FORMAL SPECIFICATION AND VERIFICATION OF THE ACCOUNTING MODEL OF TINA ARCHITECTURE USING LOTOS AND ALDÈBARAN

Leila Lisiane Rossi
leila@lrg.ufsc.br

Ivanise Volpato de Souza
ivanise@lrg.ufsc.br

Abderrahim Sekkaki
sekkaki@lrg.ufsc.br

Carlos B. Westphall
Westphal@lrg.ufsc.br

Network and Management Laboratory
Postgraduate Program in Computer Science
Technology Center
Federal University of Santa Catarina

Tel: +55.48.3319739 Fax: +55.48.3319770

ABSTRACT

This paper presents the formal description and verification of the Accoounting Model of TINA architecture, perfected on basis of the described initial model in documents TINA-C - Telecommunications Information Networking Architecture Consortium. Is was used a technique of Formal Description LOTOS to make the specification and Aldèbaran tool for verification, and guaranteeing subsidies for future implementations.

Key words: TINA-C, TINA Architecture, Accounting Model, LOTOS.

1. Introduction

Because of the growth of computer networks accured an increase in the management necessity as a form to guarantee security and control of the availables resources. TINA-C has as

main objective that is to provide an architecture that combines the best of telecommunications and information technology.

In the Networks and Management Laboratory - LRG of the Federal University of Santa Catarina - UFSC, are developed some researches in the networks management area, with the objective to define new solutions for the management functions.

Many researches was developed on the accounting management function, showing the contributions given by the LRG in the national and international context. In function of these original contributions in the functional area of the accounting management, in this work, we continue to perfect the research previously developed [NoCr 99].

This article presents the Accounting Model of TINA architecture, its formal specification with the aid of the Technique of Formal Description LOTOS and the verification through the Aldèbaran tool. It is organized in the following way: section two there is an overview of TINA architecture and its phases of development; section three presents a overview of the accounting management of TINA Architecture; section four shows the informal description of the Accounting model of TINA architecture; section five presentes the Technique of Formal Description LOTOS; section six describes the function of the Aldèbaran tool; section seven shows the formal specification of the Accounting model of TINA architecture using the Technique of Formal Description LOTOS; section eight describes steps made for the verification of the model with the aid of the Aldèbaran tool; section nine presents the conclusion of the paper, and perspectives for future works. Finally, section ten presents bibliographical references.

2. TINA Architecture Overview

TINA architecture (Telecommunications Information Networking Architecture) was created through of consortium made up of over 40 companies, such as Telecom vendors, Telecom operators and software vendors [Kris 97]. First phase of its development (1993 – 1997), is aimed at defining a global architecture for telecommunication systems advanced software technology. Second phase (1998-2000) includes working groups and special interest groups, defining the specifications and initiating activities of standardization as a TINA Forum that coordinates these activities. TINA architecture is composed basically by TINA business model and Service architecture.

2.1. TINA Business Model

TINA business model defines one framework to specify reference points (they support platforms DPE (distributed processing enviroment) in all used points for TINA) and propagate petitions (requirements) on the system TINA.

It supplies mechanisms to specify, to add and to modify reference points and roles in the TINA system [Muld 97]. TINA business model specifies one framework of common business that defines a set of conditions on which can be made:

- creation of new business roles and reference points;
- distinction of business roles existing and reference points to deal with changes TINA roles, using reference points which are already defined.
- An initial set of business roles and relationships of these business to apply the TINA methodology.
- Petitions (requirements) imposed by TINA system to give covering to a particular set of services.

2.2. Service Architecture

TINA service architecture is a set of concepts and principles to construct, to develop and to operate TINA services [Abar 97].

