TECHNISCHE UNIVERSITEIT EINDHOVEN

Department of Mathematics and Computing Science

MASTER THESIS

Quality improvement through architecture analysis using the ATAM

by

J.J.M. Baijens

Supervisor: dr. M.R.V. Chaudron
Advisors: ir. H. Vesters, ir. J.L.M. Verhoeven

November 2002
Preface

This document describes the research I performed for my master's thesis at the Department of Mathematics and Computing Science of the Eindhoven University of Technology. My research was done in two steps, first at Ericsson in Rijen later at CMG in Eindhoven.

A special word of thanks goes to the people from whom I got a lot of help and support during my work. This would be Michel Chaudron from the Eindhoven University of Technology, Rene Peeren and Hans Vesters from Ericsson and Jan Verhoeven from CMG. Furthermore I would like to thank the people with whom I have enjoyed working together from both Ericsson and CMG.

Joris Baijens
Eindhoven, November 2002
Summary

This thesis describes two case studies in which the use of the architecture trade-off analysis method (ATAM) produces improvements on the software architectures. The goal of this study is to get insight into how architecture analysis can improve an architecture and how well a method like the ATAM works.

The cases have a different setting. The first case was at Ericsson in the telecom area. The architecture of intelligent networks and the information and business (I&B) product had to be described first, which showed already room for improvements. The analysis showed furthermore organizational and business aspects which the ATAM does not include.

In the second case at CMG the architecture of interactive television is evaluated. The different context showed other aspects of architecture analysis. In this case better quantification of quality attributes was needed. Modeling and prototyping showed useful to get those figures. The architecting process can benefit from the ATAM but many more aspects, which are not covered by the ATAM (e.g. costs, available competence) play a role and can influence the overall quality of a system.
# Contents

Preface i
Summary iii
List of Figures vii
List of Tables ix

1 Introduction 1

2 The goals of the project 3
  2.1 Goals ................................................. 3
  2.2 Specific aspects of the case studies ..................... 4
  2.3 ATAM research questions .......................... 4

3 Architecture analysis 5
  3.1 Architecture ......................................... 5
  3.2 Architecture analysis ............................ 5
  3.3 ATAM .................................................. 6
  3.4 Cases .................................................. 8

4 Case 1: Intelligent Networks 9
  4.1 The Ericsson Organization .......................... 9
  4.2 Intelligent Networks .................................. 10
    4.2.1 History ........................................... 10
    4.2.2 Architecture .................................... 10
    4.2.3 Example .......................................... 12
    4.2.4 Network Provisioning ............................ 13
  4.3 Architecture description ........................... 13
    4.3.1 Conceptual Architecture ........................ 13
    4.3.2 Deployment view ................................ 17
    4.3.3 Use cases ........................................ 17
    4.3.4 Physical View ................................... 20
  4.4 Architecture analysis ................................ 22
  4.5 Analysis results and recommendations for improvements 25
    4.5.1 Improvements on the architecture ............... 25
    4.5.2 Recommendations for the architecting process ... 27
# List of Figures

4.1 A plain telecom-network ........................................... 11  
4.2 A IN telecom-network ............................................... 11  
4.3 Example of an IN-call .............................................. 12  
4.4 Intelligent Network Components ................................. 14  
4.5 Overview of Intelligent Network functions ....................... 15  
4.7 User interaction with intelligent networks ....................... 17  
4.6 Product deployment view ........................................... 18  
4.8 Service agent layer interaction ................................... 19  
4.9 Management component interaction ............................... 20  
4.10 Physical view of an Intelligent Network ......................... 21  
4.11 Quality attribute utility tree ...................................... 24  
4.12 Measurements of time spent in the components ................ 26  
4.13 Thin clients in management component ........................... 27  
4.14 Design, Advice and Warranty ..................................... 29  

5.1 Architecture meta-model views ..................................... 32  
5.2 Interactive TV overview ............................................ 32  
5.3 States of a return-server ......................................... 33  
5.4 Layers of the application-server ................................ 34  
5.5 Interaction of the application-server ............................ 34  
5.6 Compilation infrastructure ........................................ 35  
5.7 Logical overview .................................................. 35  
5.8 Quality attribute utility tree ..................................... 38  
5.9 The data-flow in the current architecture ....................... 41  
5.10 The data-flow in the proposed alternative architecture ........ 41  

B.1 The data-flow in the current architecture ....................... 75  
B.2 The data-flow in the alternative architecture ................. 76
List of Tables

4.1 I&B feature scripts ........................................... 16
B.1 Time measurements of $T_2$ ................................... 76
B.2 Time measurements of $T_2'$ .................................. 76
C.1 Configured values used for better network performance .... 79
C.2 Iperf options used for measuring ............................ 80
Chapter 1

Introduction

"If builders built buildings the way programmers wrote programs, the first woodpecker would destroy civilization."

– Weinberg’s Law

Despite improvements in software engineering, the so called ‘software crisis’ of the 1960s and 1970s remains. Projects are still not finished within the given budget and deadlines, and not all requirements are fulfilled. The software engineering improvements cannot keep up with the ever growing complexity of the systems. The problems are strengthened by fast changing markets, which demand new and more features in less time.

In the last decade a new approach to software engineering has evolved, which tries to control the complexity from a higher level, the so called architecting process. Following the ‘spiral model’ instead of the classic waterfall or prototyping models, the architecting process tries to balance the requirements from all the stakeholders by identifying the risks that can endanger the quality of the system. The architecting process is fairly young, which can be seen in the way architectures are described. Almost everyone uses his own notation and views to describe an architecture. Although some techniques are more used than others (e.g. the 4+1 model from Philippe Kruchten [Kruch95] and UML [UML]) comparing architectures is difficult. Therefore different architecture analysis methods have been developed the last five years. Especially the Software Engineering Institute [SEI] is working on those methods. The most commonly used method is the ATAM, Architecture Trade-off Analysis Method [ATAM] which arose from the SAAM, Software Architecture Analysis Method.

This thesis looks at architecture analysis in general and evaluates the ATAM in particular. This is done through two case studies. One at Ericsson on the architecture of the Information and Business product and one at CMG on the architecture of interactive TV.

Outline

In Chapter 2 the goals of the project are explained. In Chapter 3 a overview of architecture and architecture analysis will be given. The Architecture Trade-
1. Introduction

off Analysis Method will also be explained. Chapter 4 describes the case study on Intelligent Networks at Ericsson. Chapter 5 describes the case study on interactive TV at CMG. In Chapter 6 the two cases are compared and evaluated. Finally Chapter 7 describes conclusions and options for future work.
Chapter 2

The goals of the project

"Once you have learned how to ask relevant and appropriate questions, you have learned how to learn and no one can keep you from learning whatever you want or need to know."
– Neil Postman and Charles Weingartner

In this chapter the goals of the project and the accompanying research questions are described.

2.1 Goals

Software design had always an emphasize on functional requirements. Nowadays non-functional requirements are getting more and more important. With the use of architecture, one tries to fulfill the non-functional requirements. In the literature many articles can be found about architecture, the architecture process and architecture evaluation. Most of the articles are recently published and try to build a theoretical framework for architecture evaluation. The practical side of architecting is not well described.

In this project an architecture evaluation approach is applied on real-life situations by doing two case studies on complex systems. The main question that will be answered is how this approach works in practice.

