created. Processing then continues at decision block 732. Otherwise, if decision block 728 returns false (no), processing continues at decision block 732.

In decision block 732, a check is made to determine if there is a redemption coupon for the client. If decision block 732, returns false (no), processing continues at step 736. Otherwise, if decision block 732 returns true (yes), processing continues at step 734. In step 734, a redemption coupon is created. Processing then continues at step 736.

In step 736, an event message is assembled. In step 738, a service server event is posted. In decision block 740 a check is made to determine if processing has ended for the service request. If decision block 740 returns false (no), processing continues at step 726.

Otherwise, processing continues at decision block 742. In decision block 742, a check is made to determine if the end of the context has been reached. If decision block 742 returns true (yes), processing continues at step 744. In step 744, the context control block is deleted. Processing then continues at step 746. Otherwise, if decision block 742 returns false (no), processing continues at step 748. In step 748, the context control block is saved. Processing then continues at step 746. In step 746, processing of the service request terminates.

Processing of Service Server Event in Service Client

Fig 8 is a flow diagram illustrating an example of processing of a service server event at a service client that is network centric. Processing commences in step 800. In step 802, the next service server event is obtained from the service server. In decision block 804, a check is made to determine if the service server is valid. If decision block 804 returns false (no), error processing is carried out in step 806. Otherwise, processing continues at decision block 808. In decision block 808, a check is made to determine if the server event is valid. This may involve checking to determine if the event message including the client context token is valid and has not been tampered with, the source is authenticated, etc. The digital signature, if available, is used here. Again, if decision block 808 returns false (no), error processing is carried out in step 810. Otherwise, if decision block 808 returns true (yes), processing continues at decision block 812.

In decision block 812, a check is made to determine if the service server event is unsolicited. If decision block 812 returns true (yes), processing continues at decision block 816. Otherwise, if decision block 812 returns false (no), processing continues at step 814. In step 814, the context is retrieved using the client context token. Processing then continues at decision block 816.

In decision block 816, a check is made to determine if a redemption coupon is available. That is, a check is made to determine if a coupon is available to subsequent redemption by the service client at a service server. If decision block 816 returns true (yes), processing continues at step 818. In step 818, the redemption coupon is saved for later use. Processing then continues at step 820. Otherwise, if decision block 816 returns false (no), processing continues at step 820. In step 820, the event is processed. This may involve extracting event data elements

and decrypting selected ones of them. Processing then continues at decision block 822.

In decision block 822, a check is made to determine if the next service request is to be submitted to the service server. If decision block 822 returns false (no), processing continues at decision block 826. Otherwise, if decision block 822 returns true (yes), processing continues at step 824. In step 824, a new client context token is created and a service request is submitted to the service server. This may involve generating a client context token, assembling data elements (including encrypting selected ones), obtaining a redemption coupon that was saved earlier, and building a digital signal signature. Processing then continues at decision block 826.

In decision block 826, a check is made to determine if the service server event is an unsolicited one. If decision block 826 returns true (yes), processing continues at step 830. Otherwise, if decision block 826 returns false (no), processing continues at step 828. In step 828, the existing client context that had been associated with the processed service server event is deleted, and processing then continues at step 830. In step 830, processing of the event terminates.

Only one embodiment of the invention has been described. However, it will be apparent to those skilled in the art that modifications and/or changes can be made thereto without departing from the scope and spirit of the invention.



CLAIMS:

1. A method of communicating between a service server and at least one service client in a network, said method comprising the steps of:

publishing a service specification of said service server which uses a mix of none, or 1 or more of standard or non-standard devices, said service specification being defined in accordance with a service specification model for a service server in said network;

submitting a service request by said service client, said service request defined in accordance with said service specification model; and

generating a service server event by said service server, said service server event defined in accordance with the said service specification model.

2. The method according to claim 1, wherein said service specification model defines a framework of semantics for modelling message formats and flows between said service server and said service client, both of which are independently developed with respect to each other.

3. The method according to claim 1, further comprising the step of incorporating a client context token in said service request.

4. The method according to claim 1, further comprising the step of either generating a solicited service server event, by said service server, in response to said service request, or generating an unsolicited service server event by said service server.

5. The method according to claim 4, further comprising the step of playing back said client context token in said solicited service server event.

6. The method according to claim 4, further comprising the step of incorporating a redemption coupon in said solicited service server event or said unsolicited service server event.

