Accounting Management based on TINA Service and CORBA Architecture

Luis Marco Cáceres Alvarez[1] Wagner Tatsuya Watanabe[1] Carlos Becker Westphall[1] Abderrahim Sekkaki[1,2]

[1]Network and Management Laboratory
Technological Center
Federal University of Santa Catarina
PhoneCaixa Postal 476
88040-970, Florianópolis - SC - Brazil
Tel: +55.48.3319739 Fax: +55.48.3319770
caceres, wagner, ekkaki, westphal@lrg.ufsc.br

[2]Department of Mathematics & Computer science
University Hassan II, Faculty of sciences- Ain chok
P.O Box 5366, Maarif – Casablanca. Morocco
Tel : +212 2 2230680, Fax +212 2 2230674
[1]sekkaki@facsc-achok.ac.ma

Abstract
TINA (Telecommunications Information Networking Architecture) concepts and principles are introduced with the objective to remedy problems of centralized service control and service data model existing in IN (Intelligent Network). Now, it is becoming clear that future sophisticated services breaking away from the simple telephony call model, e.g., multimedia, multi-party conferencing, etc., will need to be rapidly and efficiently introduced, deployed, shared, operated and managed. In this context, TINA developed a comprehensive architecture for multi-service networks that shall enable multimedia communications and give access to information for business and private consumers. On the other hand, the provisioning of all management context instrumentation for TINA services (i.e. FCAPS service management functions) is still an open research question.
In this paper, we discuss accounting features and requirements, security of accounting management, and issues for their integration in a TINA-based service environment. A prototype has been implemented to validate the concept and the results are also presented.
Keywords: TINA, accounting management, security of accounting, CORBA

1. Introduction

TINA might be considered an architecture that is relevant to the existing telecommunications services infrastructure (i.e. the IN). This is mainly due to one of the TINA assumptions, which is based on the presence of the DPE (Distributed Processing Environment), and CORBA (Common Object Request Broker Architecture) as signaling infrastructure. Services in TINA are realized as distributed applications and provided within a distributed computing environment.

If today TINA Consortium coordinates works to refine and extend the specifications to ensure internetworking with legacy systems, and explain how TINA applies to IN and answers the IN evolution challenges, but the specifications concerning the management domain remain incomplete.

TINA has defined the Network Resource Architecture (NRA) [1] that covers the management areas of connection, configuration, fault and accounting management. Some of these areas have been defined in a great level of detail (e.g. connection management[2]) while some others areas have only started being designed (e.g. fault management and accounting management architecture[3]) and some are not yet defined at all (e.g. performance management). The security management is also not yet fully addressed in TINA. Hence, to begin with, we have been working for the framework of the TINA accounting management architecture, as one of the service management areas known as FCAPS that is an indispensable part of all the TINA services. We also discuss the relevance of the security of accounting management and how to protect the accounting information.

This paper is organized in the following way: Section 2 presents a TINA overview: service architecture, service management and TINA accounting architecture. Section 3 presents the accounting management system, the main components of TINA accounting management and their features. Section 4 discusses security services, mechanisms and presents security considerations in accounting. Section 5 describes the specifications of the development of the prototype in TINA-based service environment. Finally, a conclusion is given in Section 6.