In both case studies we try to improve the overall quality of the system, this in terms of quality attributes like performance, scalability, maintainability. The architecture evaluation is done with the use of the Architecture Trade-off Analysis Method (ATAM). This is one of the many methods developed for analysis. Most of them focus on some quality attributes but because the ATAM is very generic we chose to use this method. We try to discover if the use of the ATAM has a clear advantage compared to just using a development method like the waterfall model [Royce70], which are commonly used and focus on the functional requirements.

This means the goals are twofolded:

- Improvement of the overall quality of the systems in both case studies.

- Evaluating the architectural approach, using the ATAM as architecture analysis method.
2.2 Specific aspects of the case studies

When using a method to analyze and compare architectures some aspects become important. All the architecture analysis methods measure the quality attributes in a relative way. There exists no absolute scale that gives a grade to an architecture. Many times only back of the envelope calculations are made to get an overview of single quality attributes. The points where an architecture influences more than one quality attributes and how they are influenced, are even harder to quantify. We try to find a way to get those so-called trade-off points better quantified.

Another aspect is that the architecting process is not isolated, it has important relations with business processes. Things like the market circumstances, the organizational structure and available competence are many times not mentioned but have huge influence on the success of an architecture approach. We try to find the conditions needed for the successful use of an architectural approach.

2.3 ATAM research questions

The goal of the ATAM-evaluation can be written as a group of research questions. The project tries to answer the following questions:

- How well is the ATAM applicable?
- What are the benefits of the ATAM?
- How does the theory work in practice?
- How large is the overhead of the ATAM?
- Is the ATAM suited for every system?
- Does the ATAM have a benefit above other methods?
- Can some metrics be given for quality attributes?
- How well are the business drivers handled with the ATAM?
- Which aspects are not covered with the ATAM?
Chapter 3

Architecture analysis

"Software architecture is a design in a clownsuit."

– WICSA 2001

3.1 Architecture

How can we make better software? That is the question software engineers keep asking. ‘Software architecture’ is a research area that tries to find answers to that question. Compared to other research areas software architecture is very young. In the early 1960s software developers started reasoning about the large-scale structures of their systems. The emergence of the software architecture area started late 1980s with a significant contribution from Mary Shaw’s 1989 paper: “Larger Scale Systems Require Higher Level Abstractions” [Shaw89]. The last decade architecture evolved with the help of the analogy from architecture in the building world. This led to the different representations of an architecture by different views similar to traditional building design. Also different definitions of software architecture have been discussed the last decade. Nowadays a fairly broadly accepted definition of architecture is the IEEE definition [IEEE1471], it states, architecture is:

“The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution”

3.2 Architecture analysis

This section gives an outline of the methods currently available for analysis of software architecture quality attributes. The recently developed methods from different academic groups often are refinements of the SAAM or the ATAM. They usually restrict themselves to a limited set of quality attributes. Below a few of the currently available methods are given [METHOD].

1. SAAM, Software Architecture Analysis Method,
3. Architecture analysis

2. ATAM, Architecture Trade-off Analysis Method,
3. CBAM, Cost Benefit Analysis Method,
4. ALMA, Architecture Level Modifiability Method,
5. FAAM, Family - Architecture Analysis Method
6. SBAR, Scenario-Based Architecture Re-engineering
7. SAEM, Software Architecture Evaluation Model

3.3 ATAM

The purpose of the ATAM is to assess the consequences of architectural decisions in light of quality attribute requirements. Because the ATAM focuses on quality attributes it is critical to have precise characterizations for each quality attribute. Simply naming the quality attributes which are often written as:

- the system shall be robust,
- the system shall be highly modifiable,
- the system shall be secure from unauthorized break-in,

is not precise enough to judge an architecture for suitability and is subject to misunderstanding. The first thing in an evaluation is to get clear to which specific quality goals the architecture will be judged. Scenarios are used for this. A scenario is a short statement describing an interaction with the system. These scenarios resemble very much use cases used in object-oriented settings. A scenario is specified as a stimuli and a desired response of the system. Three types of scenarios can be distinguished, use case (typical uses of the existing system), growth (anticipated changes to the system) and exploratory scenarios (extreme changes that are expected to “stress” the system). For example:

- Use case scenario - A uses requests a database report during a peak period and receives it within five seconds (performance).
- Growth scenario - Double the maximum number of tracks to be handled by the system and keep the latency of track data to the screen within 200 ms.
- Exploratory scenario - Improve the system’s availability from 98% to 99.99%.

With the use of scenarios the quality attributes can be prioritized and a mapping of architectural approaches (an architectural descision influencing the quality attributes) to the quality attributes can be made. The mapping shows how the approaches achieve or fail to achieve the desired quality attributes and this results in a list of risks, sensitivity points and trade-off points. Sensitivity points are key architectural decisions which have influence
on highly ranked attributes. Trade-off points are sensitivity points where more than one attribute is affected.

The process of prioritizing the scenarios and quality attributes, the making of the mapping and identifying the risks, sensitivity points and trade-off points are divided in nine steps. These steps are taken in a meeting of the stakeholders.

The steps are as follows:

- **Step 1** - Present the ATAM. The method is described and explained to the assembled stakeholders. This way they know what the goals are.

- **Step 2** - Present business drivers. The project manager describes what business goals are motivating the development effort and hence what will be the primary architectural drivers. The following issues are discussed:
  
  - The most important functional requirements
  - Technical, economical, political and business constraints
  - The most important stakeholders
  - The most important quality attributes and their market needs

- **Step 3** - Present architecture. The architect will describe the proposed architecture, focusing on how it addresses the business drivers. The presentation has to include:
  
  - The technical constraints such as operating system, hardware or middleware which are prescribed
  - The other systems to which an interface should exist
  - The architectural approaches used to meet the quality attribute requirements

- **Step 4** - Identify architectural approaches. Architectural approaches are identified by the architect, but are not analyzed. These approaches describe the important components of the system and how the system can grow or change.

- **Step 5** - Generate quality attribute utility tree. The quality factors that comprise system ‘utility’ are elicited, specified down to the level of scenarios, annotated with stimuli and responses, and prioritized. This is an important step which influences the rest of the analysis.

- **Step 6** - Analyze architectural approaches. Based upon the high-priority factors identified in Step 5, the architectural approaches that address those factors are elicited and analyzed. During this step architectural risks, sensitivity points, and trade-off points are identified. The output is a list of the identified points and risks.
3. Architecture analysis

- Step 7 - Brainstorm and prioritize scenarios. The set of scenarios is prioritized via a voting process involving the entire stakeholder group. The scenarios involve both change and growth scenarios.

- Step 8 - Analyze architectural approaches. This step reiterates step 6, but here the highly ranked scenarios from step 7 are considered to be the test cases for the analysis of the architectural approaches determined thus far. These test case scenarios may uncover additional architectural approaches, risks, sensitivity points, and trade-off points which are then documented.

- Step 9 - Present results. Based upon the information collected in the ATAM the ATAM team presents the findings to the assembled stakeholders and potentially writes a report detailing this information.

