Web Services-based Charging Control in Multimedia Networks

Ivaylo I. Atanasov, Evelina N. Pencheva, Dora A. Marinska

Abstract**—**Parlay X is a set of Web Services interfaces for open access to network function. One of the alternatives for deployment of Parlay X functionality is an add-on to the Open Service Access (OSA) functionality. The OSA gateway is a special type of application server that translates the OSA interfaces in the network control protocols. The paper suggests a mapping of OSA Charging interfaces onto Diameter protocol messages. A formal approach to specification of OSA gateway behavior is suggested. The approach is based on the theory of Labeled Transition Systems and the concept of bisimilarity. Using the available mapping of Parlay X interfaces onto OSA interfaces, the possibilities of Parlay X charging control on sessions in all IP-based multimedia networks are investigated.

Keywords**—**Service Oriented Architecture, Application Programming Interfaces, Charging, Interface to protocol mapping, Labelled Transition Systems, Behavioural equivalence

I. Introduction

OPEINING the network using application programming interfaces (APIs) allows third party applications to access communication capabilities in public networks. A standardized architecture for open access to telecom functions in mobile networks is Parlay/Open Service Access (OSA) [1]. The Parlay/OSA APIs expose communication functions defined as Service Capability Features (SCFs) in a neutral way for both the network technology and the programming language aspects [2, 3]. Having common programmability paradigm for mobile, fixed, and managed packet networks, the APIs provide a medium level of abstraction of the network capabilities [4]. The APIs are an abstraction from different specific protocol stacks, but the abstraction level of Parlay/OSA APIs is not oriented to what we call traditional group of web developers and this could affect usability. In order to make the accessibility of the network capabilities available to a much wider audience Parlay X provides a set of high level interfaces that are oriented towards the skills of the web developers.

The Parlay X [5] is the name standing for a set of interfaces allowing access to Parlay/OSA APIs via Web Services. In addition to being Web Service interfaces, the Parlay X interfaces are much simplified presentation of Parlay/OSA APIs. A typical Parlay X Web Services deployment model allows publication of Parlay X Web Services through a registry, making those Web Services available for discovery. The applications can use Web Services' access operations within the Parlay X gateway, where the Web Services interfaces are implemented.

The research is partially funded by the Project DO-02-135/2008, Research on Cross Layer Optimization of Telecommunication Resource Allocation, National Science Fund, Bulgarian Ministry of Education, Youth and Science.

There exist different deployment scenarios for Parlay X Web Services. One possible scenario addresses solutions where the Parlay X functionality is an add-on to the Parlay/OSA functionality. Thus, the Parlay X gateway is connected to OSA gateway through Parlay/OSA interfaces, forming a kind of interface wrapping. Another scenario for Parlay X deployment addresses hybrid solutions which combine Web Service interfaces and network protocols. The Parlay X gateway attaches to the network through an interface defined by the corresponding network element and provides interoperability between Parlay X interfaces and network control protocols [6].

In this paper, we study deployment issues of Parlay X interfaces in all IP based multimedia networks [7]. The focus is on Parlay X interfaces that support reservations, pre-paid and post-paid types of payment, and their mapping onto Diameter protocol. Based on the available mapping of Parlay X interfaces onto OSA APIs we consider alternatives for interoperability between the Parlay X Payment Web Service, the OSA Charging interfaces and the Diameter protocol stack. We suggest a formal approach to specification of the OSA gateway behavior which may be useful in design and implementation of OSA gateway supporting charging interfaces. The approach supports the conformance testing aimed to check weather an implementation of OSA gateway conforms to its specification.

The paper is organized as follows. In Section II, some aspects of Parlay X deployment in all IP-based multimedia networks are discussed. The Parlay X Web service and OSA API that provide access to charging functions in telecommunication networks are presented in brief. Section III presents the suggested functional mapping between the OSA Charging SCF interfaces and Diameter protocol used for signaling related to charging in managed IP-based multimedia networks. Formal description of the finite state machine representing charging session handling and the Diameter peer state machine is provided in Section IV, where the state machines' behavioral equivalence is proved. An example of Parlay X payment application is presented before concluding the paper.

II. Open Access to Charging Control in Multimedia Networks

A. Parlay X Deployment in IMS Architecture

Internet Protocol Multimedia Subsystem (IMS) is service control architecture intended to provide all types of multimedia services based on IP connectivity [7]. In the IMS control architecture, Application Servers run 3rd party applications. The information published on the Web Services Registry provides Applications with information required to connect with the Parlay X Gateway. Applications utilize Web Services to access and interact with the network, and have not visibility to the OSA implementation behind the Parlay X Web Services Gateway. The Parlay X Web Services Gateway attaches to the OSA Gateway through an OSA interface. The OSA Gateway is a special type of Application Server which provides interoperability between third party control and network functions. The OSA Gateway communicates with Serving Call Session Control Function (S-CSCF) responsible for users' registrations and session management relying on Session Initiation Protocol (SIP) signaling. The Home Subscriber Server (HSS) is a database which stores the users' profiles. The access to user data in the HSS is based on Diameter signaling. The Online Charging System is responsible for the process through which the real time charging information can affect the services rendered and, therefore, directly interacts with session/service control. The IMS online charging capable entities send charging information to Online Charging Function (OCF) via Diameter based interface [8].