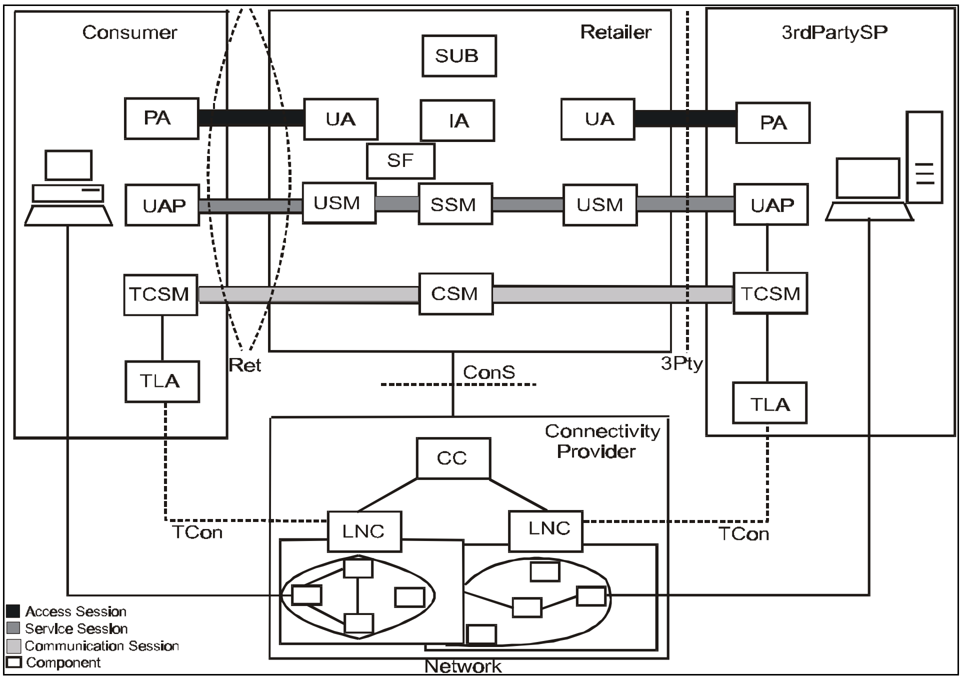
2. TINA overview

2.1. TINA Service architecture

TINA defines a flexible architecture which consists of several modeling concepts and specifications. As well as the service architecture[4], the main specification involves the business model [5], the computational model, and the service component specification [6]. Figure 1 shows a mixture of the business model and the computational model based on TINA specification though many details are omitted.

In the business model, the participants (or stakeholders) in a service are categorized into five groups according to the roles they play in the service. The roles defined within TINA are Consumer, Retailer, 3rd Party Service Provider, Connectivity Provider and Broker (not shown in Figure 1). The Broker and 3rd Party Service Provider roles have not been described in detail in TINA. The Consumer, Retailer, and Connectivity Provider roles, however, have been specified in TINA with sufficient detail, including the interfaces between them. The interface among the participants are defined as References Points. The most important reference point for service management is the Ret-Reference Point [7] defined between the Consumer and the Retailer.

The service architecture also defines two types of sessions namely an access session and a service session. The access session is responsible for the identification and authentication of the user. It is also used to search available services and executes the services by creating a service session. The service session is responsible for managing the condition of the service. The service session also creates and controls stream bindings, which is an abstract representation of end-to-end connection among applications. The functionality of the service session is defined by a number of functional groups called Feature Sets. The Feature Sets contain various functions, such as: joining a service, inviting parties in the service session, establishing stream connectivity requirements, issuing voting, etc. These functions are grouped into the Feature Sets, such as ‘basic’, ‘multiparty’, ‘stream-binding’, ‘voting’, etc. Subscription subsystem and subscription information model defined in TINA allow an user to discover the new services and eventually subscribes to them. This is independent of the access technology used because the end-users require more and more flexibility, and demand the ability to activate and deactivate service subscription themselves, and alter the related profiles[8].



CC: Connection Coordinator	**SF: Service Factory**	**UA : User Agent**
CSM: Communication Session Manager	**SSM: Service Session Manager**	**UAP: User Application**
IA: Initial Agent	**SUB: Subscription**	**USM: Usage Session Manager**
LNC: Layer Network Coordinator	**TCSM: Terminal Communication Session Manager**	
PA: Provider Agent	**TLA: Terminal Layer Adapter**	

Figure 1.- TINA Service Architecture overview.

