(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 January 2013 (24.01.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/012399 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/TR2012/000112

(22) International Filing Date:
11 July 2012 (11.07.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2011/07021 15 July 2011 (15.07.2011) TR

(71) Applicant *(for all designated States except US)*: **NETAS TELEKOMUNIKASYON ANONIM SIRKETI** [TR/TR]; Alemdag Cad. No:171 umraniye, 34768 Istanbul (TR).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **KARADOĞAN, İbrahim Cuma** [TR/TR]; Alemdag Cad. No:171 Umraniye, 34768 Istanbul (TR).

(74) Agent: **DESTEK PATENT,INC.**; Konak Mah. Lefkose Cad. NM Ofis Park B, Blok No: 36 / 5 BEŞEVLER NİLÜFER, 16110 Bursa (TR).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) **Title**: SERVICE MONITORING SYSTEM HAVING AN EXTENSIBLE SERVICE DEFINITION LANGUAGE

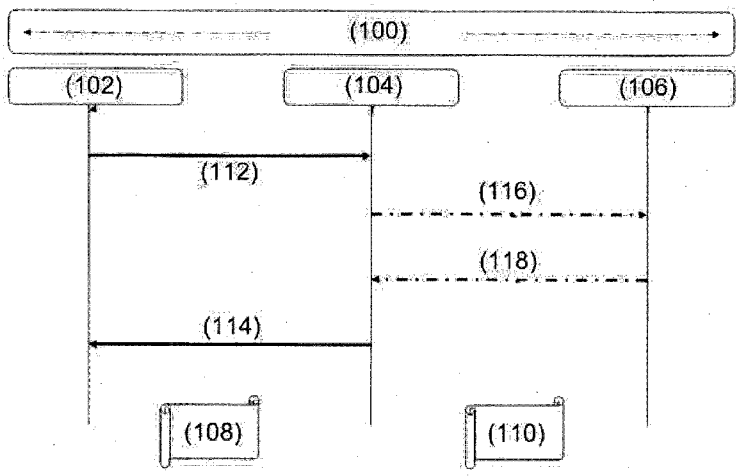

Figure 1

(57) **Abstract**: Service monitoring is a working area which intercessantly examines a system using transaction in order to carry out required operations of a service. A transaction is spread to one or multiple underlying communication protocol messages getting assembled in order to realise a desired service. Present invention relates to monitoring transaction-related service systems and providing required reports depending on the data being monitored. In order to monitor said services, present system taps the physical communication lines via TAP devices or monitored key connection ports and parses protocol messages. It parses protocol messages and identifies transaction state by using ESDL definitions of services monitored. By associating parsed messages with transaction and service sessions it stores them in system data base.

WO 2013/012399 A2

DESCRIPTION

SERVICE MONITORING SYSTEM HAVING AN EXTENSIBLE SERVICE DEFINITION LANGUAGE

The Related Art

The invention relates to monitoring operational service systems and providing required reports depending on the data being monitored.

The invention relates in particular to a service monitoring system having Extensible Service Definition Language (ESDL) focusing on monitoring service data carried over relevant protocol messages as the data carried.

The Prior Art

Service monitoring is a working area that intercessantly studies a system using transaction in order to carried out required operations of a service. A transaction covers communication protocol messages coming together to realise an underlying service. A service can additionally comprise multiple protocols in order to complete its operations.

Today, monitoring systems cover only protocol monitoring. This state means that said monitoring systems monitor only parsed protocol messaging via examining request/response peer of the protocols being monitored. That is, they monitor whether request successfully has respectively arrived destination and the response has been received by the source or not.

Monitoring only protocol message excange between the source and destination can not satisfactorily meet the requirements of today's Value Added Service-VAS) monitoring. VAS is the essential system behind the standard voiced calls, data communication and facsimile transmission services.

In case a service transaction expands in a way covering multiple prioritized message Exchange, monitoring of separate messages as separate entities will not ensure full monitoring service state. For example, in case transaction needs 5 protocol request

messages in order to fulfil a service and in case four of these request messages are successful yet the last one is not successful, all of the service transaction is unsuccessful. However, since monitoring of entire service state with the presently used monitoring systems is not possible, each one of these five messages is monitored a separate entry.

Another disadvantages in the prior art, on the other hand, is from the viewpoint of basic protocol, monitoring of only request/response messages are possible errors likely to occur in determination of service state. For example, while reception of a HTTP 200 OK response for a HTTP GET request means, from the viewpoint of this protocol, the protocol change has been successful, message body of HTTP 200 OK message can carry an error message belonging to the service itself in the background such as "Your credit limit is not sufficient to get this message". In cases like this, monitoring only the protocol messages yields misleading results with respect to entire state of the service.

Apart from this, in cases where service is using multiple protocols such as SMPP (Short Message Peer-to-peer Protocol), HTTP (Hypertext Transfer Protocol) and RTP (Real-time Transport Protocol), monitoring of each protocol transaction as a separate entity does not suffice alone either. In such cases, it is necessary that a service session should be monitored as a single entity together with all protocol transaction covered.

Existing monitoring methodologies and systems focus on monitoring of executive protocol messages which have separate priorities. That is, this kind of systems cover only protocol monitoring. They monitor only parsed protocol message exchanges by working out request/response peers of the protocols being monitored. It is monitored whether request has respectively arrived destination successfully and the response has been received by the source or not.