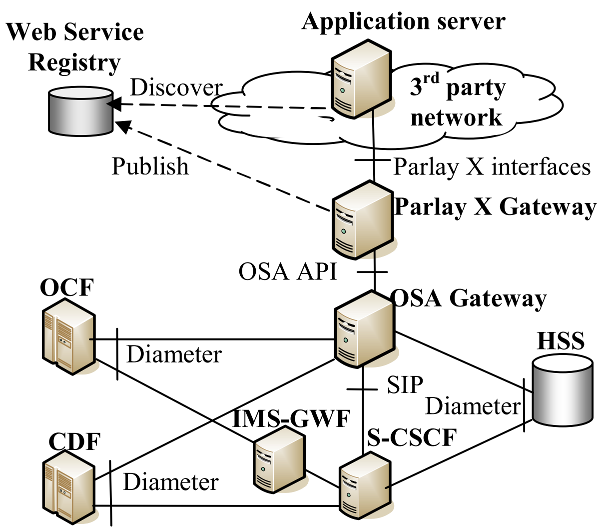

Fig.1 Open access to charging control in IMS

The OCF performs both event- and session-based charging, and also credit control. The S-CSCF interacts with the OCF via IMS Gateway Function (IMS-GWF). The offline charging system supports the traditional charging model, i.e. the charging information is collected over a particular period and, at the end of the period, it posts a bill to the customer's account. The IMS offline charging capable entities collect charging information and send it to the Charging Data Function (CDF) via Diameter. The CDF constructs the Call Detail Records and sends them to the billing system via Charging Gateway. Fig.1 shows the IMS functional entities considered in the context of charging and session control.

B. Abstraction of Charging Functions through Open Interfaces

As to [9], the Payment Web Service supports functions that allow the application to:

- Charge/refund an account by volume/amount.
- Calculate an amount from a volume for a specific account.
- Reserve an amount/volume on an account.
- Charge a prior reservation to the account.
- Release a reservation by returning to an account the amount/volume remaining in a reservation.

The Payment Web Service interfaces may be used for creation of quality of service based charging [10].

OSA Charging SCF APIs [11] provide methods for charging session control supporting crediting and debiting user accounts with specified amounts in number of monetary or non-monetary units. User accounts may be credited and debited directly or on reservation basis.

Parlay X Application can access charging functions just by invoking operations of Payment Web Services. The invocation of Web Service operation opens an OSA charging session. The OSA charging session handling is determined by invocation of OSA Charging interface methods. The OSA gateway maintains a finite state machine representing the charging session handling. Each charging session opens a Diameter dialogue and therefore the OSA gateway needs to maintain also Diameter peer state machine as shown in Fig.2. A functional model requires both state machines to be synchronized i.e. to expose equivalent behavior.

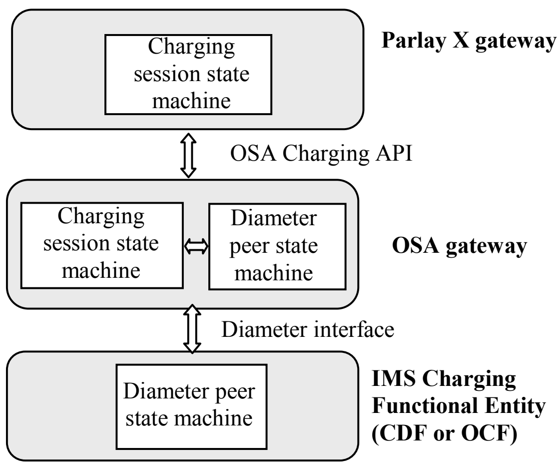

Fig.2 Interoperability behavioral model of OSA gateway

The Parlay X gateway converts Web services operations into OSA interface methods invocation and vice versa. The mapping of Parlay X Payment Web Services onto Parlay/OSA Charging API is defined in [12]. No mapping is defined for OSA Charging API onto Diameter protocol. The Diameter base protocol that has to be implemented by all Diameter applications is defined in [11]. The Diameter charging application used for charging functionality and charging management in IMS is specified [13].

In a Web Services to network element deployment scenario it is the Parlay X gateway that makes the transformation between Parlay X operations and the Diameter protocol operations [14].

In the next section, we define a functional mapping between the OSA Charging SCF interface methods [11] and the Diameter messages defined in [13], [14].

III. Mapping of OSA Charging Interfaces onto Diameter Protocol

The OSA methods setCallback() and setCallbackWith-SessionID() are specific for the OSA framework interface and they do not have corresponding Diameter messages.

The OSA IpChargingManager interface is the service manager interface for the Charging Service. The application programmer can use this interface to start charging sessions.

The OSA methods createChargingSession() and createSplitChargingSession() of the IpChargingManager interface are used to create an instance of the IpCharging-Session interface to handle the charging events related to the specified user or users. These methods are mapped onto Start event by which the Diameter application initiates a connection with the peer.

Fig.3 shows the message sequence related to creation of OSA charging session and opening a Diameter dialogue with the Diameter peer.