7. The method according to claim 6, wherein said redemption coupon contains information from said solicited service server event relating said service request to another service request.

8. The method according to claim 6, wherein said redemption coupon contains information from said service server to enable said service server to later determine if one or more prerequisite conditions for accepting a service request have been satisfied.

9. The method according to claim 8, wherein said one or more prerequisite conditions comprises at least one of the group consisting of:



a set of service server events, where acceptance of said service request by said service server occurs with prior occurrence of one or more events of said set for said service client; and

accumulated mileage points from one or more prior service server events, where said accumulated points are reset when said redemption coupon is used.

10. The method according to claim 8, further comprising the step of submitting another service request by said service client in response to said server event, said another service request including said redemption coupon.

11. The method according to claim 10, further comprising the step of determining, by said service server, a client context for processing said another service request by using said redemption coupon of said another service request.

12. The method according to claim 1, further comprising at least one of the group of steps by the service server consisting of:

validating said service request for a supported service and matching service specification of said service server;

validating and/or authenticating said service client; and

validating and/or authenticating said service request message including its redemption coupon.

13. The method according to claim 1, further comprising at least one of the group of steps by the service client consisting of:

validating and/or authenticating said service server event message by said service client;

validating and/or authenticating said service server.

14. The method according to claim 1, wherein said service client implemented in a computer system implementing said service server.

15. The method according to claim 14, wherein said service client is implemented using application software downloaded to and executed by said computer system.

16. A system for communicating between a service server and at least one service client in a network, said system comprising:

means for publishing a service specification of said service server which uses a mix of none, or 1 or more of standard and non-standard devices, said service specification being defined in accordance with a service specification model for a service server in said network;

means for submitting a service request by said service client, said service request defined in accordance with said service specification model; and



means for generating a service server event by said service server, said service server event defined in accordance with said specification model.

17. The system according to claim 16, wherein said service specification model defines a framework of semantics for modelling message formats and flows between said service server and said service client, both of which are independently developed with respect to each other.

18. The system according to claim 16, further comprising means for incorporating a client context token in said service request.

19. The system according to claim 16, further comprising means for either generating a solicited service server event in response to said server request or an unsolicited service server event by said service server.

20. The system according to claim 19, further comprising means for playing back said client context token in said solicited service server event.

21. The system according to claim 19, further comprising means for incorporating a redemption coupon in said solicited service server event or said unsolicited service server event.

22. The system according to claim 21, wherein said redemption coupon contains information from said solicited service server event relating said service request to another service request and also contains information from said service server to enable said service server to later determine if one or more prerequisite conditions for accepting a service request have been satisfied.

23. The system according to claim 22, further comprising means for submitting another service request by said service client in response to said server event, said another service request including said redemption coupon.

24. The system according to claim 23, further comprising means for determining a client context for processing said another service request by using said redemption coupon of said another service request.

25. A computer program product comprising a computer readable medium having a computer program recorded thereon for communicating between a service server and at least one service client in a network, said computer program product comprising:

means for publishing a service specification of said service server which uses a mix of none, or 1 or more of standard and non-standard device(s), said service specification being defined in accordance with a service specification model for a service server in said network;

means for submitting a service request by said service client, said service request defined in accordance with said service specification model; and



means for generating a service server event by said service server, said service server event defined in accordance with said specification model.

26. The computer program product according to claim 25, wherein said service specification model defines a framework of semantics for modelling message formats and flows between said service server and said service client, both of which are independently developed with respect to each other.

27. The computer program product according to claim 25, further comprising means for incorporating a client context token in said service request.

28. The computer program product according to claim 25, further comprising means for generating either a solicited service server event in response to said service request or an unsolicited service server event independently of service request processing.

29. The computer program product according to claim 27, further comprising means for playing back said client context token in said server event.

30. The computer program product according to claim 28, further comprising means for incorporating a redemption coupon in said solicited service server event or said unsolicited service server event.

31. The computer program product according to claim 30, wherein said redemption coupon contains information from said solicited service server event relating said service request to another service request and also contains information from said service server to enable said service server to later determine if one or more prerequisite conditions for accepting a service request have been satisfied.

32. The computer program product according to claim 31, further comprising means for submitting another service request by said service client in response to said server event, said another service request including said redemption coupon.

33. The computer program product according to claim 32, further comprising means for determining a client context for processing said another service request by using said redemption coupon of said another service request.