2.2. TINA Service management

Service management is a very broad discipline and it is also an important part of TINA service architecture. It involves (includes) management functions in TINA service architecture. It controls the end-to-end transactions for TINA services to provide FCAPS properties dictated by binding management contexts. There are four conceptual axes associated with service management:

- Partitioning axis: TINA is partitioned into three layers (service, resource and DPE).
- Functional axis: is represented most notably by FCAPS functions. To support the FCAPS integrity of a service session, constructs such as management context and service transaction are provided.
- Computational axis: represents computational support for management needs.
- Life cycle axis: represents the life cycle issues, including service life cycle management and user (consumer) life cycle management.

To develop the management functions on a DPE (Distributed Processing Environment) platform, referring to the existing standards such IUT-T/TMN and ISO/ODP (Open Distributed Processing), TINA uses a common object interface description language (IDL or ODL). It uses also the Network Resource Information Model (NRIM), which enables us to describe systematically the network management information using unified abstract models of network resources [9].

2.3. TINA Accounting architecture

In addition to the existing standards such as X'742 [10] and of the different works achieved on enriching the OSIMIS (OSI Management Information Service) platform with a structure supporting the developments of accounting applications [11], TINA dedicated one document to this topic[3]. Because of the concept TINA, a higher level concept is necessary, which maintains service management context across multi-domains (stakeholders), and separates service management from service as clearly as possible[12], and on-line charging is still a challenging problem.

TINA accounting management consists of four cycles namely Metering, Classification, Tariffing and Billing. It introduces a concept of Accounting Management Context (AcctMgmtCtxt) associated with Service Transaction (ST). The purpose of AcctMgmtCtxt is to guarantee that accountability be preserved through a set of activities of distributed objects, which constitute the service. We want to emphasize that accountability is not a property or an attribute of a single object. It is rather a set of quantities measured or calculated over a set of activities of distributed objects throughout the service. When a ST is activated by a Service Session Manager (SSM), its AcctMgmtCtxt is interpreted in accordance with its Session Component description (unit of service to be accounted) and with the tariff structure within the SSM component. Then, the SSM passes control and necessary parameters to resource or computational level mechanism such as Communication Session Manager (CSM), notification server, metering managers and service management.

A Session Component (SC) is not necessarily an object, it is a component which constitutes a session covered by a Service Transaction, and the SC itself is divided into various events. It is introduced in order to respond to dynamic and distributed accounting needs.

The Service Transaction (ST) concept can be seen as it consists of three phases:

- Set-up phase : it is a phase of negotiation between the user and the provider (e.g. Tariff structure , etc.). The agreed schema is submitted as AcctMgmtCtxt of the ST.

- Execution phase :The service is being offered to the user, as it was specified in the first phase. The description of information of the service session is specified as a part of AcctMgmtCtxt.
- Wrap-up phase : The service is concluded, account record or charging information may be sent to the user at the conclusion of the transaction, if it is specified in AcctMgmtCtxt.

3. TINA accounting management architecture - Components

In reference [11], an extension to the OSIMIS management platform through implementing the Usage Metering Function defined in the OSI functional model has been developed and validated. This function specifies generic managed objects supporting the construction of OSI applications involving accounting activities. Our present work is the continuity of activities undertaken of network management domain in our laboratory (the Network and Management Laboratory of Federal University of Santa Catarina).

3.1. Architectural model

Figure 2 shows the architectural model of accounting management context with the interaction between the main components of TINA service architecture.

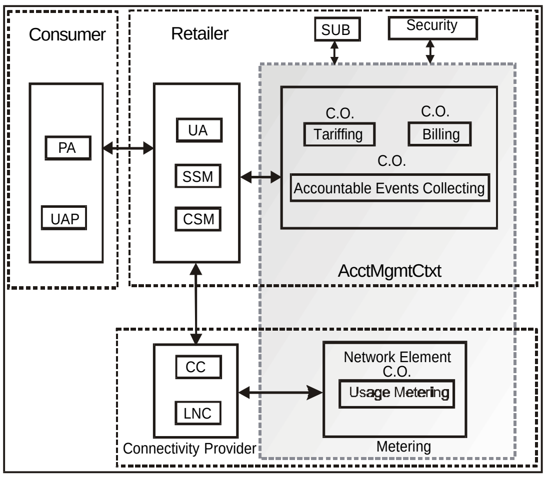


Figure 2.- Architectural model.