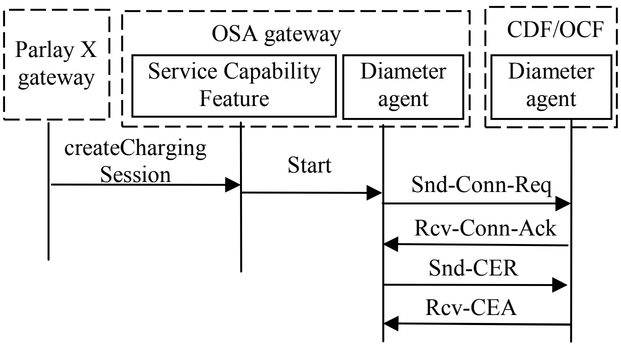

Fig.3 Creation of OSA charging session and opening a Diameter dialogue

The OSA methods sessionAborted() and abortMultiple-ChargingSessions() of the IpAppChargingManager interfaces are used to indicate to the application that the charging session (sessions) has aborted abnormally. These methods are mapped onto the Diameter Peer-Disc message with which the Diameter peer is signaled that the transport layer connection is disconnected.

Fig.4 shows the indication that the charging session object (at the gateway) has aborted or terminated abnormally due to disconnection of the transport layer connection.

The OSA IpAppChargingSession interface is the callback application interface for the Charging Service. It provides the application with charging session management functions.

The OSA method sessionEnded() of the IpAppCharging-Session interface indicates to the application that the charging session has terminated in the charging server and it is mapped onto Diameter Peer-Disc message.

The OSA method release() of the IpChargingSession interface terminates the session and it is mapped onto Stop event by which the Diameter application signals that the dialogue should be terminated.

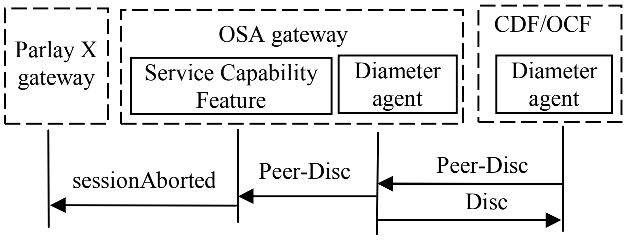

Fig.4 Indication about Diameter dialogue and charging session termination

Fig.5 shows the message sequence in case the application releases the charging session.

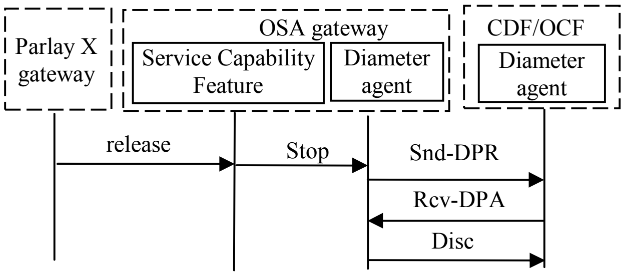

Fig.5 Release of OSA charging session and closing Diameter dialogue

The OSA methods creditAmountReq(), creditUnitReq(), debitAmountReq(), debitUnitReq(), directCreditAmountReq(), directCreditUnitReq(), directDebitAmountReq(), directDebit-UnitReq(), reserveAmountReq(), reserveUnitReq(), rateReq(), extendLifeTimeReq(), getAmountLeft(), getLifeTimeLeft(), and getUnitLeft() of the IpChargingSession interface require actions to be taken in the network and these methods are mapped onto the Diameter message that has to be sent (noted as Send-Message in the Diameter peer state machine [13]). The Diameter messages that have to be sent are Diameter application specific and for Diameter Charging Application [14] these messages may be Accounting Request (ACR) in case of offline charging or Credit Control Request (CCR) in case of online charging. The OSA Charging gateway determines the charging model based on data in the user profile. In IMS, the application server (OSA gateway) accesses user data stored in the HSS through the Diameter based interface (not regarded in this paper).

The OSA methods creditAmountRes(), creditUnitRes(), debitAmountRes(), debitUnitRes(), directCreditAmountRes(), directCreditUnitRes(), directDebitAmountRes(), directDebitUnitRes(), reserveAmountRes(), reserveUnitRes(), rateRes(), and extendLifeTimeRes() of the IpAppChargingSession interface are used to indicate that the corresponding request was successful and these methods are mapped onto the Diameter message that has to be received (noted as Rcv-Message in the Diameter peer state machine [13]). The Diameter messages that have to be received for the Diameter Charging applications [14] are Accounting Answer (ACA) in case of offline charging or Credit Control Answer (CCA) in case of online charging.

Fig.6 shows the message sequence for crediting an amount towards the reservation associated with the session in case of online charging. The amount left in the reservation will be increased by this amount.