3.4 Cases

To get some answers on the research questions two case studies are done. Both studies try to identify improvements with the use of the ATAM. In the Ericsson case first an architecture description had to be made. The iTV architecture was already clear. The making of the architecture description at Ericsson was very time consuming because of the interviews with different stakeholders who each had their own view. The use of the ATAM was therefore pretty easy, the architecture trade-offs were known from making the description. In the second case we have elaborated more on the ATAM and have tried to follow the method as closely as possible.
Chapter 4

Case 1: Intelligent Networks

"An architect's most useful tools are an eraser at the drafting board, and a wrecking bar at the site."

– Frank Lloyd Wright

4.1 The Ericsson Organization

Ericsson was founded in Sweden in 1876 when Lars Magnus Ericsson started a repair shop for telegraph equipment. He soon realized that there was a need for improvement of the telephones that were available. In 1892 he began to manufacture telephones himself. Since then, the Ericsson company has developed into one of the world’s leading manufacturers of advanced telephone equipment.

The product Unit Service Capability servers and Applications (SCSA) is the unit where the Intelligent Networks organization is part of. It is seated in Sweden.

Intelligent Networks

The Intelligent Networks organization has been reorganized last year. Thereby the amount of design centers has been decreased from nine to three. These local design centers (LDC’s) are located in The Netherlands, in Mexico and in Sweden. Apart from the Local design centers there is a central unit which is steering the R&D in the LDC’s and also takes care of the Strategic Product Management, Strategic Market Development and Supply and Support. This unit is also located in Rijen, The Netherlands. Apart from these units there are the global Marketing and sales organizations and the global Supply and Support organizations.

The Intelligent Network mission

The mission of Intelligent Networks is formulated as follows: “We offer an attractive and profitable solution portfolio for the IN market by using maximum creativity in exploiting current technological possibilities fed by customer demands and customer satisfaction”. In the area of products and the way of
working the organization is trying to make a change from a product focused organization to a service focused one. The most basic definition of a solution is that what is delivered to the customers. In many cases this is an application, a management system, a platform but also training and support. Such a combination of products must work together, must be installable together and easy to upgrade from single product to full blown solutions. Besides the solution focus it has become clear that the current Intelligent Networks need to be positioned parallel to the next generation solutions like UMTS also called 3G. The expectation is that for a number of years the current solutions will coexist and therefore have to co-work with next generation solutions.

4.2 Intelligent Networks

This section will describe the basic concepts of ‘Intelligent Networks’. Most of these architectural concepts are general for Intelligent Networks and are not Ericsson specific.

4.2.1 History

The complexity of telecommunications networks is growing rapidly. Networks have grown from the simple connectivity layer between simple telephones to the big international service networks of today. A great contribution to the growth was the deregulation of the telecommunications monopolies which increased the competition and therefore the need of new services. The services are becoming more important these days as they increase the revenue of a subscriber. To deal with the complexity the concept of intelligent networks arose. Intelligent networks consists of a set of principles that make possible

- rapid engineering of new services through the introduction of standardized, general, service-independent platforms for programming

- quick introduction and ubiquitous availability of new services because the whole network is considered as one system instead of a conglomerate of individual nodes

- easier introduction of new technology in the network because the service logic has been rendered independent of the physical implementation of signaling, switching technique and transmission.

In short intelligent networks provide a framework for introducing and enhancing new network services. This framework is based on the separation of the connection layer and the service layer from the OSI model.

4.2.2 Architecture

The most important characteristic of the architecture of intelligent networks is that it makes the service logic independent of the actual implementation of the
transport network. It does not matter if it is a wireless or not or if it is digital or not. The decoupling of the service logic from the network layer makes it possible to create and implement services quickly and easily.

Figure 4.1 shows a normal plain network and Figure 4.2 shows an IN-network.

![Figure 4.1: A plain telecom-network](image1)

![Figure 4.2: A IN telecom-network](image2)

When a new service has to be introduced in the normal network each switch has to be changed to incorporate the new functionality. In the IN-network this functionality is added only in the Service control point (SCP). All the traffic that wants to use the service now has to be routed to the SCP.
4.2.3 Example

In this section an example will be given to illustrate the concepts of an intelligent network.

Starting this year the police in the Netherlands can be called with 0900 - 8844. There is only one telephone number for the whole country and when dialed a call is established to the local police office.

The call is established through a few steps:

1. After dialing the number the nearest switch, the local exchange, recognizes that it is an IN number.

2. The exchange then routes the call to a service switching point (SSP). This is a ‘smart’ switch with some extra features.

3. The SSP recognizes the dialed number and activates the needed service script in the SCP.

4. The script starts running and from there on controls the call.

5. From the extra information from the SSP the number of the nearest police office is found and the call is connected to that number.

This example shows that the actual functionality is centered in the scripts running in the SCP. Thus there is only one point in the network that has to be changed when for example a new office opens. Also adding functionality such as rerouting after working hours becomes easy.
4.2.4 Network Provisioning

The separation of the service layer from the connection layer makes adding or changing subscribers and subscriptions easier. Only the data and scripts in the SCP have to be changed. To facilitate this the SCP has a management interface which is connected to the management system. This makes the management component is another layer above the service layer.

The management system from Ericsson consists of several products. First the Service Management Application System was made, which incorporates a command-line interface to the user called Generic Service Adapter (GSA). This led later on to the SMABase and GUI components, which implement a client-server architecture with a fully graphical user interface.

4.3 Architecture description

This section describes the architecture of the Information and Business service (I&B) version 3.0. The description is based on the “4+1” view model from Philippe Kruchten [Krucht95]. The I&B service is one of the services that Ericsson makes and deploys on the Intelligent Network. It consists of a set of service scripts which offer the functionality to create special telephone-numbers, like free-numbers, premium rate-numbers and universal access numbers. Furthermore it consists of the components needed for creating and updating subscribers and subscriptions this includes a graphical user interface and database application.

The description of I&B is documented in the I&B System Evolution Plan (see Appendix A for details). This document was written during the project and will be a basis for other systems evolutions plans.

4.3.1 Conceptual Architecture

The I&B architecture is embedded in the total IN architecture or network architecture. Two parts can be distinguished (Figure 4.4): the traffic related part and the management part. The traffic part has two layers, the connectivity layer and on top of that the service agent layer.
These two layers take care of all the calls in the network. The management part has two sub-parts, the service creation component and the service management component. The service creation component will not be discussed here because it is not part of I&B. The service management component is a large part of the I&B application and therefore will be discussed. The connectivity layer is just the connection between telephones and has no other functionality than the simple routing and transport of calls. The service agent layer adds more functionality to the network, making it an Intelligent Network. The layer can be divided into the Service Switching Function (SSF) and the Service Control Function (SCF). These functions can be located anywhere in the network and are highly centralized. A call that needs a specific service is now routed to a service switching point, which is connected to a service control point where the service is executed. The separation of the connectivity and service agent layer gives the operator a lot of flexibility to add or change services in his network. A service in the SCF consists of a number of scripts called SIB’s, service independent blocks. The SCP is a runtime environment for those scripts. An operator only has to change the scripts to change a service. This is done via the service management component. The management component is also used for adding and changing subscribers and subscriptions of a service. The service creation component allows easy creation of new IN-services.

Functional view

The functional view shows the functional entities that can be distinguished together with their relations.