Components are interpreted here as the units of reuse. TINA AcctMgmtCtxt consists of separate components, covering different aspects of service, and network control and management.

Accountable Event Collecting

The Accountable Events Collecting component receives, collects and collates the accounting events associated with the Session Components status or Session Component state changes, and events associated with charging information generated by the SSM. It is divided into two modules: *EventManagememt* and *SessionComponent.*

Tariffing

Tariffing component in TINA AcctMgmtCtxt can be represented by two different modules:

- *Recovery* Module, which gives charging-rate as a function of the current Session State. It converts the collected accountable events to charging record and stores it into the charging record database. In addition, it allows to restore the information collected when failure in the service.
- *Tariff* Module, which gives accumulated current charges as a function of event sequence. It calculates the tariff, using the charging formula according to the contract of subscription and stores it into the billing record.

Billing

The billing component may be automatically issued at the end of a billing period as negotiated and defined in the customer's contract. It make out the bill based on charging information from billing records. There are four billing configuration :On-line charging, Shared billing, Third-party billing, and Credit-debit billing. The billing implementation in the application (prototype) is conFigured in the on-line charging.

Usage Metering

During the life time of the connection, the usage metering component collects and controls data acquisition concerned with the use of network resources generated by the LNC. It also registers the collected data for future processing (fault and performance management*). In our context, a metering is not necessary, since a clock is used to time stamp the start and stop times of service.*

3.2. Definition the AcctMgmtCtxt components

The model of implementation uses the CORBA objects with Visibroker 3.04 [13]. The objects are defined such as interfaces in IDL, so they can access and to be accessed by any other CORBA object independently of the programming language[14]. The definitions in IDL of the AccMgmtCtxt components are defined in Figure 3.

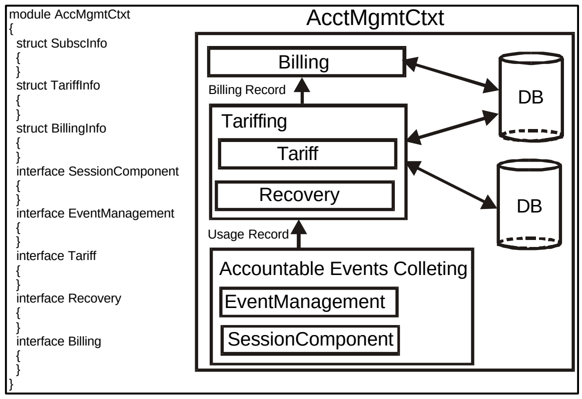


Figure 3.- Definition of AcctMgmtCtxt components.

The building of an IDL specification for implementation the prototype allows to creation of the interfaces that communicate between them. The data-bases defined in IDL are divided: "*SubscripInfo.mdb*" which defines the subscription user, the "*TariffInfo.mdb*" which registers the events of Tariffing component while the service execution occurs, and "*BillingInfo.mdb*" which defines the charging-billing of the user.

The module AcctMgmtCtxt is the most important component in this class, it is generated in the set-up phase of the ST and it is destroyed when the service transaction is concluded. The interface *SessionComponent* is used for controlling and managing the SC, as also start and stop metering actions accounting object. The SC refers one service to the specified user. *EventManagement* defines the events management parts (for example: delivery).

In the prototype implementation, one session will be used to test the service session: the "digital video-audio-conference" between the users and a provider. The components forming parts of the session are extended to generate accountable events towards the AcctMgmtCtxt components and allow to each consumer to retrieve on-line billing information about an ongoing shared service session. In the beginning of the execution, a graphical interface is visualized (see Figure 4), subdivided into the following modules:

- **Subscription:** login as a known user: authentication and verification of the user features; display data user;
- **Up-date Subscribed**: up-date of user (Insert, Modify, Delete, etc);
- **Monitoring service**. display the video conference service for each user; and
- **Error messages**: show the errors.