In conclusion, developments are being made in the service monitoring systems, therefore new embodiments eliminating the above disclosed disadvantages and will bringing solutions to the existing systems are needed.

Purpose of the Invention

The present invention relates to a service monitoring system, having an extensible service definition language, that monitors message exchanges between services, meeting above disclosed requirements, eliminating all of the disadvantages and bringing some additional advantages.

A purpose of the invention is to target and resolve via Extensible Service Definition Language (ESDL) an entire transaction, and what's more, clear problems caused unrelated protocol message as part of a service session.

Another purpose of the invention is mounting of every service used also to the system with ESDL. Each service definition realised with ESDL comprises at least one protocol transaction definition, each protocol transaction definition comprises at least one protocol transaction initial message and one protocol transaction termination message. Each protocol transaction definition may contain intermediate messages used as transition state within a transaction. Each protocol transaction definition may contain error messages needed to monitor service messages transmitted between service demander and service node.

Another purpose of the invention is that, each protocol defined with ESDL contains a set of regular expressions matching each protocol transaction with a definite state, in order to parse and handle protocol messages coming from ESDL engine.

Another purpose of the invention is that each protocol has a directory used to determine state of service sessions and to monitor sequence of actions in a service session.

Another purpose of the invention is to do service monitoring by analysing messages coming according to service definitions. To realise this, service definition of each service that will be monitored is composed. Service definitions are read-out from data base and are loaded to service definition objects. Depending on regular expressions of related service definitions, load carried over the message is parsed. The message which has been parsed depending on service definition model is analysed and transaction service state for each handled message is determined.

Another purpose of the invention is elimination of monitoring complexity of multiple message transaction, thanks to ESDL used. To achieve this, each process is regarded as a part of it and is saved to system data base.

Another purpose of the invention is elimination of requirement for fixed program coded service messages by means of ESDL utilisation. Since service definitions are supplied to the system in XML format via system console interface, these definitions can be changed via same interface. And ESDL engine auto-detects the service definition changed.

Another purpose of the invention is successful addition of an ESDL sample for each new service instantly, without a need for an additional software code. Since new service definitions are supplied to the system with XML format via system console interface, ESDL engine auto-detects the new service definition.

Another purpose of the invention is realisation of determining transaction state and service transaction state by system via handling a definite service as ESDL.

Another purpose of the invention is, on the other hand, storage of service sessions and transaction, in their own state, in a memory data based, saving each transaction and service session to system memory, with unique keys. These keys are used to search for and retrieve existing transaction and service sessions after a new message that needs to be associated with a transaction or a service session has been received.

The structural and characteristic properties and all of its advantages of the invention will be understood more clearly from the drawings provided below and from the detailed description written by making references to these drawings and therefore the assessment should be made by taking these drawings and the detailed description into account.

Brief Description of the Drawings

To understand the embodiment and the advantages of the present invention with the additional elements in the best way, it should be evaluated together with the figures whose descriptions are made below.

Figure 1 is a schematic view relating to an outlined sample of a service session consisting of two protocols able to make message exchanges (request/response).

Figure 2 a schematic view relating to service monitoring system's tapping communication media and its storing the messages to system data base is given.

Figure 3 a schematic view as to entire structure of service monitoring system, together with system interfaces and their internal and external communications/connections is given.

Figure 4 a flow chart of the sub system of message handling system is given.

Figure 5 is XML scheme definition which is used to make service definitions.

Figure 6 is a view relating to a sample service definition constituted by using service definition scheme.

The drawings need not absolutely be put to scale the details not necessary to understand the invention may have been omitted. Furthermore, the elements that are at least substantially identical, or at least have substantially identical functions are indicated with the same number.

Reference Numbers

100. Service session
102. User
104. Application 1
106. Application 2
108. Protocol Transaction
110. Protocol Transaction
112. Request message of Protocol A
114. Answer message of Protocol A
116. Request message of Protocol B
118. Answer message of Protocol B

200. User
202. Key connection port
204. Application 1
206. Key connection port
208. Application 2
210. Service monitoring system
212. System data base
214. Report client

300. Network TAP
302. Report client
304. Data base
306. System web console

308. Network tapping interface

310. Reporting interface

312. Data Access interface

314. System console interface

400. Message handling model

500. XML scheme definition

ESDL: Extensible Service Definition Language

XML: Extensible Markup Language

Detailed Description of the Invention

In this detailed description, the preferred embodiments of the service monitoring system (210) having configured extensible service definition language, which is the subject of the invention, are described only for a better understanding of the subject-matter and in a way not constituting any limiting effect.

Present invention relates to monitoring transaction-related service systems and providing required reports, based on data being monitored. This invention and configured Extensible Service Definition Language (ESDL) focus on monitoring of service data carried over related protocol messages.

Successful detection, control and correction of potential and/or emerged problems before their affecting critical commercial transaction is realised via service monitoring. With service monitoring, a system using transaction consisting of IP (Internet Protocol) based protocol messages to fulfil necessary operations of a service is reviewed incessantly. A transaction covers one or multiple basic protocol messages coming together to render a service requested. Here, the term transaction technically defines the systems consisting of Internet Protocol (IP9 based messages. These protocol messages can be carried over via utilisation of User Datagram Protocol (UDP) or Transmission Control Protocol (TPC).