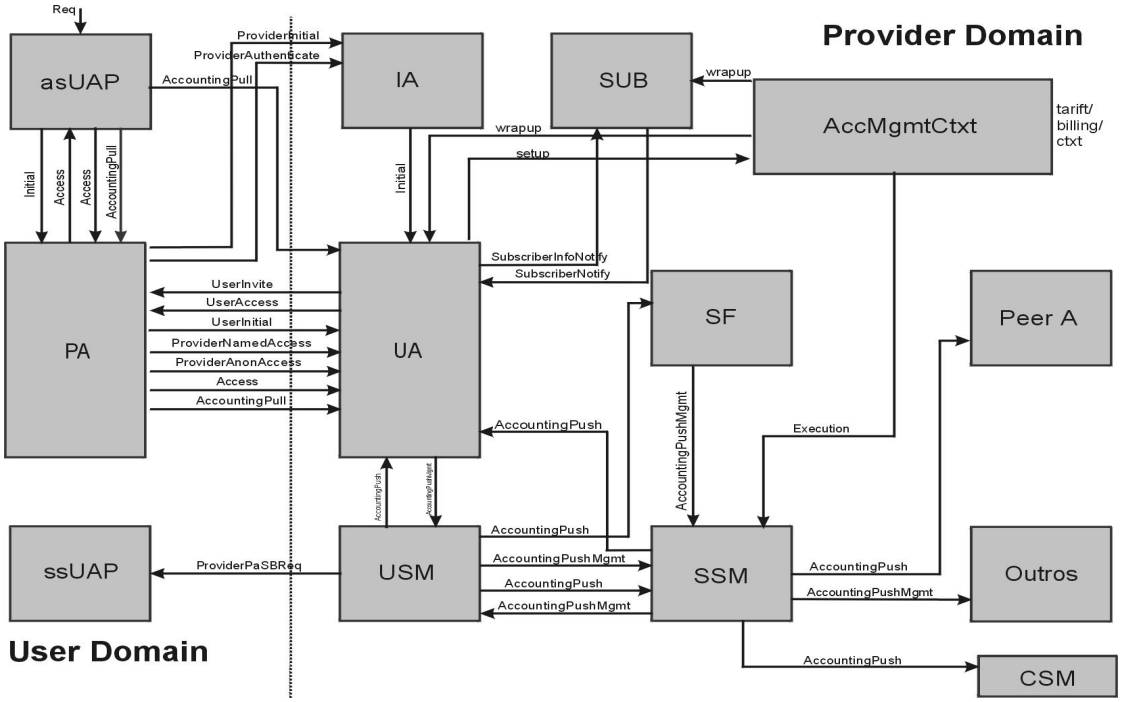

Figure 2. Consumer – Retailer Functioning Model of the TINA Architecture

It identifies components to construct services and to describe as they combines and interact themselves [Kris 97]. This architecture is divided on access session and use session. The access part encloses the requested interactions for two parts to establish the use parts. The use part is divided on service (interactions between components are requested to control behaviors of the services) and communication (interactions are requested to establish and to keep connections between components that implement the service).

3. Accounting Managment Overview

Accounting management consists of four cycles, namely metering, classifying, charging, and tariffing [Hama 96].

- *Metering* – record the use of resources.
- *Classification* – refers to classify metering information into a set of classes based on usage of service, resources being used, etc.
- *Tariffing* – refers to a step to calculate charging information through the classification obtained in the previous cycle and the tariff structure.

- *Billing* – refers to a process of charging information being stored and then the information is sent to a consumer.

4. Informal description of the Accounting Model of TINA Architecture

The role of the Consumer in the TINA architecture is to use services supplied by Retailer [Abar 97].

The Retailer business roles serves stakeholders in the consumer business roles providing them with access to service. A retailer may use other provider to support the provisions of services to consumers.

asUAP - Access Session User Aplication, can be used by human users and/or other applications in the user domain.

The PA - Provider Agent is an independent service of a component service defined by an end point user of an access session.

It acts as function of user access and allows to make use of services through of a access session.

The IA - Initial Agent has the capacity of authenticate the requeriments in the provider domain and establishes the access to a session. It is the end point access of the user with the Provider domain through the PA.

The UA - User Agent is an independent service that represents the user in the Provider domain. It makes the control and management of the services that the user needs to use, allowing to be an identified or anonymous user.

The USM - User Service Session Manager manages the user session (finishes or suspends a session). It allows to accounting services that the UA is using.

The SSM - Service Session Manager supplies or manages resources for the USM and UA. It supports service capabilities that are shared among members (parties, resources, etc.) in a service session.

Peer A - It supports contracts between the user domains. It is a mirror of the user domain.

SF - Service Factory allows to create and to manage objects for a service session. It eliminates, suspends and reduces the capacity of these objects, and returns to the customer the references from the interfaces.

ssUAP - Service Session User Aplication participates in the service session, supports the User Service Session in the User Domain.

AccMgmtCtxt - Accounting Management Context guarantees that the accounting is preserved through a set of activities of distributed objects which constitute the service.

Sub - Subscription Management Component allows the management of subscribers and users for the set of services supplied for a provider.

CSM - Communication Session Manager allows the communication between two parts It is responsible for end-to-end service (application) level connections.

Interfaces used for the communication between Consumer - Retailer are the following :

Req, the user does a requisition.