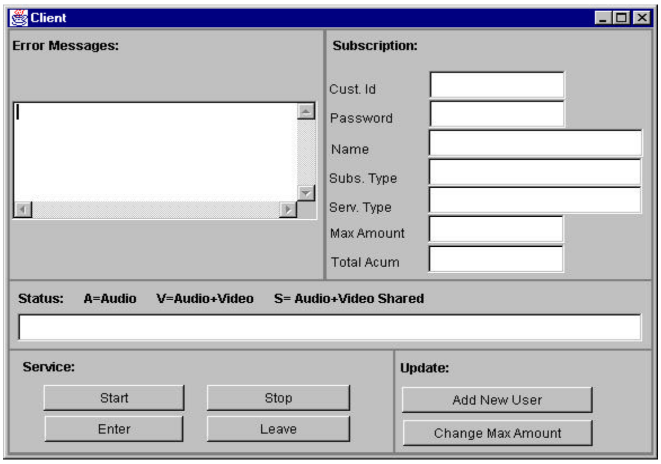


Figure 4.- Graphical Interface.

4. Security of TINA accounting management

In distributed systems, security is essential and a natural part of it. Of course accounting is not an exception. Security of accounting implies: guaranteed accountability, trustful accounting, integrity of accounting information, etc.

- Guaranteed accountability refers to the services transaction that should ideally offer a mechanism to guarantee integrity of the service. This means that " you do not pay if you do not get the agreed service"
- Trustful accounting means that accounting information should be assured. The requirement come from the openness of TINA. A user and a service provider which are totally unknown to each other can connect by using an on-line Yellow Page or by using a trader (situation where security mechanism is urgently needed). How can user trust in the service provider and conversely how can a service provider trust in the user in reverse? The first question is related more to accounting management whereas the second is related more to security management. It is similar to the case where an unknown phone company sends you an expensive bill based on questionable service usage. Openness is not necessarily a good thing unless both the user and the service provider are properly protected. Accounting information should be trustworthy and should be able to be recorded on both sides in a non-modifiable and non-refutable manner.
- Integrity of accounting information – infers that integrity of accounting information should be preserved over network failures and over disruption of services, considering the service through different management domains.

In our system, the security functions are performed by the SBS (Security Base Server)[15]. It provides an authentication service, a ticket distribution, an access control and a SMIB (Security Management Information Base). These services are implemented by CORBA objects. When a principal invokes a service on a server object, there is a request and a reply interacted between them. This provide security services, secure objects having security functionality using the Security Interceptor(SI) module to intercept the request and the reply. The mechanism is transparent to the principal. The most of the security functions are performed by the Security Base Server during the invocation of the target object. Figure 5 presents the general architecture of the model with components of the Security Base Server.

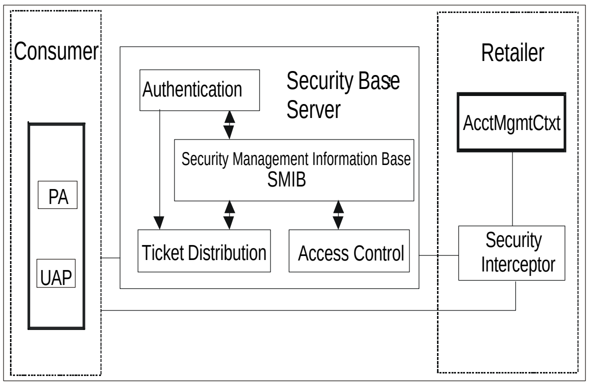


Figure 5.- Architecture of the Security Base Server.

Authentication service provides identification and authentication of the principal along with the authentic information which is provided by the user application and the Security Management Information Base SMIB. Authentication information is encrypted with a key that is generated by the client SBS and SI (Security Interceptor). The user is mapped in the SMIB. An encrypted ticket is generated and sent to the user.