However, in case that different protocols such as Signalling System #7 (SS7) and similar are converted into any IP based protocol messages, the system which is the subject of the invention and related ESDL defined in this invention can also monitor these protocols.

In Figure 1, a scheme outlining service session consisting of two protocols that can make an exchange of messages is given. The service monitoring system (210) disclosed in this invention accounts for tapping, parsing, recording of communication protocol messages (112,114,116,118) that form a transaction together. In order to constitute a service session (100), one or multiple protocol transaction (108,110) are assembled. A protocol message (112) can be sent and received between user (102) and application (104) or between two applications (104,106). A service session (100) is a Value Added Service typically recommended by Telecom operators or service providers to their customers, and are not solely limited to this.

It is strategically important that service providers monitor whether the transaction constituting their Value Added Service applications are accomplished in a successful way or not. In case transaction' being a failure, it is required that the reason for this problem is identified and highlighted before a plurality of customers are affected by the problem.

In Figure 2, a schematic view relating to service monitoring system's tapping the communications media and storing the messages to system data base is given. To provide and fulfil these functions, the system (210) which is the subject of the invention taps the physical communication line between the applications (204,208) and the user (200), via TAP devices or monitored key connection ports (202,206) and parses protocol messages. It determines transaction state by using ESDL definitions of the services monitored. It stores parsed messages at the data base (212) by associating them with processes and service sessions (100). Stored data can be used by service monitoring system (210) also in later times through retrieval from the data base. Transaction and service session (100) association/constitution logic is processed in real time and data become ready for querying while protocol messages are being changed with the real extreme points of service.

In addition, disclosed system (210) comprises a built-in reporting interface (310). By utilisation of said reporting interface (310), necessary real time reports can be constituted relating to service state and other associated data. With report client (214), query results can be requested over the reporting interface (310). The query

results, on the other hand, can be displayed in form of diagrams, tables and/or other formats.

If it should be explained with a more detailed description, the subject of the invention is dynamically making definitions to the service monitoring system (210) described in ESDL. Present invention and the system which is structured are a group of components combining transaction such as;

- Tapping related system transaction,
- Parsing transaction messages according to priority protocol/business rules,
- Associating the messages within protocol transaction messages and associating the transaction with services sessions,
- Saving processed data to permanent storage units for use in subsequent times,
- Creating reports based on saved data in order to build up various historical points of view of the system,
- Creating logs and/or alert to reflect real time state of the services monitored.

As disclosed above, the invention utilises a service definition model termed as Extensible Service Definition Language (ESDL). The system (210) which is the subject of the invention needs a different ESDL definition for each service. The reason for this is that each service is unique due to data realised by communication protocol messages of Value Added Services. Therefore, each service has a meaning and syntax relating to the messages constituting a transaction together. For example, both of two different service providers such as SP_A and SP_B use HTTP protocol to provide service to their customers. However, said service providers, despite using the same application layer protocol to carry over their service messages, message of each service provider is different in terms of both syntax and meaning.

SP_A service provider may use 'Weather Today' request message in its HTTP GET message to convey it to a client making a request for daily weather state . On the other hand, SP_B service provider may use "Stock exchange prices <CompanyName>' request message realised in HTTP GET message to convey latest stock Exchange price information relating to query requested. As it is seen,

although the application protocol (HTTP) is the same for both services, messages have different meanings.

Therefore, it is necessary to create a common model, to service monitoring system (210), that guides for learning the syntax and meanings of each service monitored. To provide this, present invention comprises a built-in ESDL and an appropriate transaction engine to parse, save and monitor services.

ESDL is an XML based service definition language defining a general model for service definitions. Each monitored service utilises this model file, which is also an XML file, to create its own ESDL sample. ESDL offers a name and a couple of protocols which basically constitute the service and a service model together. Each service comprises at least one protocol unit.

A protocol unit is matched with an application layer protocol whose messages together create a protocol transaction. Each protocol has a name and a directory. Said directory is utilised to monitor sequence of actions to create the service. Each ESDL definition contains a series of services message string and service name and other require information such as the protocols covered and short service numbers (at GSM). Service message strings written as Regular Expressions (REGEX). This state, on the other hand, provides definition elasticity of possible multiple messages by using REGEX.

Each protocol definition contains a message group matching with said Regular Expressions (REGEX) in order to create transaction. These regular expressions are atomic units used to analyse incoming messages. By using regular expressions, messages are analysed, each message is then associated with a protocol transaction associated with service sessions (100).

Each monitored service must have its own ESDL definition prepared by the system (210). This preparation transaction is realised via system web console (306). Recorded ESDLs are saved to system data base (212).

In Figure 3, system interfaces and their internal and/or external communications/connections as well as general structure of service monitoring system are given. The data received from Network TAP (300) that provides access to information flowing over a network is tapped via network tapping interface (308). The service monitoring system (210) which is the subject of the invention has certain interfaces in order to carry out its functions. Said network tapping interface (308) is utilised for interception of IP packets carrying service messages. Service messages can be loads which any application layer protocol carries. After receiving the packet from the system tapping line, it parses the message via appropriate protocol analysers and associates the message with a transaction, and then with a service. Afterwards, on the other hand, it stores this message in its data base (304).

Said system (210) stores the incoming service messages to the data base (304) by using RDMBS provided. While storing the messages, it utilises service definitions configured to associate the messages with the transaction and services. To carry out data base (304) transaction, a data access interface (312) working two- way with the data base (304).