Figure 4.5 shows the functions that are involved or functions that implement parts of the I&B service. Other IN-functions that do not affect I&B are left out.
4.3. Architecture description

Figure 4.5: Overview of Intelligent Network functions

Service Control Function and Service Scripts

The service control function provides a runtime environment for the Service Scripts. These scripts define the actual functionality of the I&B service. They define how a call is handled. The SCF takes care of the communication between an instantiation of a service script and other functional entities like the SSF.

The I&B scripts are split into eight groups. The possible feature scripts are shown in Table 4.1. By combining several features a specific service can be created.

The example of the 0900-number of the police can be created with three features. First is the Identification (ID) feature used so that the caller can be traced. This feature is chained to the Statistics (STAT) feature to gather statistics, e.g. how many call attempts or how many actual calls are made. Then the Origin Dependent Routing feature is used to connect to the right police office gathered from the information from the Identification feature.

System Management System

The system management system (SMS) provides an interface for other management components to the SCP and thus to the service. SMS has the possibility to change data directly in the SCP. This also includes the service scripts.
4. Case 1: Intelligent Networks

<table>
<thead>
<tr>
<th></th>
<th>Feature Scripts</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Identification (ID), Statistics (STAT), Universal International Subscription Number (UISN)</td>
</tr>
<tr>
<td>1</td>
<td>Call Screening (CS), Authentication (AUTH), Emergency Routing Plan (ERP)</td>
</tr>
<tr>
<td>2</td>
<td>Ordinary Call Transfer (OCT), Time Dependent Routing (TDR), Origin Dependent Routing (ABR, EBR, RBR, TBR and LDR), Very Important Person (VIP), Call Distribution (CD), Originating User Prompter (OUP), Interactive Voice Response (IVR), Call Prompter Database (CPD)</td>
</tr>
<tr>
<td>3</td>
<td>Call Limiter (LIM)</td>
</tr>
<tr>
<td>4</td>
<td>Call Forwarding (CFC), Call Hold with Announcement/Call Queuing (CHA/QUE)</td>
</tr>
<tr>
<td>5</td>
<td>En Route Announcement (ERA)</td>
</tr>
<tr>
<td>6</td>
<td>Global Outgoing Call Screening (GOCS)</td>
</tr>
<tr>
<td>7</td>
<td>Post Answer Redirection (PAR)</td>
</tr>
</tbody>
</table>

Table 4.1: I&B feature scripts

**Generic Service Adapter**

The generic service adapter serves as an abstraction layer for SMS. It gives an open service independent interface to SMS for other components.

**I&B Service Management Application and SMABase**

The I&B Service Management Application (SMA) allows for the provisioning of the service, adding and changing subscriptions and subscribers. It is based on a 2-tier client server model and developed on SMABase, a framework that provides functionality like language support, authentication and session handling.

**Internet Gateway Webserver**

The Internet gateway web-server (IWS) is a platform for Java/HTML based web-applications for IN-services. The Internet Customer Control (ICC) function is implemented on this platform. IWS provides, for example authorization services to ICC.

**Internet Customer Control**

Internet Customer Control (ICC) allows web-based provisioning for the end-user of the service (thus not the operators). It is a web-based counterpart of the old DTMF customer control where with a normal telephone the configuration
4.3. Architecture description

can be changed. The use of the web allows easier and more powerful customer control. A standard browser can be used to access this function which is now implemented as a Java applet. Although this function is currently implemented in the management component and not, like DTMF, in the service agent layer the implementation is different from the SMA. There is no common business logic. Therefore there is no guarantee that the two implementations behave the same or check the same constraints. This problem will be addressed in the proposed improvements.

4.3.2 Deployment view

The deployment view is the view where the functions from the functional view are mapped on the physical view. Within Ericsson another type of deployment is important. The mapping of the functions to products. This because the IN-organization is product oriented. The organizational structure is built around the products and everybody thinks in terms of products. Therefore a product deployment view is used (see Figure 4.6 and Figure 4.4).

4.3.3 Use cases

The use cases show the possible interactions with users. An Intelligent Network has three types of users. First there is the operator, which uses the network for provisioning of the subscribers and subscriptions. Second there are the users, which makes IN-calls in the network and third are the retail users, which manage their own subscriptions. This is shown in Figure 4.7.

![User Interaction Diagram](image)

Figure 4.7: User interaction with intelligent networks
Figure 4.6: Product deployment view
Remarkable is that the retail users can adjust their subscriptions through the interface in the management component and also through the interface in the traffic component. They offer similar functionality but from a totally different part of the system.

The user-interface in the link-layer is just a telephone or any other type of terminal. The user in this case is somebody who wants to make a call. An IN-call, a call that uses an IN-service is routed to a SSP. The SSP sends a signal to the SCP where the service script is started. The script will further handle the call (connecting to the right number, charging, playing announcements, etc). This is shown in Figure 4.8.

The management part of the network has two user-interfaces, the I&B SMA-client and ICC. A user triggers the interface, which then sends requests to other components. Through SMABase or IWS and GSA the requests are send to SMS where the information is processed and changes in the SCP can be made. The pipe- and filter-style of this process can easily be seen from Figure 4.9 where the interaction from the SMA is shown.

Figure 4.8: Service agent layer interaction
4.3.4 Physical View

The physical hardware and deployment of the system is quite flexible. Especially in the traffic component where the network should be able to deal with many different situations. From small networks at small operators to huge networks at large operators. Figure 4.10 shows a possible configuration of the hardware for the I&B service. The deployment of the functional units is depicted.
4.3. Architecture description

Figure 4.10: Physical view of an Intelligent Network

The connectivity layer of the network consists of many switches which can be from several vendors. These are connected to one or more SSP's. These SSP's are deployed on an APZ type of machine, which is a telecom purpose computer. The SCP is also deployed on an APZ and there is the possibility that the SCP and SSP are deployed together on one machine. Then it is called SSCP. For larger networks there are many SSP on different machines connected to an SCP. For reliability more SCP's can be configured as mated pair so that when one fails the other one can immediate take over all the traffic. There are more interfaces in the network. The SSP takes care of the billing of the calls. Therefore there is a connection to a billing gateway where all the post processing is done. The SSP simply sends the so called tickets with the needed information of the call to the billing system. These are however not IN-related and therefore left out in the picture.

In the management component the functionality is deployed on UNIX and/or Windows machines. In the I&B service only the client application is deployed on windows machines and the other parts on UNIX machines. This can be a bit different in other services where also UNIX-clients are supported or where windows is used as platform for the SMABase functionality. Figure 4.10 shows only one server for both SMAS and SMABase. This is not necessary and can be split in the deployment on different machines.
4.4 Architecture analysis

The analysis is done in a meeting with Hans Vesters (business manager) and Michel Chaudron (external advisor).

Michel first briefly explains the ATAM and the steps to follow. The ATAM analysis will be specially related to the management component and not to the traffic component. The reason for this are the apparent problems in this component:

1. The performance of the management component is too low, both in throughput and latency.

2. The quality of the system is too low. Quality is measured here in the number of trouble reports from the customers.

Some reasons for the problems can be identified:

1. The broad scope of the component. The component must be able to work with many services. Therefore it is made very general and flexible which adds to the complexity.

2. This component is an information system fully based on information system technology. Where Ericsson is a telecom company and has huge competence of telecom systems it has little competence of information systems.

3. The component is built on several third party products which are not as controllable as Ericsson's products. This makes the alignment and versioning harder to accomplish.