Access control service performed by the SBS provides authorization and access control to decide whether a principal can access an object. The process is based on the client's capability (principal security attributes, target control attributes, services and other relevant information that principal can access).

In our model, the SBS acts as a Security Manager. It controls and manages the security context of the interactions between the consumer and the retailer. The Security Interceptor (SI) is a security agent factory that helps in the access control of the service in the retailer domain, including the AcctMgmtCtxt.

5. Description and visualization of specifications

5.1. Description mechanism

The prototype is divided into two phases: subscription phase and monitoring phase.

a) Subscription phase

- The prototype begins a service when it is solicited from user (module Service, click button "Start" in Figure 4) that registers his log, entering his Id and password (module Subscription in Figure 4).
- Immediately, a verification (access to database "SubscripInfo.mdb") and an authentication are carried out through the security management (SBS in Figure 5). The authorized user presents himself under two types:
  1. the user client which has a contract with provider and priority of full service uses: audio (speaking-listening) , and video (vision-image), i.e. A+V;
  2. the user, which uses the service for the first time, and does not have a contract with provider, assists only a partial service and he is seen as an Audio participant : audio (listening), video (vision).The new user will be registered in database (module Up-Date: "Add New User" in Figure 4) to assure that his amount of available credit is enough to

make use of the service or in any other way, giving a log-out, not permitting the use of the service.

Notice that the security mechanism is present during a session processing. All requests destined to the provider are transparently intercepted by the Security Interceptor (SI) that redirects it to SBS (see Figure 5), making possible the principal authorization by checking his identity and rights in SMIB. Data exchanged will be encrypted and it will guarantee security in DPE like TINA [15].

b) Monitoring phase

- For each user authorized, the proper AcctMgmtCtxt is created, starting service (module Service, click button “Enter” in Figure 4).
- Automatically, the AcctMgmtCtxt creates the objects (interface *SessionComponent* and interface *EventManagement*) that identifies the management events during the execution.

```
interface SessionComponent
{
 SubscInfo SubsValid(in SubscInfo information);
 void AddNewUser(in SubscInfo information);
}
```

```
interface EventManagement
{
 boolean RegService(in string cid, in long service_type);
 long GetServiceType(in string cid, in long service_type);
 long EnterEvent(in string cid, in long  service_type);
 long LeaveEvent(in string cid, in long  service_type);
}
```

At the same time, a Session State Vector is generated (SSVec) where each position (pos) registers the information relating the events of service:

(1): initial time(it);
(2): final time(ft);
(3): accumulated time “full service use”: A+V (fs) or “partial service use”: (ps); and
(4): accumulated time “shared service use” of the users (ts).

- Click on button "Enter", the "it" is initialized while activating "fs" or "ps". The vector is seen as a dynamic and volatile structure, any failure that occurs during the service is registered as information in database. The back-up is necessary to store the information (events) in the database ("*TariffInfo.mdb*") because its interaction with module Tariffing (interface *Tariff*) is periodic.

```
interface Tariff
{
 Tarval Tariff_val();
}
```

- During the period of the service, the provider must verifying that the time expended for the user does not pass 80% of the available limit. If it occurs, the provider must notify the user and consults him if he desires to modify his available quota (module Up-Date: "Change Max Amount" in Figure 4), or log-out of service (active the interface *Billing* for its charging).

```
Interface Billing
{
 void BillingSave(in string cid, in string name, in  string  date,
 in long totalsecs,  in  long totalunits, in  string  inittime,
 in  string endtime, in long asecs, in  long avsecs,
 in long  ssecs);
}
```

- The user that makes use of the service can leave the session (module Service, click button "Leave" in Figure 4). In this moment, the user is in the state of auditing (A). He has also the option to retake the service (module Service, click button "Enter" in Figure 4) or end of session (module Service, click button "Stop" in Figure 4).
- The option also exists for several users to share the service the same time. This happens when the button "Enter" is activated and then it actives "ts" and stop "fs". To leave the shared state, click button "Leave" to active "fs" or "Stop" to quit.
- If the user is in the state "Stop", the timers are desactivated, and "ft" is registered. Finally, the on-line charging (interface *Billing*) is activated for the supplied service of the user and the information is stored in database ("*BillingInfo.mdb*"). See Figure 6.