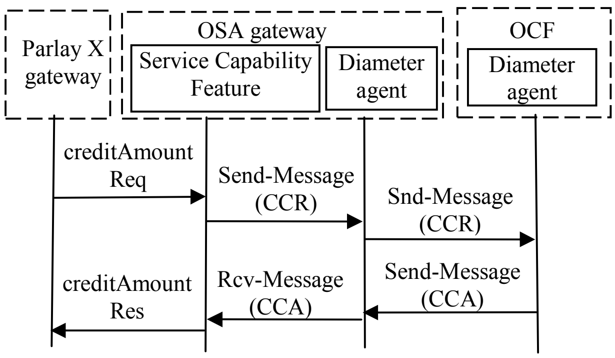

Fig.6 Crediting of user's account

Fig.7 shows the message sequence for debiting a volume of application usage towards the user in case of offline charging. Units may be measured for example in minutes or kilobytes. The volumes in a possible reservation associated with this session are not influenced.

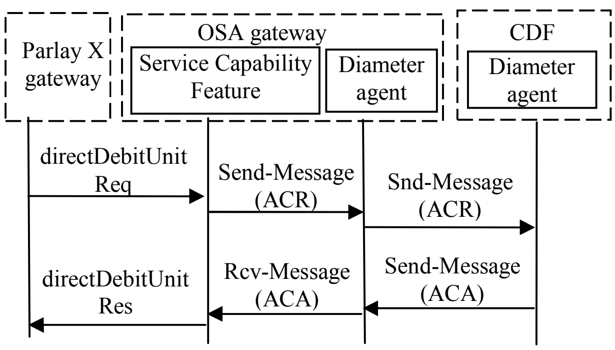

Fig.7. Direct debiting of user's account with a number of units

The OSA gateway can examine the result code parameter in CCA or ACA Diameter messages to report any failures to the application by invoking the corresponding OSA method including creditAmountErr(), creditUnitErr(), debitAmountErr(), debitUnitErr(), directCreditAmountrr(), directCreditUnitErr(), directDebitAmountErr(), directDebitUnitErr(), reserveAmount-Err(), reserveUnitErr(), rateErr(), and extendLifeTimeErr() of the IpAppChargingSession interface. These methods are also used to report errors, such as unavailable peer or invalid route specification. The OSA gateway can use a Diameter timer value to determine, for example, when the OCF fails to respond to a credit authorization request. The timer is set when a CCR is sent to the OCF. The timer resets when the corresponding CCA is received.

Fig.8 shows the message sequence when the OSA gateway indicates to the application that the corresponding request failed completely and that no units have been debited

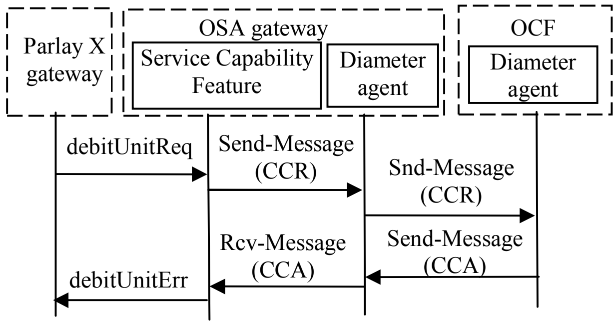

Fig.8 Indication that the corresponding request failed completely (no units have been debited)

IV. Formal Specification of Charging Model

Formal verification of OSA gateway is based on mathematical methods that examine the state space of OSA charging session handling and Diameter peer and verify whether the required properties are met. Specification and verification of concurrent systems are complex tasks because the number of states can increase exponentially with the number of parallel branches [15]. The formal approaches use modeling languages to system description, specification languages to express the required correct system behavior and provide analysis techniques [16, 17]. In this section, we exploit the formalism of Labeled Transition Systems to prove the behavioral equivalence and hence the conformance of OSA charging model and IMS charging model. This may be used for automatic generation of test cases during the OSA gateway verification.

A. Labeled Transition Systems and Behavioral Equivalence

To prove behavioral equivalence of finite state machines formally, the notion of *Labeled Transition Systems* is used [18].

Definition 1: A *Labeled Transition System* (LTS) is a quadruple $(S, Act, \rightarrow, s_0)$, where S is countable set of states, Act is a countable set of elementary actions, $\rightarrow \subseteq S \times Act \times S$ is a set of transitions, and $s_0 \in S$ is the set of initial states.

We will use the following notations:

- $s \xrightarrow{a} s'$ stands for the transition (s, a, s');
- $s \xrightarrow{a}$ means that $\exists s': s \xrightarrow{a} s'$;
- $s \xRightarrow{\mu} s_n$, where $\mu = a_1, a_2, ..., a_n : \exists s_1, s_2, \ldots, s_n$,

such that $s \xrightarrow{a_1} s_1 \ldots \xrightarrow{a_n} s_n$;

- $s \xRightarrow{\mu}$ means that $\exists s'$, such as $s \xRightarrow{\mu} s'$;
- $\xRightarrow{\hat{\mu}}$ means $\Rightarrow$ if $\mu \equiv \tau$ or $\xRightarrow{\mu}$ otherwise,

where τ is one or more internal (invisible) actions.

The concept of *bisimulation* [19] is used to prove that two labeled transition systems expose equivalent behavior. The strong bisimulation possesses strong conditions for equivalence which are not always required. For example, there may be internal activities that are not observable. The strong bisimulation ignores the internal transitions.