The invention is a system console interface (314) that works with data Access interface (312) and allows the administrators to administer services, manage configurations, administrate panel and panel queries, monitoring system and subscriber activity by creating queries and making queries, and working in interaction with a system WEB console (306).

Reporting interface (310), on the other hand, provides successful reception of monitor queries and successful response to these queries inside the service and transaction data stored at the data base (304). In addition to the report request coming from the report client (302), a great number of reports relating to service and transaction used to monitor system services and their related transaction can be obtained from the system by means of the reporting interface (310).

In Figure 4, a flow chart belonging to message transaction sub system of service monitoring system (210) is seen. The operations steps concerning message transaction phase, is understood over this flow chart in a clear and net way. In the

pre-loading phase of the system (210), ESDL definitions being different for each service are read-out from the data base (304). This operation is carried out via a module termed as service loader. Responsible for administration of the services, this module is system administrator.

Stages that occur for handling (400) the message after this step are as follows:

- After pre-loading phase, service monitoring system (210) is ready to read-out service messages. Messages are taken from the network adapters, and are processed by protocol processors.
- After message is taken from the network, firstly protocol of the message is determined and appropriate protocol processor is retrieved. Protocol message is parsed to page title and the load carried and the carried load message is sent to ESDL engine.
- The message load and service definition arriving at the ESDL engine take carried load end passing through all ESDLs of services monitored in order to find match between REGEXes. XML indications of each service is read-out by the service provider and these XML data are converted to objects. Thus, ESDL engine utilises these service definition objects.
- In the event that a match is found by ESDL engine, the engine identifies transaction and service state of the incoming message and sends state object to protocol processor.
- ESDL engine identifies state of each transaction and service session (100) to which the incoming message belongs. To carry out this step, a transaction state factory object is used. The transaction state factory used to realise this works with this logic:

Since same service session (100) for multiple user occurs as plural, it is not possible to determine when a message belonging to a certain state will be received. Therefore, transaction state factory works by putting detailed emphasis on all service definitions. When it is looped via all defined services, it concurrently utilises an inside loop for protocol definitions of each service definition. Since likelihood of arrival of the transaction initiation messages in inside loop is higher than the other messages, it controls these messages first.

Then it controls intermediate state messages, and then the phase end and transaction end messages. What is controlled last are message type error messages. During control and determination loop, in case that message content matches with any one of the ESDL defined messages, factory creates a transaction state object for the processed incoming message and stops the transaction state determination process for this message. The created transaction state comprises, at the same time, the state of the service session (100) to which this definite transaction belongs.

- Incoming and processed message processed after transaction and service session state are determined is associated with transaction and service. State of transaction and service are changed in memory map and data base (304). Finally, message is written to system data base (212).
- Report relating to results can be obtained via querying the system by message reporting clients (214,302) comprising its own main transaction and service session, after running of this transaction model.

Present invention has, in its system memory, a memory built-in data base (304) to save each protocol process (108,110) and service session (100). With administration of state s in memory data base, message transaction performance is increased.

Described transaction methodology provides successful storage of service messages in form of transaction (108,110) and service sessions (100) as messaging between parties occurs. Since messages are associated with real time, monitoring queries can be achieved instantly then on the data saved In addition, query runtime is significantly shortened with this transaction and replies are created very quickly. Quick query creating capability allows provision of real time panel images. However, identification of said problem is possible as fast as possible before the problem affects a great number of users.

In Figure 5, XML schemes (500) wherein service definitions are made are found. In Figure 6, on the other hand, a sample service definition created by using this definition scheme (500) is contained. This definition (500) comprises name of service and definitions of the protocols used in this service. Each protocol definition, on the

other hand, specifies, protocol's name and definitions carried by utilisation of protocol, to what phase of related process they belong.

CLAIMS

1. The invention is a service monitoring system (210) that targets and solves clear problems posed by unrelated protocol message as part of an entire system (108,110) or even a service session, used in definition of each service monitored, comprising an extensible service definition language (ESDL) eliminating monitoring complexity of multiple message transaction (108,110) via service definition created for each service definition, and a service monitoring system (210) having a suitable ESDL engine which processes data, and it is characterised in that; it carries out the steps
   - tapping reciprocal system transaction,
   - mounting services to the system via utilisation of said extensible service definition language (ESDL),
   - parsing the transaction messages according to priority protocol/business rules,
   - associating the messages with the protocol transaction (108,110) messages, while the transaction with the service sessions (100),
   - saving processed data to storage unit for use in future times,
   - creating report based on the data saved to the system (210),
   - creating directory and/or alerts to reflect real time state of the services monitored.
2. Service monitoring system (210) according to Claim 1, and it is characterised in that; IP packets carrying service messages are intercepted via network tapping interface (308), received packet is parsed via appropriate protocol analysers, message is associated first with a transaction (108, 110), then with a service and said message is saved in the data base (304).
3. Service monitoring system (210) according to Claim 1, and it is characterised in that; data base (304) transaction are realised during storage of data to the data base (304) via application of data access facade (312).
4. Service monitoring system (210) according to Claim 1, and it is characterised in that; system (210) and subscriber activity are monitored by means of

administration of services, configurations, panel and panel queries via console interface (314), creation of a query and realisation of the query.