AccessUA and *accessPA* allow the PA to inform the consumer of events associates with its sessions of access, new sessions, etc.

Initial allows the consumer to get in contact with a retailer. The PA offers this operation for a asUAP, to get in contact with a nominated provider and to allow the consumer to establish an access session.

AccountingPull allows the UAP saves accounting datas of the PA. ProviderInitial the PA asks for the IA to give name of the provider.

ProviderAuthenticate allows information on authentication.

UserInvite invites the user for a session.

User Access, retailer finds the output on the interfaces in the consumer domain.

User Terminal, retailer uses to access information of the terminal configurations.

UserAccessSessionInfo, retailer uses it to inform the consumer of changes states for service sessions which this consumer has with its retailer.

UserInitial allows retailer to begin a access session with the consumer.

ProviderNameedAccess allows a known user to access its subscripts services. This interface is returned when an user has been authenticated for a provider and an access session has been established.

ProviderAnonAccess allows an unknown user to access its services.

DiscoverServicesIterator allows the services recuperation.

AccountingPush allows the customers to store accounting datas in the namedUA.

AccountingPushManagement allows the management of accounting events. Reports of accounting are sent by an interface of accounting for the SSM as a consequence of session activities.

Execution allows transaction of services, setup, execution and ending for the accounting.

Setup the user presents his/her accounting schema, and the service provider also presents its accounting schema.

Wrapup the service is concluded, and the metering information on possible distributed locations are collected and summarized.

5. Technique of Formal Description LOTOS

LOTOS (Language of Temporal Ordering Specification), is a technique of formal specification that is structuralized in algebric methods for the representation of processes [BoLa 95]. This tool allows to define applications in the area of management networks that use oriented objects. Technique of Formal Description LOTOS is formed by two parts: Dynamic part: it indicates the order in which events occur. Static part: it deals with the representations and expressions of values and data structure of the LOTOS specifications.

7. Formal Specification LOTOS of the Model

The specification in the highest abstraction level of TINA service architecture corresponds to a formalization of the users requirements [Nori 97]. This specification is used as a base for the posterior refinements of the conception, during elapsing of the design, and it is used in the correction test of the formal specification of the system.

Such specification demonstrates the behavior between the domain of the user and the provider through the Ret Reference Point which is divided in access and use [Muld 97]. The access part was totally specified while in the use part only the accounting service was specified [Fawe 97]. The corresponding specification LOTOS can be observed as following:

```
specification 1s[req,initial1,providerinitial,initial2,userinvite,providerauthen
ticate,initial3,userinitial,providernamedaccess,setup,useraccess,access1,accountingpull1,accountingpull2,access2,accountingpull3
,access3,subscriberinfonotify,subscribernotify,providerpasbreq,accountingpushmanagement1,accountingpushmanagement2,acco
untingpush1,accountingpush2,accountingpushmanagement3,accountingpush3,accountingpush4,accountingpushmanagement4,acc
ountingpush5,accountingpushmanagement5,accountingpush6,execution,wrapup1,wrapup2]:noexit

        behaviour

        1s[. . .]

        where

        process  1s[. . .]:noexit:=
```

```
        req;(i;initial1;providerinitial;initial2;userinvite;providerauthenticate;initial3;userinitial;providernamedaccess;setup;user
access;access1;accountingpull1;accountingpull2;access2;accountingpull3;access3;1s[. . .]
            []i;account[. . .]
            )
        where
          process account[. . .]:noexit:=

          req;(i;subscriberinfonotify;subscribernotify;providerpasbreq;accountingpushmanagement1;accountingpushmanagement
2;accountingpush1;accountingpush2;accountingpushmanagement3;accountingpush3;accountingpush4;accountingpushmanageme
nt4;accountingpush5;accountingpushmanagement5;accountingpush6;execution;wrapup1;wrapup2;account[. . .]
        []i; 1s[. . .]
            )
          endproc
          endproc
endspec
```