Definition 2: [19] Two labeled transition systems $T = (S, A, \rightarrow, s_0)$ and $T' = (S', A, \rightarrow', s_0')$ are *weekly bisimilar* if there is a binary relation $U \subseteq S \times S'$ such that if $s_1\ U\ t_1 : s_1 \subseteq S$ and $t_1 \subseteq S'$ then $\forall a \in Act$:

- $s_1 \xRightarrow{a} s_2$ implies $\exists\ t_2 : t_1 \xRightarrow{\hat{a}}{}' t_2$ and $s_2\ U\ t_2$;
- $t_1 \xRightarrow{a}{}' t_2$ implies $\exists\ s_2$: $s_1 \xRightarrow{\hat{a}} s_2$ and $s_2\ U\ t_2$.

B. Charging Session Handling Described as Labeled Transition System

The OSA charging session handling is described in [11] as a finite state machine in five states and is shown in Fig.9.

In Null state the Charging Session does not exist. In Session Created state, the Charging session is created, no reservations have been made, and the application has the possibility to perform direct debits and credits on the user's account and to request rating. In Amount Reserved State and Volume Reserved State, a reservation has been made for a certain maximum amount or certain maximum volume respectively. In these states, the application has the possibility to perform incremental debits/credits on this reserved amount/volume until either the application chooses to close the reservation or the reservation limit is reached, or the charging session is released. The application can also extend the reservation. In Reservation Ended state the reservation has been closed by the application, or the reservation limit has been reached. The charging session may remain active in order to carry out non-reservation related tasks such as direct credit or debit operations.

We describe formally the charging session handling as seen by OSA charging application as a LTS which is intentionally simplified. The actions correspond to the method invocations.

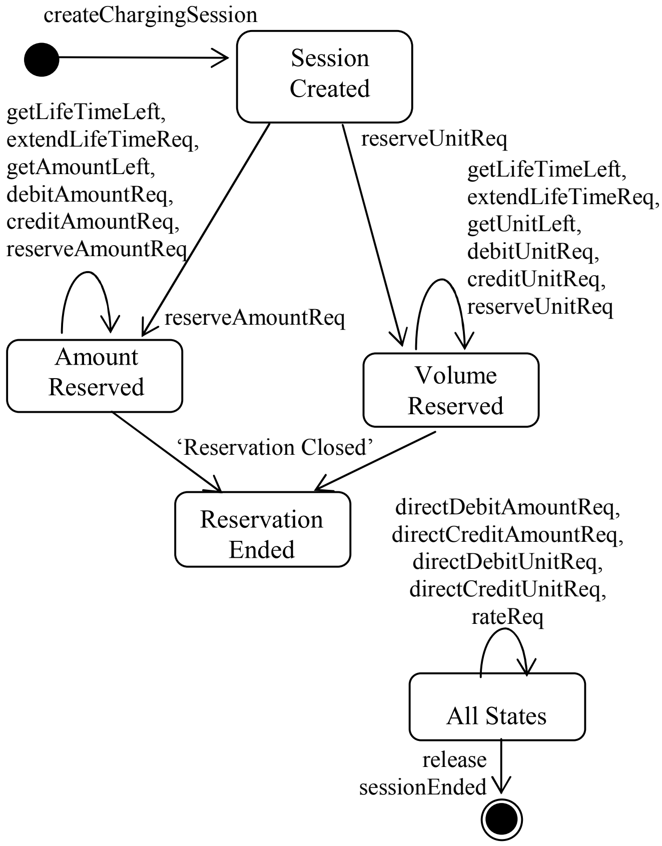

Fig. 9 OSA application view of Charging Session object states

To avoid exhausting repetition of similar expressions in the definition below we use the following notations:

- commonList denotes any of the following methods: directDebitAmountReq, directCreditAmountReq, directDebitUnitReq, directCreditUnitReq, rateReq
- amountList denotes any of the following methods: getAmountLeft, debitAmountReq, creditAmountReq, reserveAmountReq
- volumeList denotes any of the following methods: getUnitLeft, debitUnitReq, creditUnitReq, reserveUnitReq.

The transition to the Reservation Ended state as a result of application initiated reservation closing or reaching the original reservation limit is represented by the action reservationClosed.

By $T_{AppCSH} = (S_{AppCSH}, Act_{AppCSH}, \rightarrow_{AppCSH}, s_0')$ we denote a LTS representing the OSA charging session handling where:

- S_{AppCSH} ={ Null, SessionCreated, AmountReserved, VolumeReserved, ReservationEnded};
- Act_{AppCSH} = {createChargingSession, reserveAmountReq, reserveUnitReq, getAmountLeft, debitAmountReq, creditAmountReq, getUnitLeft, debitUnitReq, creditUnitReq, directDebitAmountReq, directCreditAmountReq, directDebitUnitReq, directCreditUnitReq, rateReq, reservationClosed, release, sessionEnded};
- $\rightarrow_{AppCSH}$ = {Null createChargingSession SessionCreated, SessionCreated commonList SessionCreated, SessionCreated release Null,

SessionCreated sessionEnded Null,
SessionCreated reserveAmountReq AmountReserved,
AmountReserved amountList AmountReserved,
AmountReserved commonList AmountReserved,
AmountReserved reservationClosed ReservationEnded,
AmountReserved release Null,
AmountReserved sessionEnded Null,
SessionCreated reserveUnitReq VolumeReserved,
VolumeReserved volumeList VolumeReserved,
VolumeReserved commonList VolumeReserved,
VolumeReserved reservationClosed ReservationEnded,
VolumeReserved release Null,
VolumeReserved sessionEnded Null,
ReservationEnded commonList ReservationEnded,
ReservationEnded release Null,
ReservationEnded sessionEnded Null};

- $s_{0'}$ = {Null}.