An important part of the service is when failure occurs. For example, events are lost during the session owing to the loss of timer synchronization, accounting SC state is interrupted for failed node, loss of energy, etc. At this moment, the interface *Recovery* stores and saves the information in database ("*TariffInfo.mdb* "). If it occurs (according to the user), restart the session and the timers to continue with the accounting of the events, or stop session (to quit a failure window, click " X ", and " Start " to continue in Figure 4).

```
interface Recovery
{
 boolean  CheckStatus (in string cid);
}
```

5.2. Scenario of execution

We consider in Figure 6, three users who share the same session of “multi-party conferencing service” provided by the retailer. Each user in the execution of the service has its proper graphical interface “Client”, that defines the Accounting Management Context, which allows the management of the available resources in the session to start, as described in item 5.1.

When the session is finished, the graphical interface “Billing” is displayed, that shows the on-line charging of the service supplied for the Retailer.

The users have three options of access to the service according to their contracts.

V (Video+Audio), A (Audio) and S (Video+Audio **S**hared). The use of the “V option” costs 3$/unit, the “A option” costs 1$/unit and the “S option” costs 2$/unit.

6. Conclusion

The CORBA facilities and services offer a set of interface that enables the construction of TINA service management. This paper provides an overview of the TINA concept specifying the accounting management context. We described accounting features, requirements, and some accounting issues. Then, we proposed the accounting management architecture specifying an architectural model which contains specific components such as AcctMgmtCtxt, Session Component, etc., covering different aspects of service control and management. We also integrated the security of TINA accounting management, security services and mechanisms. In order to realize our model, we described the accounting components in IDL , their roles and their interactions to accomplish the accounting functionality. We illustrated a prototype in TINA-based service environment considering an used on-line billing.

7. References

[1] F. Steegmans et al., “TINA Network Resource Architecture,”

TINA-C, http://www.tinac.com, 1997.

[2] G. Pavlou et al., “Issues in Realizing the TINA Network Resource Architecture,” in: Interoperable Communication Networks Journal,” Vol. 2, No. 1, March 1999.

[3] T. Hamada, “Accounting Management Architecture,” TINA-C, http://www.tinac.com, 1996.

[4] L. Kristiansen et al., “TINA Service Architecture 5.0,” TINA-C http://www.tinac.com, 1997.

[5] H. Mulder et al., “TINA Business Model and Reference Points,” TINA-C, http://www.tinac.com, 1997.

[6] P. Farley et al,. “TINA Retailer Reference Point Specification,” v. 1.0, TINA-C, http://www.tinac.com, 1998.

[7] C. Abarca et al., “TINA Service Component Specification,” v. 1.0b, TINA-C, http://www.tinac.com, 1998.

[8] M. Pampaey and A. Couturier, “Using TINA Concepts for IN Evolution,” in: IEEE Communication Magazine – June 2000.

[9] F. Steegmans et al., “TINA Network Resource Information Model Specification,” TINA-C, http://www.tinac.com, 1997.

[10] ITU-T Recommendation X.742, Information Technology – Open Systems Interconnection – Systems management: Accounting meter function- July 1994.

[11] L. F. Kormann, C. B. Westphall, and A. Coser, “OSI Usage Metering Function for OSIMIS Management Platform,” in : Journal of Network and Systems Management, Vol. 4, No. 3, 1996.

[12] P. Hellemans et al., “Accounting Management in a TINA-Based Service and Network Environment,” in: Sixth International Conference on Intelligence in Services and Networks , (IS&N’99), Barcelona, April 1999.