In the specification of protocols the processes *access* and *account* are detailed. In the process *access* the authentication of the user is performed allowing it to request services. In the sequence, it qualifies the process *account* in which is made the control and the accounting of the used service. Observes the specification as following:

```
specification 1p [req,initial1, providerinitial, initial2, userinvite, providerauthenticate, initial3, userinitial, providernamedaccess,
setup, useraccess, access1, accountingpull1, accountingpull2, access2, accountingpull3, access3, subscriberinfonotify,
subscribernotify, providerpasbreq, accountingpushmanagement1, accountingpushmanagement2, accountingpush1,
accountingpush2, accountingpushmanagement3, accountingpush3, accountingpush4, accountingpushmanagement4,
accountingpush5, accountingpushmanagement5, accountingpush6, execution, wrapup1, wrapup2] : noexit

behaviour

1p[. . .]

          where

          process 1p[. . .]:noexit:=
             acesso [. . .]
             >>
             account [. . .]
          endproc

          process acesso [. . .]:exit:=
```

```
                req;(i;initial1; providerinitial; initial2; userinvite; providerauthenticate; initial3; userinitial;
providernamedaccess; setup; useraccess; access1;accountingpull1; accountingpull2; access2; accountingpull3; access3;acesso [. . .]
            []i;exit
                )
        endproc

        process account[. . .]:noexit:=

        req;(i;subscriberinfonotify;subscribernotify;providerpasbreq;accountingpushmanagement1;accountingpushmanagement2;accountingpush1;accountingpush2;accountingpushmanagement3;accountingpush3;accountingpush4;accountingpushmanagement4;accountingpush5;accountingpushmanagement5;accountingpush6;execution;wrapup1;wrapup2;account[. . .]
        []i;1p[. . .]
        )

        endproc
endspec
```

8. Verification of the Model

Verification is the formal test of specification satisfies properties desirable using rigorous mathematical methods. It can be of two types [NoCr 99]:

- Verification for Reduction: graphs generated by compilers are reduced according to equivalence relations that are strong and weak equivalence.
- Verification for Comparison: a specification is compared with another, under discretions as strong and weak equivalence. Strong equivalence requires each event in a system of transistions corresponds to an equal event in the other system of transistions. In the weak equivalence, an event in a system of transistions does not need to be related to an event in the other system of transistions.

To validate the formal specification was used the Eucalyptus tool that integrates CADP [Cadp 99] and Aldèbaran and the method used to compare was the verification for comparison. The correct results about syntactic and semantics analysis and deadlocks of the specification were obtained. The verification for comparison is showed below:

```
--------------------------------------------------------------------------------
----------------------
aldebaran    -bddsize 4   -oequ -std 1s.bcg ./1p.bcg | tee aldebaran.seq
TRUE
--------------------------------------------------------------------------------
```

Figure 4. Observational Equivalence between service and protocol specifications

9. Conclusion

This paper presented an overview of TINA architecture and detailed the Accounting model perfected on basis in information TINA Consortium, besides showing the formal specification and validation of the model through technique of formal description LOTOS and Aldèbaran tool, becoming this model, "basic model" for the specification and to the implementation of the others management funtions in future works.

10. Bibliographical References

[Abar 97] Abarca, C., et al., ***TINA Service Component Specification***, v. 1.0b, TINA-C, http://www.tinac.com, 1997.

[BoLa 95] Bolognesi, T.; Lagemaat, J. van de; Vissers, C. ***LOTOSphere: Software Development with LOTOS.*** Kluwer Academic Publishers, 1995. ISBN 0-7923-9527-8.

[Cadp 99] CADP ***Caesar/Aldebaran Development Package***. [on line] Documento disponível na internet via WWW:<URL:http://www.inrialpes.fr/vasy/cadp.html>, oct 1999.

[Hama 96] Hamada, T. ***Accounting Management Architecture***, TINA-C, http://www.tinac.com, 1996.

[Kris 97] Kristiansen, L., et al. ***TINA Service Architecture 5.0***, TINA-C http://www.tinac.com, 1997.

[Muld 97] Mulder, H., et al. ***TINA Business Model and Reference Points***, TINA-C, http://www.tinac.com, 1997.

[NoCr 99] Notare, M.S.M.A.; Cruz, F.A.; Riso, B.G.; Westphall, C.B. ***Wireless communications: security management for cloned cellular phones.*** In: IEEE WCNC'99 – IEEE Wireless Communications and Networking Conference. Proceedings...New Orleans, LA, USA, set 1999, p.1412-1416.

[NoCr 99] Notare, M.S.M.A.; Cruz, F.A.; Riso, B.G.; Westphall, C.B. ***Security management against cloned cellular telephones.*** In: IEEE ICON'99 – The IEEE Conference on Networks. Proceedings... Brisbane, Queensland, Australia, set 1999, p.356-363.

[NoRi 97] Notare, M.S.M.A.; Riso, B.G.; Lorena, P.S.; Neto, M.C. de O.P.; Westphall, C.B. ***Formal Design of a Platform for Telecommunication Heterogeneous Network Management.*** DSOM'97 – Distributed Systems Operations & Management, Sydney, Australia, Oct 1997, p.263-278.