5. Service monitoring system (210) according to Claim 1, and it is characterised in that; screen queries are received via reporting interface (310), these queries are replied within the service and transaction data stored in data base (304), report is given relating to the services and transaction (108,110).
6. Service monitoring system (210) according to Claim 5, and it is characterised in that; query results can be displayed in form of diagrams, tables and/or other required formats.
7. Service monitoring system (210) according to Claim 1, and it is characterised in that; physical communication line is tapped via key connection ports (202, 206) monitored or TAP devices in order to ensure tapping of protocol messages (112,114,116,118) that form a transaction when they get together.
8. Service monitoring system (210) according to Claim 1, and it is characterised in that; for each service monitored, its own ESDL definition is prepared by the system (210) via system web console (306).
9. Service monitoring system (210) according to Claim 1, and it is characterised in that; with ESDL
   - Each service definition comprises at least one protocol transaction (108,110) definition,
   - Each protocol transaction (108,110) definition is defined in a way to comprise at least one protocol transaction initial message (112) and at least one transaction termination message (114).
10. Service monitoring system (210) according to Claim 1, and it is characterised in that; via a set of regular expressions, protocol messages coming from ESDL engine are parsed and processed, each protocol transaction (108,110) is matched to a definite state.
11. Service monitoring system (210) according to Claim 1, and it is characterised in that; via a directory, state of service sessions (100) are determined and each protocol follows operations step in a service session (100).
12. Service monitoring system (210) according to Claim 1, and it is characterised in that; message analysis coming according to service definitions is made via the operations of

- Creating a service definition, via ESDL, for each service to be monitored ,
- Reading out service definitions from data base (212) and loading them to service definition objects,
- Parsing message carrying load is parsed according to regular expressions of related service definitions,
- Analysing the message parsed according to service definition model,
- Determining transaction and service state for each message.

13. Service monitoring system (210) according to Claim 1, and it is characterised in that; message transaction is carried out with the operation steps of

- Reading out ESDLs created for different services from data base (304) and loading them to system memory,
- Transaction messages received from Network TAP (300) by protocol processors,
- Determining message protocol and retrieving appropriate protocol processor,
- Parsing protocol message as page title and carried load, and sending carried load message to ESDL engine,
- ESDL engine's taking the carried load end passing through all ESDLs of the monitored services in order to find match between incoming message load and regular expressions (REGEX),
- Finding of a match by ESDL engine and said engine's identifying transaction and service state of the incoming message.