[13] Visibroker 3.04 http://www.inprise.com

[14] Object Management Group. The Common Object Request Broker Architecture 2.0/IIOP – specification Revision 2.0 – OMG Document – August 1996.

[15] A. Sekkaki, D. Nguessan, M. D. Müller, C. B. Westphall. “ Security within TINA Accounting architecture management,” to appear in IEEE International Conference on Communications, ICC2001, Helsinki – Finland, June 11-15 2001.

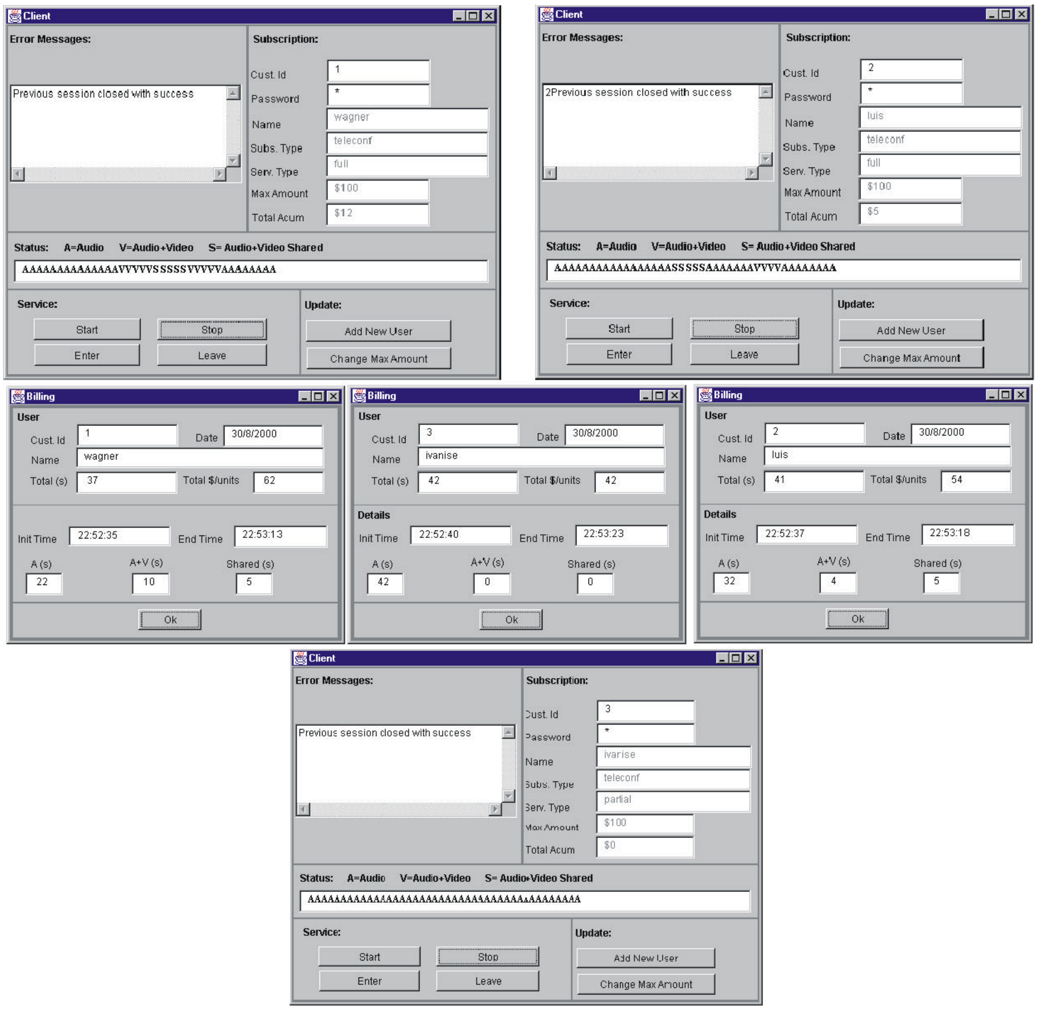


Figure 6.- Example of scenario