Adding the barely existing architecting process to this complex does not make it easier.

In the second step of the ATAM, Hans presents the most important current business drivers for I&B. The main business drivers are:

- Simplicity of the GUI. Operators do not want a complex user-interface but a simple one which is also easy to use.

- The speed of the response from the GUI has to be improved. The response takes too long and is frustrating the users.

- The GUI has to be stable. Which means that it should be highly available and should contain few bugs.

- The GUI should be flexible. Operators each have their own preferences and therefore the GUI should be configurable so that most of them can be accomplished.
4.4. Architecture analysis

- The alignment with other IN-applications. The user interface of all applications should have a common look and feel. This means that the same components should be used and that different projects should exchange more knowledge and source code.

The current implementation can not fulfill some of the desired quality attributes. Especially performance and scalability. The current pipe and filter style is identified as the architectural approach which has the most influence on these attributes. In the rest of the analysis the change to a client server style is evaluated. This new approach should give an advantage in performance and scalability. In step five the quality attribute tree as shown in Figure 4.11 is created. Behind the scenarios the prioritization relative to each other is given using High (H), Medium (M) and Low (L) rankings. The scenarios are prioritized along two dimensions. The first ranking is the importance of the scenario to the success of the system, the second ranking is the degree of difficulty to achieve the scenario. For example, (M,L) means that it is of medium importance to the success of the system and low difficulty to achieve is expected.

The change from the pipe and filter style to a client server style should give a better performance. In the current implementation the performance scenarios can not be met due to the large overhead created in the different filters. The client-server approach will also give more flexibility and it will therefore be easier to scale the system to handle a larger group of users.

During the analysis a few sensitivity and trade-off points are identified.

1. Trade-off between simplicity and customizability
2. Trade-off between costs and advantage of a change
3. Trade-off between local changes and product-line changes

Sensitivity point 1

The more one can customize the user-interface the more complex it is going to be. This opposes the simplicity requirements from the user-friendliness scenarios. A solution can be to move the configuration of the GUI into a separate part, configuration files, and thus not to combine it into one program.

Sensitivity point 2

Another trade-off point is related to the product-lifecycle of the systems. The lifecycle is coming to an end and that means that no big investments in new features are made. Every scenario has therefore a trade-off point with the associated costs. This is the reason why the software migratability scenarios have a low priority. The costs do not outweigh the profits.

Sensitivity point 3

The I&B service is not a stand alone product, but is part of a product-line. This gives another trade-off point. A new feature can be introduced in several
Figure 4.1: Quality attribute utility tree

- **Performance**
  - **Response Time**: On average load (60%) response time 1 second on install button press \( (H,H) \)
  - **Throughput**: With 10 simultaneous users 1 second response time \( (M,H,M) \)

- **Customizability**: It should be possible to customize the GUI with respect to language and feature—windows with no downtime in 80 man hours \( (M,M) \)

- **Availability**: Less than 5 minutes unplanned downtime a year \( (H,H) \)

- **Scalability**: Number of users should scale to 1000 for web—access \( (M,7) \)

- **Simplicity**
  - **Userfriendliness**
    - Simple to use
    - Person can learn to use the GUI within one day \( (I,J,) \)
    - New subscription can be made within 10 button clicks \( (H,L) \)

- **Migratability**
  - **Software**: It should be able to upgrade a running system with less than 15 minutes downtime of the traffic component and 45 minutes downtime of the management component \( (H,L) \)
  - **Validity**: After change a of the API, the validation of the system can be done within 1 hour \( (H,L) \)
ways. As a local change with small changes and low costs, this is called a service specific correction, or as a product-line change where several components have to change. This results in bigger changes and higher costs. The local change introduces the risk of growing diversity in the product, which can result in higher maintenance costs. The architectural approach should be to make as little specific corrections as possible and to solve problems in the product-line. At this moment components behave in different ways, for example the GUI of different services, and that should be changed.

4.5 Analysis results and recommendations for improvements

By making the architecture description many points in the architecture where improvements can be made were found. So even before the ATAM-session it was clear that improvements could be made, this by just using common knowledge. The recommendations and improvements can be split up in two parts. The first part are the improvements on the current architecture which are focused on the management component. The second part are the recommendations and improvements on the architecting process which is currently barely existing. These recommendations are more organizational oriented.

4.5.1 Improvements on the architecture

Performance

Analysis of the architecture described in Chapter 6 shows that a request from the GUI travels all the way down to SMAS and back up again. The layers in between are doing checks and are transforming the data to pass it to the next layer. Each layer provides an interface at a higher level of abstraction than the layer beneath it. This makes it very flexible. This flexibility however has a cost, it generates extra overhead.

There are several options to solve the performance problem. One option is to improve the scalability. This could be done by adding more or faster hardware. This would change the deployment of the components on the hardware as shown in Section 4.2.4. By identifying the performance bottlenecks it is tried to find the largest benefit from a deployment change. This is done in a specific study [RFI] by one of the system managers.

For each component the time that a request takes is given (Figure 4.12). These times come from measurements on the interfaces of a live system.

These figures show that a large part of the time is spent in SMAS. Some further investigation of SMAS learned that the database is far from optimal. Because of the flexibility in the layers above the database the stored procedures inside are very generic. The system has to be able to deal with very many different parameters and services.

This bottleneck, the time spent in SMAS, is very hard to adjust by changing the deployment because the database and the stored procedures are one big monolithic component. It can not simply be split up in different processes
and deployed on different hardware. What can be done, is changing the deployment of the GSA processes. There can run more instances such that more clients can be served. This has the side-effect that there are more locking problems in the database.

The conclusion that can be drawn is that adding scalability by changing the deployment is difficult and not much improvement has to be expected.

Another option is a logical change by changing the layers. For many services there is a lot of flexibility in the different layers that is just not used. This unused flexibility still generates overhead and this is on the critical path. When looking at SMAS the performance study shows that for a request many stored procedures are called. Also some procedures are called several times with the same parameters. A huge improvement could be made when the GSA layer is by-passed and specific service dependent stored procedures are made. These procedures can be fully optimized which should reduce the time spent in SMAS significantly. Also the overhead of GSA will be no longer there.

Quality

The quality of the user-interface is very poor. Customers send a lot of trouble-reports. When looking at the architecture and the current implementation there are several causes that can be pinpointed.

The management application consists of 2 parts, SMABase and the GUI. In the architecture there is a nice interface drawn between them. In the implementation however that interface can not be distinguished. The two components are built as one piece of code where the use of CORBA hides the actual deployment of classes. Furthermore, the powerbuilder code is not well structured and there is no design documentation. This altogether certainly gives raise to a lot of mistakes in the code and thus the large amount of trouble-reports.

Apart from the standard management application there is also a web interface. This is called ICC and is made for retail-users to provision their own subscription. ICC is now built on GSA and incorporates its own business logic. The implementation is very different but the functionality is just duplicated. This means that more software has to be developed and maintained than necessary.

A solution to the quality problem of the user-interface could be a redesign
of the management application where a clear interface (well specified and documented) will be made between the client and server (Client application and SMABase). This can be done for example with the use of an interface description language (IDL). When this is done there will not only be an interface in the architecture but it will also exist in the implementation. The interface could be used by several thin clients (Figure 4.13) which provide only the GUI and the application/business logic would be centralized in the SMABase component. This has another architectural advantage. The application/business logic will be centralized not only for the GUI used on a PC but also for other clients like web clients or specific systems that an operator uses. This means that the ICC application also has to be redesigned to use the new logic.