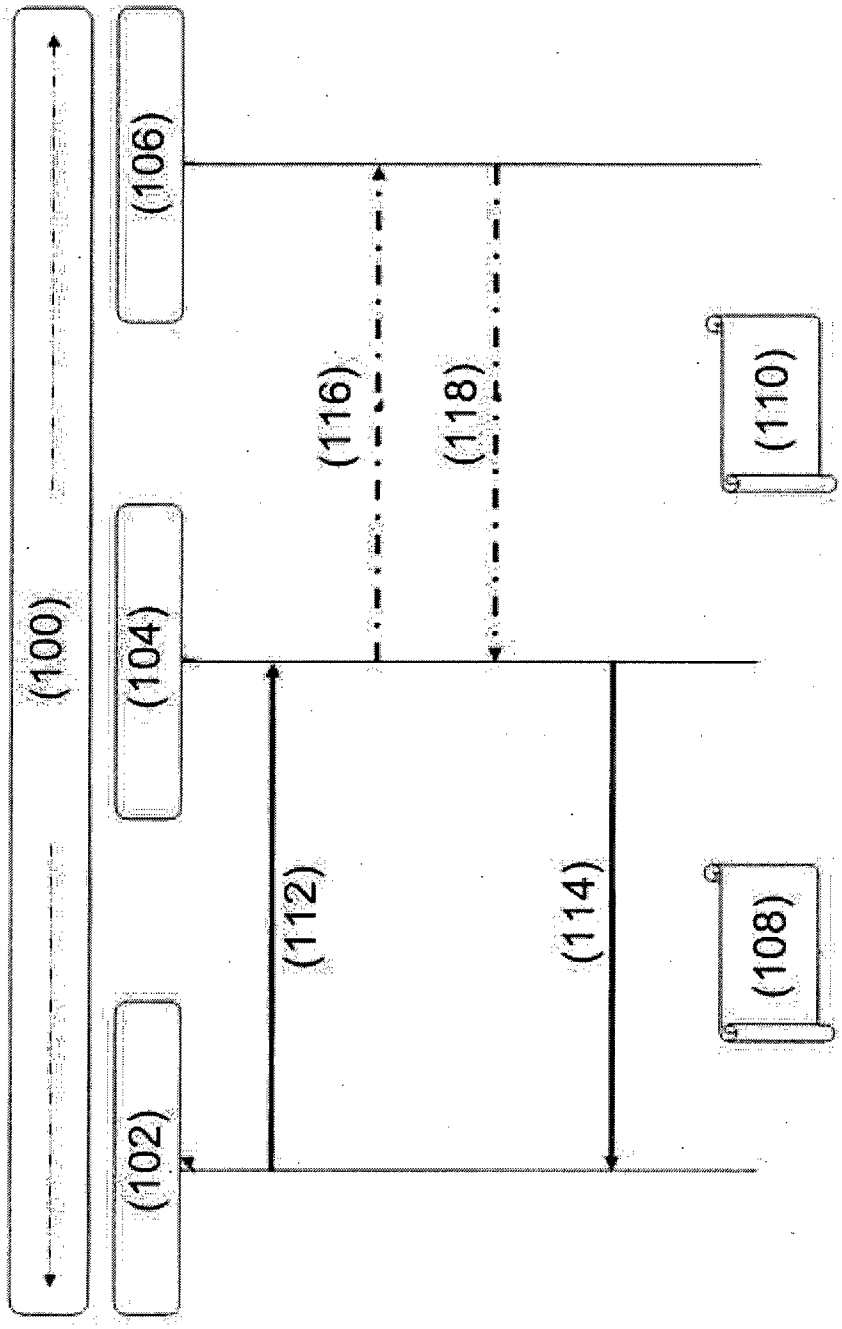

Figure 1

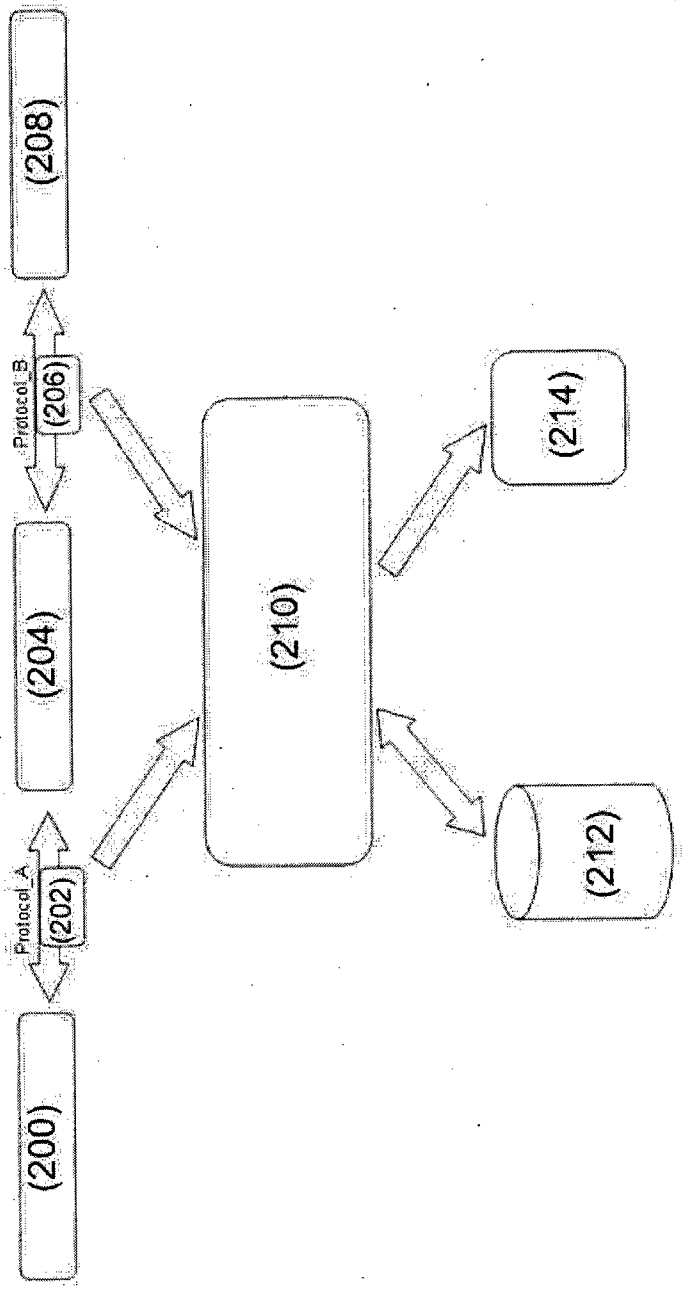

Figure 2

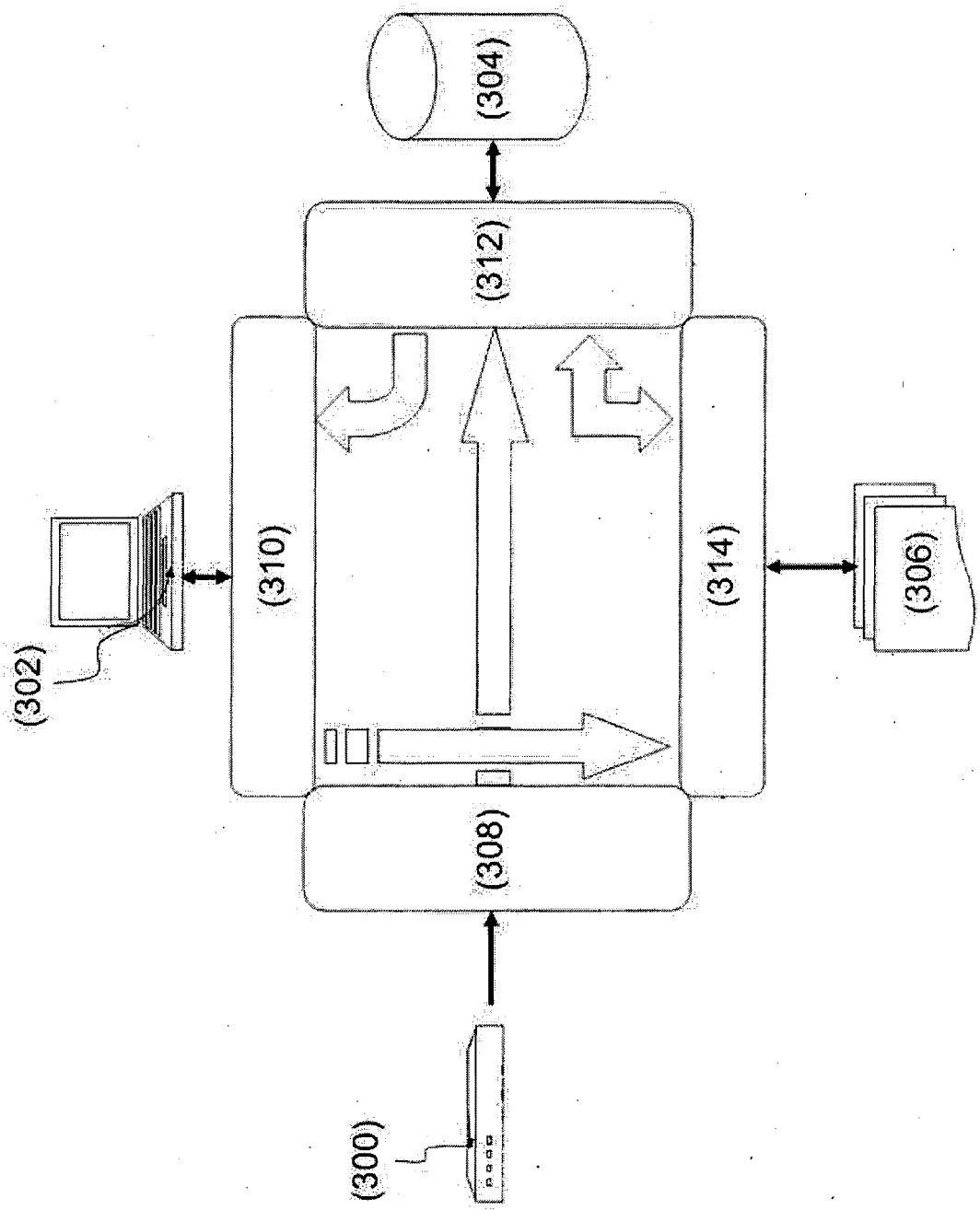

Figure 3

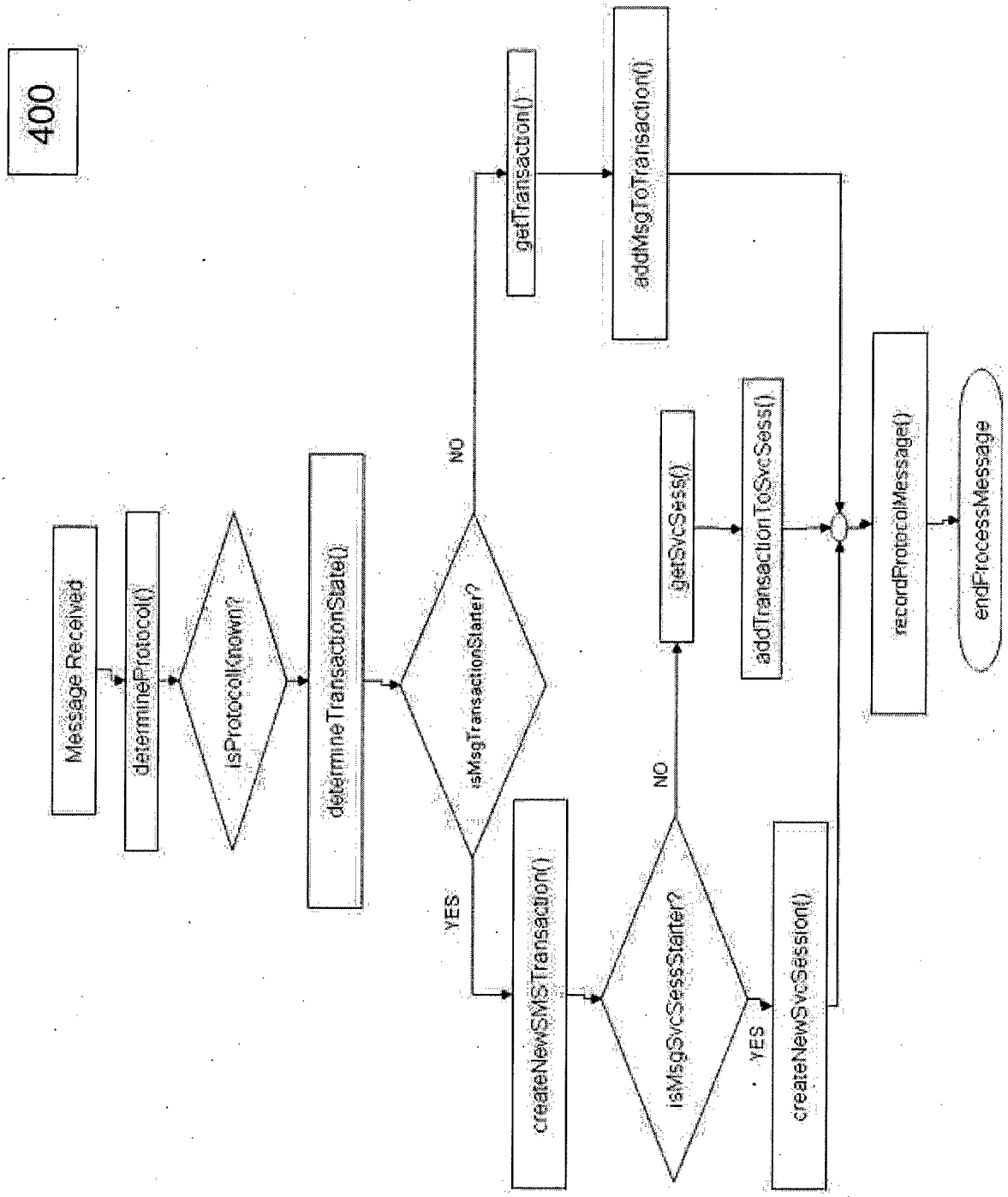
400
Message Received
determineProtocol()
isProtocolKnown?
determineTransactionState()
isMsgTransactionStarter?
YES
NO
createNewSMSTransaction()
getTransaction()
addMsgToTransaction()
isMsgSvcSessStarter?
YES
NO
createNewSvcSession()
getSvcSess()
addTransactionToSvcSess()
recordProtocolMessage()
endProcessMessage