C. Diameter Peer State Machine Described as Labeled Transition System

The Diameter peer state machine is defined in [13]. For methods invoked by the application, the OSA gateway behaves as Diameter client agent, while for notifications the OSA gateway behaves as Diameter server agent.

By $T_{Diameter} = (S_{Diameter}, Act_{Diameter}, \rightarrow_{Diameter}, s_0)$ it is denoted a LTS which represents a simplified Diameter peer state machine where:

- $S_{Diameter}$ ={Closed, Wait-Conn-Ack, Wait-I-CEA, Open, Closing};
- $Act_{Diameter}$ = {Start, Rcv-Conn-Ack, Rcv-CEA, Send-Message, Rcv-Message, Stop, Rcv-DPA, Peer-Disc};
- $\rightarrow_{Diameter}$ = {Closed Start Wait-Conn-Ack,
Wait-Conn-Ack Rcv-Conn-Ack Wait-I-CEA,
Wait-I-CEA Rcv-CEA Open,
Open Send-Message Open,
Open Rcv-Message Open,
Open Stop Closing,
Closing Rcv-DPA Closed,
Open Peer-Disc Closed};
- s_0 = {Closed}.

D. Conformance of OSA Charging model and IMS charging model

To prove the interoperability between charging models in OSA and IMS we have to prove that state machine representing the OSA charging session handling and Diameter peer state machine expose equivalent behavior. The behavioral equivalence is proved using the concept of weak bisimilarity.

Proposition: The labeled transition systems T_{AppCSH} and $T_{Diameter}$ are weakly bisimilar.

Proof: To prove the bisimulation relation between two labeled transition systems, it has to be proved that there is a bisimulation relation between their states. With U it is denoted a relation between the states of T_{AppCSH} and $T_{Diameter}$ where U = {(Null, Closed), (SessionCreated, Open)}.

Table 1 presents the bisimulation relation between the states of T_{AppCSH} and $T_{Diameter}$. The mapping between the OSA Charging interface methods and Diameter messages defined in Section III shows the action's similarity. Based on the bisimulation relation between the states of T_{AppCSH} and $T_{Diameter}$ it can be stated that both systems expose equivalent behavior.

V. An Example of Parlay X Payment Application

Let us consider an example application that uses the Parlay X Payment Web services. The application grants a bonus of a free minute to users talking more than 10 minutes. We assume that it is a post-paid model of charging.

The sequence diagram in Fig.10 shows that the application reserves an amount of the end user account where the reservation is specified as a volume (10 minutes) (steps 1-13). The application adds a volume to the existing reservation as the call lasts more than 10 minutes (steps 14-19), and a bonus is granted (steps 20-25). Subsequent charging against the existing reservation takes place at the end of the call (steps 26-31). If the reservation times out, the remaining volume is returned explicitly to the account (steps 32-38).

VI. Conclusion

The paper investigates the charging control in Parlay X Web Services charging model and the charging models in IP-based manageable multimedia networks. The transformation of application facing Parlay X interfaces to the Parlay/OSA APIs is done at the Parlay X gateway which communicates with the OSA gateway. The OSA gateway makes translation between OSA interfaces and the control protocol in the underlying network. To provide access to charging functions to application logic, the Parlay X gateway needs to expose Parlay X Payment interfaces while the OSA gateway needs to "talk" Diameter protocol towards the network.

Using the available mapping of Parlay X Payment Web Services onto OSA Charging SCF we provide mapping the OSA Charging APIs onto Diameter protocol. We suggest a formal approach to specification of OSA gateway which has to maintain two state machines representing the OSA charging session and Diameter peer respectively. The approach is based on the mathematical formalism of labeled transition systems and the notion of bisimilarity.

Because of OSA gateway complexity, model-based testing techniques assist in its systematization. By starting from a formal model, test cases can be derived automatically in order to prove the conformance of implementations with respect to their specifications. Automation of some parts of the testing activity, using models and formal methods is a way to improve the quality and to reduce the cost of design.

References

[1] H. Hanrahan, *Network Convergence: Services, Applications, Transport, and Operations Support*, Wiley, 2007.

[2] H. Chua, C. Tan, "Service Broker Function in IMS Architecture - Issues and Considerations", in *Proc.12th WSEAS International Conference on Computers*, Heraklion, Greece, 2008, pp.179-186.