In the broad architecture definition also the tools that are used should be considered. Now SMABase is built with the application server from Sybase called Jaguar. Jaguar provides the powerbuilder language and the client is now written in powerbuilder. This language is not designed to build large user-interfaces. It is too much database-oriented for that. The many bugs that are found in the GUI can partly be explained by the use of this language. When the management application is redesigned and a common application/business logic is introduced, another language and tools should preferably be used. Jaguar provides also a runtime environment for Enterprise Java Beans and Java. These are excellent mechanisms for implementing this middleware. First of all, it is platform independent so that different platforms can simply be supported. Another advantage is that nowadays it is easy to find skilled designers with Java knowledge in contrast with powerbuilder designers. The use of Java with a clear IDL interface could also mean that the expensive Jaguar applications server is not needed anymore. Things like persistence and transactions that the application server provides are not needed for the simple interface. This will be very attractive for the operators which will have to invest less.

4.5.2 Recommendations for the architecing process

Many times during my project I saw architectural problems which had little to do with technical aspects but were more organizational problems. First of all, the IN organization was a product organization. This means that the organization was built around the different products. Each product had its own
managers and design unit and they were responsible for their product. The products are the components from the product deployment view shown in Figure 4.6 combined with the services that are built with them. This organization had many problems with the alignment of the products. A customer does not want to buy separate products but wants to buy a complete IN system. Not only the alignment in terms of time and interoperability is an issue but also the technical alignment. A few examples of the differences:

- SMABase has a requirement that it can be deployed on UNIX and Windows NT machines. All services except one are using UNIX machines for deployment and some of them have not even verified if the service works on Windows NT. This means that now a customer has to buy two platforms when he wishes to use several services. There is also more development needed to support two platforms.

- The GUI used for the service provisioning has a different look and feel for every service. Some are made in Powerbuilder based on SMABase and some use an automatically generated GUI.

- Documentation of products differs widely. Some products have elaborate documents about dimensioning and network impact while others have not.

This results in the use of different technology, different interfaces and the use of different standards. Nobody is thinking about the product-line issues.

In an article from Michiel Perdeck [Perdeck01] some critical success factors are given for working under architecture and these factors have to be implemented in the organization to be successful. The factors are divided into three parts as shown in Figure 4.14. First a blueprint of the architecture has to be designed based on input from the market, the organization and management. This blueprint should be the technical equivalent of the strategical and tactical business plans that are written now within Ericsson and can be incorporated in the System Evolution Plans documents that already exist. With the help of that blueprint an architect advises management and projects. The blueprint creates consistency in the advises. The final part is the verification of the blueprint with the actual implementation and check if the principles and guidelines are really being followed during projects. Figure 4.14 shows also two lines back to the design part. This is to make clear that in the process new insights can be obtained which lead to a change of the blueprint.

The current situation with the local design units and different units for platforms and services should make the architects focus on the interfaces of the components. This emphasis is not there right now, which leads to units who are quarreling about changes. Also cross-component changes are made by units, which should not be allowed. A nice example of this situation is the service specific correction. When a service is designed and some problems are found in for example the platform (e.g. SCF, SSF) a trouble report is written and sent to the responsible unit. This unit can state that this is not a fault on their system because it is not in their requirements and reject the
request to make a fix. The service then has to make their own fix (the so called service specific correction). This can be done with the help from the platform but the responsibility lies in the other unit. Also due to the lack of interface management between components the platform does not know all the corrections and can not change the problems in a next release. The services have to map the corrections themselves in a new release. This situation is far from ideal and another organizational structure with a central architectural unit as Jan Bosch describes in this article [Bosch01], should be considered. In such an organization the critical success factors as stated above can come into existence and a real architecting process can be created.

This unit, the architecture board, should have the knowledge and authority to:

- Ensure technical alignment of the components.
- Decide on product-line issues.
- Settle disputes on where changes in the architecture need to be made.
- Verify the implementation with the architecture blueprint.

The organization is already trying to change and is trying to get a so called 'solution' focus. This is mainly done in the steering organization where the word 'solution' is placed in front of the function-name of people. This is also done for the system managers, who are technically responsible for the solutions. These solutions system managers should only get more powerful and should become the real architecture board.
4. Case 1: Intelligent Networks
Chapter 5

Case 2: Interactive Television

"One has to build the right system and one has to build it right."
- anonymous

5.1 Setting

CMG is an international IT-services and solutions group that applies information and communication technologies to create added value for its customers. Interactive television is one area where CMG is active. In the Netherlands the market today is very small but the growth expectations are high and therefore CMG invests in the technology. The competence centre Technical Software Engineering (TSWE) at CMG Eindhoven has developed an architecture for the deployment of interactive TV. This includes the broadcasting of applications to set-top boxes together with the digital videostream, and the creation of a return channel via a cable modem, short message service (SMS) or an interactive voice response system (IVR) (see Figure 5.2). The broadcasted applications are executed on the set-top boxes and some information (e.g. answers on questions) can be sent back to the return servers.

A few concepts for interactive programs exist. A simple example is a quiz, viewers can answer the questions that are asked in a program and by answering them a viewer can win a prize. The development of a return channel makes it possible to immediate broadcast the results in the broadcast channel. Thereby creating real interactive television.

The strong point about CMG's offering is the integration of different systems. Not only the development of a set-top box application but also the return channel through SMS, voice response and cable can be offered.

5.2 Architecture description

The available architecture documents use six views which are based on three levels of abstraction as shown in Figure 5.1 [Dinther]. For every level a static and a dynamic view is described.
5. Case 2: Interactive Television

Figure 5.1: Architecture meta-model views

The conceptual view is a black box view and describes the context and use of the system. The logical view is a white box view and describes the design. The physical view describes the implementation.

5.2.1 Context

Static view

Figure 5.2 describes the environment of the return- and application-servers.

Figure 5.2: Interactive TV overview

A set-top box application is sent through the broadcast channel to all the set-top boxes. The set-top boxes send information back via the cable-modem to the return-servers. The return-servers, which receive all the information, insert the information in the results database. An application server can get the results out of the database, process them, and insert them in the broadcast
stream. In a pilot project for a broadcast organization a connection to MM-base, a content management system is made. Some statistics from the results combined with results from other media are sent back to the application server.

**Dynamic view**

The set-top boxes make a TCP/IP connection to a return-server. The connection is made at the start of the program and is kept open throughout the program. For scalability and availability reasons a load-balancer can be put before the return-servers. This way the load is equally spread over the available servers. This makes it also possible to add or change a server on the fly.

![Figure 5.3: States of a return-server](image_url)

**5.2.2 Logical**

**Static view**

A return-server has two subsystems, the TCP/IP server and the database client. It accepts connections on a specific port and receives data on it, which the database client writes in the central database. The application-server has the following subsystems also shown in Figure 5.4.