Figure 4


```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:jxb="http://java.sun.com/xml/ns/jaxb"
        jxb:version="2.1">
            <xsd:element name="ServiceDefinition" type="ServiceDefinition"></xsd:element>
            <xsd:complexType name="ServiceDefinition">
                  <xsd:sequence>
                     <xsd:element name="Protocol" type="Protocol" minOccurs="1" maxOccurs="unbounded"></xsd:element>
                  </xsd:sequence>
                  <xsd:attribute name="ServiceName" type="xsd:string"></xsd:attribute>
            </xsd:complexType>

            <xsd:complexType name="Protocol">
              <xsd:sequence>
                  <xsd:element name="phase1ServiceNumbers" type="xsd:string" minOccurs="1" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase1StartMessages" type="xsd:string" minOccurs="1" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase1MiddleMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase1EndMessages" type="xsd:string" minOccurs="1" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase1EndTransactionEndMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase2ServiceNumbers" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase2StartMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase2MiddleMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase2EndMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="phase2EndTransactionEndMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="transactionEndMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="protocolSystemErrorMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="protocolBusinessErrorMessages" type="xsd:string" minOccurs="0"maxOccurs="unbounded"></xsd:element>
                  <xsd:element name="protocolUserErrorMessages" type="xsd:string" minOccurs="0" maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
              <xsd:attribute name="ProtocolName" type="xsd:string"></xsd:attribute>
              <xsd:attribute name="ProtocolIndex" type="xsd:string"></xsd:attribute>
            </xsd:complexType>
</xsd:schema>
```

Figure 5

500


```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceDefinition ServiceName="MEETING" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="Service_def.xsd">
 <Protocol ProtocolIndex="0" ProtocolName="SMPP">
  <phase1ServiceNumbers>1234</phase1ServiceNumbers>
  <phase1StartMessages>Hi</phase1StartMessages>
  <phase1MiddleMessages>How are you?</phase1MiddleMessages>
  <phase1EndMessages>Fine and you?</phase1EndMessages>
  <phase1EndTransactionEndMessages>Me too</phase1EndTransactionEndMessages>
  <phase2ServiceNumbers>5678</phase2ServiceNumbers>
  <phase2StartMessages>Where are you from?</phase2StartMessages>
  <phase2MiddleMessages>I am from world</phase2MiddleMessages>
  <phase2EndMessages>Nice</phase2EndMessages>
  <phase2EndTransactionEndMessages>Let's eat something</phase2EndTransactionEndMessages>
  <transactionEndMessages>OK</transactionEndMessages>
  <protocolSystemErrorMessages>Your request can not be processed due to System Error</protocolSystemErrorMessages>
  <protocolBusinessErrorMessages>Your request can not be processed due to Business Error</protocolBusinessErrorMessages>
  <protocolUserErrorMessages>Your request can not be processed due to User Error</protocolUserErrorMessages>
 </Protocol>
 <Protocol ProtocolIndex="-1" ProtocolName="HTTP">
  <phase1ServiceNumbers>[NONE]</phase1ServiceNumbers>
  <phase1StartMessages>Here is a good place to eat</phase1StartMessages>
  <phase1MiddleMessages>Yes it is</phase1MiddleMessages>
  <phase1EndMessages>Let's eat here then</phase1EndMessages>
  <phase1EndTransactionEndMessages>OK</phase1EndTransactionEndMessages>
  <phase2ServiceNumbers>[NONE]</phase2ServiceNumbers>
  <phase2StartMessages>What will you have</phase2StartMessages>
  <phase2MiddleMessages>I would like chicken</phase2MiddleMessages>
  <phase2EndMessages>Me too</phase2EndMessages>
  <phase2EndTransactionEndMessages>And some potatoes</phase2EndTransactionEndMessages>
  <transactionEndMessages>That is fine</transactionEndMessages>
  <protocolSystemErrorMessages>Your request can not be processed due to System Error</protocolSystemErrorMessages>
  <protocolBusinessErrorMessages>Your request can not be processed due to Business Error</protocolBusinessErrorMessages>
  <protocolUserErrorMessages>Your request can not be processed due to User Error</protocolUserErrorMessages>
 </Protocol>
</ServiceDefinition>
```


Figure 6