[3] T. Zahariadis, S. Voliotis, "Adaptive Middleware Platform for Next Generation Mobile Networks", in *Proc. WSEAS International Conference,* Cancun Mexico, 2004, Available: http://www.wseas.us /e-library/conferences/cancun2004/papers/485-255.pdf

Table 1. Bisimilarity between OSA Charging Session Handling and Diameter Peer State Machine

Transitions in T_{AppCSH}	Transitions in $T_{Diameter}$
Null createChargingSession SessionCreated	Closed Start Wait-Conn-Ack, Wait-Conn-Ack Rcv-Conn-Ack Wait-I-CEA, Wait-I-CEA Rcv-CEA Open
SessionCreated commonList SessionCreated	Open Send-Message Open, Open Rcv-Message Open
SessionCreated release Null	Open Stop Closing, Closing Rcv-DPA Closed
SessionCreated sessionEnded Null	Open Peer-Disc Closed
SessionCreated reserveAmountReq AmountReserved, AmountReserved amountList AmountReserved, AmountReserved commonList AmountReserved, AmountReserved release Null, AmountReserved sessionEnded Null, AmountReserved reservationClosed ReservationEnded, ReservationEnded commonList ReservationEnded, ReservationEnded release Null, ReservationEnded sessionEnded Null	Open Send-Message Open, Open Rcv-Message Open, Open Send-Message Open, Open Rcv-Message Open, Open Send-Message Open, Open Rcv-Message Open, Open Stop Closing, Closing Rcv-DPA Closed, Open Peer-Disc Closed, Open Send-Message Open, Open Rcv-Message Open, Open Send-Message Open, Open Rcv-Message Open, Open Stop Closing, Closing Rcv-DPA Closed, Open Peer-Disc Closed
SessionCreated reserveUnitReq VolumeReserved, VolumeReserved volumeList VolumeReserved, VolumeReserved commonList VolumeReserved, VolumeReserved sessionEnded Null, VolumeReserved release Null, VolumeReserved reservationClosed ReservationEnded, ReservationEnded commonList ReservationEnded, ReservationEnded release Null, ReservationEnded sessionEnded Null	Open Send-Message Open, Open Rcv-Message Open, Open Send-Message Open, Open Rcv-Message Open, Open Send-Message Open, Open Rcv-Message Open, Open Peer-Disc Closed, Open Stop Closing, Closing Rcv-DPA Closed, Open Send-Message Open, Open Rcv-Message Open, Open Send-Message Open, Open Rcv-Message Open Open Stop Closing, Closing Rcv-DPA Closed Open Peer-Disc Closed

[4] P. Garcia-Diaz, R. Sanchez-Montero, S. Lafuente-Arroyo, S. Jimenez-Fernandez, “The role of Intelligent Network Technology in the Future Telephone Network Evolution”, in Proc. *TELE-INFO'05 Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics*, http://www.wseas.us/e-library/conferences/2005 prague/papers/493-252.pdf

[5] R. Chen, E. Su, V. Shen, Y. Wang, “Introduction to IP Multimedia Subsystem (IMS), Part 1: SOA Parlay X Web services, The Next Generation Network architecture for Telecom industry”, Available: http://www.ibm.com/developerworks/webservices/library/ws-soa-ipmultisub1/, 2006.

[6] I. Gronbak, *NGN, IMS and Service Control – collected information*, R&I Research Note, Telenor, 2006.

[7] M. Poikselka, G. Mayer, H. Khartabil, A. Niemi, *The IMS Multimedia Concepts and Services,* Third Edition, Nokia, Finland, Wiley, 2009.

[8] 3GPP TS 32.260 *Telecommunication management; Charging management; IP Multimedia Subsystem (IMS) charging*, v10.0.0, 2010.

[9] 3GPP TS 29.199-6 *Open Service Access (OSA); Parlay X Web Services; Part 6: Payment*, v9.0.0, 2009.

[10] E. Pencheva, I. Atanasov, “Quality of Service Based Charging Using Parlay X”, in *Proc. IEEE International Conference on Telecommunications in Modern Satellite Cable and Broadcasting Services TELKSIS'2009*, Nish, Serbia, 2009, vol. 2, pp.611-614.

[11] 3GPP TS 29.198-12 *Open Service Access (OSA); Application Programming Interface (API); Part 12: Charging Service Capability Feature (SCF)*, v9.0.0, 2009.

[12] ETSI TR 102 397-6 *Open Service Access (OSA); Mapping of Parlay X Web Services to Parlay/OSA APIs; Part6: Payment Mapping,* v1.1.1, 2005.

[13] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko, *RFC 3588 Diameter Base Protocol*, 2003.

[14] 3GPP TS 32.299 *Telecommunication management; Charging management; Diameter charging applications*, Release 9, v9.4.0, 2010.

[15] S. Bruda, C. Dai, “Timed Test Generation Based on Timed Temporal Logic”, *Recent Advances in Automation & Information*, pp. 15-20, Available: http://www.wseas.us/e-library/conferences/2010/Iasi/ICAI/ICAI-01.pdf.

[16] H. Guo, Y. Shin, W. Lee, “Enhanced Compositional Safety Analysis for Distributed Embedded Systems using LTS Equivalence”, in *Proc.*

6th WSEAS International Conference on Applied Computer Science, Hangzhou, China, 2007, pp 115-120.

[17] J. Sebestyénová, "Hierarchical Verification of Reactive Systems with Timing Constraints", *WSEAS Transactions on Computers*, issue 4, vol. 2, 2003, pp. 1174-1179.