1. Scheduler: Synchronize with the television-broadcast.

2. Persistent store: Read data from the central database

3. XML-Parser: Converts the results to XML and converts the XML to the OpenTV [OpenTV] standard.

4. OpenTV interface: Interfaces with OpenTV, coordinates the specific things like gmake and the transport of data to the OpenMUX server.
5. Case 2: Interactive Television

Figure 5.4: Layers of the application-server

Dynamic view

The interaction of the application-server as used in the pilot is described in Figure 5.5. When an event arrives from the scheduler the application starts the task of sending or retrieving information. For simplicity the return messages are left out in Figure 5.5.

Figure 5.5: Interaction of the application-server

Figure 5.6 shows the steps that are followed to update the application-data in the set-top boxes. As shown, the application has to be recompiled and sent back to the boxes.
5.3. Architecture analysis

Figure 5.6: Compilation infrastructure

5.2.3 Physical

In Figure 5.7 the systems in the network are shown.

Figure 5.7: Logical overview

The TCP/IP server sockets on the return-servers are multi-threaded. To prevent numerous context switches, 2500 clients is regarded as the maximum amount of clients per server.

Considering that the application-server has to do the tasks in series the application can be monolithic. Furthermore the application-server has to do the work after the return-servers are finished. This means that the application-server can be combined with a return-server on one machine.

5.3 Architecture analysis

In this case we tried to follow the nine ATAM-steps as closely as possible. The ATAM-analysis was held in two sessions. Both sessions were attended by
Jan Verhoeven (architect), Wim Bus (business manager) and Michel Chaudron (external advisor).

5.3.1 First session

First the ATAM and the goals of the session are explained. Apart from evaluating the architecture of iTV also the practical use of the ATAM should be analyzed. The ATAM is explained according to the nine steps stated in section 3.3.

In step two Wim presents the business drivers and the context. CMG offers both applications for set-top boxes and complete solutions including a return channel. In this context CMG has designed an architecture for the implementation of a return channel which can be used by different broadcast corporations. This architecture is still in the first phase of development and therefore it is a suitable test-case for analysis and improvement. The NOS as a customer wants to be able to offer a return channel to different broadcast corporations, this in conjunction with IVR and SMS features. This means that the system has to be very generic because it has to work with different cable networks and different set-top boxes. The system furthermore should incorporate a few interfaces to couple existing IVR and SMS systems and to make the data available for the broadcast corporations. The NOS has chosen for a pragmatic approach. This means that they first want a solution where the costs are relatively small and which works. This approach can be explained by the small budget they have and the small market. Furthermore the NOS wants to use open standards and use systems they have already. For example Linux and MySQL. Other preconditions are quality and time to market. The system has to work and television viewers cannot be left alone with a crashing device or a return channel that does not work. With time to market we mean that the deadlines have to be reached. A program will be broadcasted at a specific time and that can not be changed. In the future developments are foreseen in the area of tooling and the use of standard components.

In step three Jan gives a presentation of the architecture used in the NOS proposition and shown in section 5.2. By means of many questions from Michel the architecture gets clear for everyone.

In step four the following architectural approaches are identified:

- Decoupling of the return channel and data querying part of the broadcast corporations.
- The use of the XML, SQL and HTML protocols for different parts.
- The use of a load-balancer for the return servers.
- Only allowing one-way traffic from set-top box to return server.
- The connection of IVR and SMS to a web-server instead of a central database.
- No link with the name, address and city information of users from the cable companies.
• The availability of the gathered information through a web-server creating a single point of attention for security.

In step five the quality attribute tree as shown in Figure 5.8 is made. Most of the scenarios were worked out in detail later because of the lack of time at the meeting.

5.3.2 Second session

In the second meeting we continued with the detailed quality attribute tree. Some scenarios were discussed and clarified. Sensitivity points identified.

1. Decoupling of the return channel and the data querying part.
2. The use of a central database based on MySQL.
3. The use of XML, HTML and SQL for different parts
4. The use of a web-server to obtain data from the database.
5. The open connections throughout a program between set-top boxes and the return-server.

Sensitivity point 1

By decoupling the two parts the system gets more flexible. The gathered data remains available and can be processed in different ways without the need of the return channel. It now is simple to make a different view on the data when necessary for an application. The decoupling makes it impossible to push the data into the video-stream. Instead the data has to be pulled out of the database. This leads to a larger latency in the loop.

Sensitivity point 2

The use of one central database based on MySQL has impact on the scalability and performance. All applications are updating the same table in the same database. This means that a lot of performance in the database is needed. Apart from the updates many queries from the broadcast companies arrive. This gives the possibility of locking problems. The choice of MySQL as database management system gives furthermore a risk to performance. This because MySQL uses table-locking and there is no support for transactions so updates and selects have to wait for each other.

In the light of availability the central database is not optimal. When something goes wrong with the database all the applications do not work anymore. Furthermore there is a possibility of interference of applications, one application can slow down the other. MySQL has no support for replication what makes it harder to give a solution which is reasonable safe.
5. Case 2: Interactive Television

Figure 5.8: Quality attribute utility tree
5.4. Analysis results and recommendations for improvements

Sensitivity point 3

The use of different protocols for different parts has been chosen. This gives the advantage of having the most appropriate protocol for each part. A disadvantage is that the performance can go down because the data has to be converted a couple of times to the proper format.

Sensitivity point 4

The use of a web-server to obtain data from the database is simple and flexible. For broadcasting organizations it is also easy to build their own interface for their specific needs. For security reasons one has to be careful that an application does not allow access to other data than the data of the specific program in the central database. An advantage of the web-server is the protection of the database. The web-server is the only point of attention for security.

Sensitivity point 5

The return channel consists of open TCP/IP connections between the set-top boxes and the return-servers. These connections stay open throughout the execution of an application resulting in many concurrently open connections on the return-server. This is bad for the scalability because in the current situation a return-server can only have about 2000 concurrent connections. One option is to close the connections during the execution of an application. This however results in a trade-off point. When the connections are closed during the application, the return-servers will have less concurrent connections and the scalability will therefore be better. However the performance will go down. The reconnecting of a set-top box will cost more time and bandwidth.

5.4 Analysis results and recommendations for improvements

During the second session for each of the identified sensitivity and trade-off points another architectural approach is discussed and evaluated to see if it is a better one.

Sensitivity point 1

The goal of the new approach for sensitivity point 1 was to minimize the latency of the whole loop. By changing the decoupled return channel and querying part such that the data gets pushed to the video-stream instead of pulled out of the database a nice pipe- and filter style is used. This leads to a more continuous stream of data and hence less latency. The return servers not only have to insert data in the central database but with this change they also have to send a message to the multiplexer of the video-stream.
Whether this change has an advantage over the current situation is unclear. The precise effect on the latency is not known and the question is if it outweighs the more complex implementation.

**Sensitivity point 2**

The use of a central database based on MySQL will not scale very well. Therefore the presented approach for sensitivity point 2 is to use different databases for each application. When needed for performance the databases can be deployed on different machines, this makes it much more flexible and scalable. Furthermore the use of MySQL which does not support replication or transactions does not make it failsafe. There are many databases (e.g. PostgreSQL, Oracle, Sybase) which have better functionality and for the availability a change can be considered.

**Sensitivity point 3**

For sensitivity point 3 the new approach was to use XML for all the communication between the components. This means that no conversion has to be done between the different protocols which are used in the current architecture. This makes the system less complex and the performance could possibly be better. However within CMG little knowledge is available on the use of XML in combination with databases. In the new situation clearly a type of XML database is needed and there is no experience with the behaviour of such a database. Therefore it remains to be seen if the new approach is advantageous.