[18] X. Chena, R. Nicola, "Algebraic characterizations of trace and decorated trace equivalences over tree-like structures", *Theoretical Computer Science*, pp. 337–361, 2001.

[19] P. Panangaden, "Notes on Labelled Transition Systems and Bisimulation", [Online] Available: http://www.cs.mcgill.ca/~prakash/Courses/comp330/Notes/lts09.pdf.

Ivaylo Atanasov received his M.S. degree in Electronics from Technical University of Sofia, Sofia, Bulgaria, and PhD degree in communication networks. His current position is Associate Professor at Faculty of Telecommunications, Technical University of Sofia. His main research focus is set on open service platforms for next generation networks.

Evelina Pencheva received her M.S. degree in Mathematics from University of Sofia "St. Kliment Ohridski, Sofia, Bulgaria and PhD degree in communication networks from Technical University of Sofia. Her current position is Professor at Faculty of Telecommunications, Technical University of Sofia. Her scientific interests include next generation mobile applications and middleware platforms.

Dora Marinska received his M.S. degree in Telecommunications from Technical University of Sofia, Sofia, Bulgaria. She is a PhD student and her thesis is in the field of efficient telecommunication resource management.

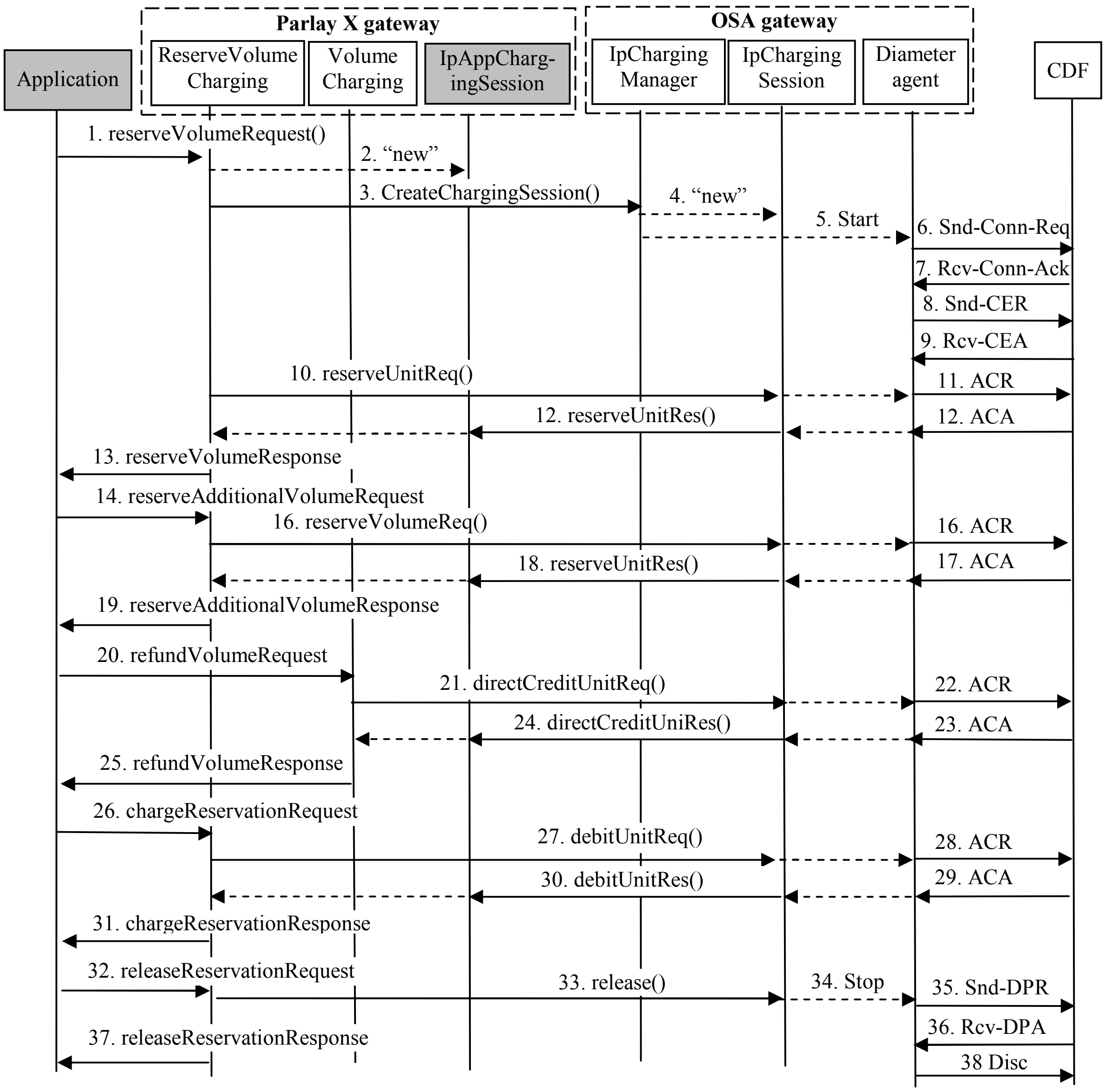

Fig.10 An example of Parlay X payment application