**Sensitivity point 4**

An option for sensitivity point 4 is to give the broadcasting organizations a direct connection to the database such that they can develop their own application and process the data in the way they want. This gives the broadcast organizations more flexibility and leverages the need for a web-server. Many connections are made through a firewall direct to the database. This makes the security harder to accomplish.

**Sensitivity point 5**

A change for sensitivity point 5 is the use of UDP instead of TCP/IP. UDP is connectionless and that is advantageous for the performance, no connections have to be established and kept open during the execution of an application. A side-effect is that there is no guarantee that all packets will arrive. When there is much traffic, packets can be dropped. For some applications, a sales application, this is not acceptable but for others, a quiz, this is not a problem.
5.4. Analysis results and recommendations for improvements

5.4.1 Quantifying

sensitivity point 1

A solution for the lack of quantitative methods as described in the former paragraph sensitivity point 1 is searched. In the literature little can be found on metrics on a whole architecture or architecture approaches. The problem is hard to solve in a generic way, therefore our approach was to look at some sensitivity points as identified with the ATAM and try to make a model for the important quality attributes relating to that point. The model should be able to give a decision on the preferred alternative.

Figure 5.9: The data-flow in the current architecture

Figure 5.10: The data-flow in the proposed alternative architecture

The first model was made for sensitivity point one, the decoupling of the return-channel, where the latency is a concern. An alternative approach, the coupling of the return-server with the application-server thereby bypassing
the database was evaluated (Figure 5.10). This sensitivity point was picked because it influences highly ranked scenario’s and it looked like it was possible to give a more quantified comparison on the alternative.

As shown in Figure 5.9 the worst-case response time is the sum over $T_1$, $T_3$, $T_5$, $T_4$. Where $T_4$ is the polling-interval of the application server. In the proposed alternative the latency is the sum over $T_1$, $T_2$, $T_4$, $T_5$. This means that $T'_3 < T_2 + T_3$ should hold for the latency to improve.

A few questions arise:

1. How much does the latency improve?

2. Is the improvement necessary to fulfill some scenario’s?

3. Does the advantage in latency outweigh the added complexity of the application server?

Only a kind of implementation can give precise answers to these questions. An implementation can be a simple prototype which can easily be made. Such a prototype can show unexpected influences which are not present in the model. For this alternative a prototype with the use of a database, some clients (the return-servers) and a application which select data from the database is made. With this simple prototype a few experiments are made (see Appendix B for the details). The experiments showed as expected a large performance increase: $T'_3$ is clearly smaller then $T_2 + T_3$. Knowing this a better decision can be made. However the question remains if the lower latency is better than the simplicity and persistency of the alternative with the database.

**sensitivity point 5**

In the current architecture TCP-connections are used for communication between set-top boxes and return-servers. This gives a bottleneck when the number of set-top boxes grows. A return-server has a maximum number of open connections. In a pilot-project a Windows-implementation is used and that gave a maximum of 2000 concurrent connections. In the tests (see Appendix C for details) we tried to use a Linux implementation, to see if we could increase the number of concurrent connections. Furthermore a change to UDP is considered and this is also tested. With the use of UDP there is no guarantee that all packets arrive but there is also less overhead. The tests show that indeed a considerable improvement can be made by changing the implementation to use a Linux server. When using UDP no connections have to be made and this changes the bottleneck to the available bandwidth. One server can handle all the messages that arrive with a fully used 10 Mbit/sec LAN-connection.
Chapter 6

Comparison and analysis

"An idea which can be used once is a trick. If it can be used more than once it becomes a method."

– George Polya and Gabor Szego

In this chapter the two case studies are compared and analyzed.

6.1 Comparison

Although the two systems which are analyzed are quite different they have both a large complexity. Many quality attributes are important and from the business drivers huge constraints are placed upon the systems. They both have components, which are based on different platforms and which have to interact and have to fulfill some end to end requirements.

A big difference however, is the fact that the I&B service is already a working product and has been sold to many customers where as the iTV platform is in the design phase and not fully implemented yet. Furthermore the I&B service is part of a product-line called Intelligent Networks which brings additional complexity. Not only from a technical perspective but also from an organizational one where different projects and units have to be aligned.

6.2 Analysis

Because of the differences between the cases, different aspects of architecture analysis and the ATAM could be observed.

6.2.1 Case 1

Before an architecture analysis could be done on I&B, the architecture had to be described. There was no real architecture description available. With the use of interviews, design documents and even some re-architecting from the implementation the description was made. Making this description showed already some aspects of the architecting process which have to be taken into
account. By interviewing different stakeholders it became clear that different views of the systems existed and that these were not consistent with each other. Using the right consistent views and explaining them such that they were understood by everyone was not an easy task. This was aggravated by the fact that the stakeholders had little architectural knowledge. The creation of a consistent view of the system was beneficial. Different stakeholders now could have a clear discussion about problems they saw with each other. It also showed some organizational problems. These problems arose largely due to the fact that the systems are made of components which are developed by different units. Because of this structure the interfaces and responsibilities have to be very clear and this was not the case. In the literature on architecting and in the analysis methods the emphasis is on the technical aspects of systems. Architecture looks broader than the functional requirements but fails to include many business aspects.

6.2.2 Case 2

In the second case we have focused more on the ATAM method. We have tried to identify the advantages of using the method and applicability to complex systems. The product-line and organizational aspects of the Ericsson case were not present in this case.

The practical result of the two ATAM-sessions which were held was the documentation of the sensitivity and trade-off points and thereby identifying the risks. This documentation could have been made without going strictly through the nine ATAM-steps. But the steps force one to think about the architecture and to document it. That is a very positive aspect. A weak point we found was the lack of quantitative methods. In order to verify if an architectural approach can fulfill a scenario some aspects have to be quantified. One can do this with some back of the envelop calculations but that is a mere ad hoc approach and is not well structured. An important aspect that also is not incorporated is the cost aspect. From a business perspective the costs are a great concern. They play a role in many technical decisions and these can not be found in the ATAM method.

6.2.3 Research questions

With the work done on the two cases some of the research questions (see Section 2.3) can be answered.

- How well is ATAM applicable?
  The two cases, which are very different showed that the ATAM is generic enough to be applicable on many systems. The ATAM does not concentrate on a specific quality attribute and therefore by emphasizing different attributes in different situations the ATAM can be adjusted to specific needs of a system.

- How does the theory work in practice? Is ATAM suited for every system? Does the ATAM have a benefit above other methods?
The use of the ATAM as shown is not a strict walk through the nine steps. Each system needs its own particular adjustments. When improvements are evident like in the Ericsson case, the ATAM-analysis can be done very quickly. In the CMG case on some points extra information was needed in order to make a good decision. Tailoring the used method to the system and its environment therefore is needed to obtain a good result. This may also be the reason why there are so many different methods (see Section 3.2). Existing methods are adjusted to specific settings and form a better method for the specific cases. As seen before, a silver bullet, a universal method suited for everything does not exist.

- What are the benefits of the ATAM?
  In both cases it showed that it is very useful to document the sensitivity points and risks. This is also the main benefit. The documentation and the common view from different stakeholders on the system is very valuable. As seen in the Ericsson case stakeholders from a different background have different ideas and views and they tend to have those until its too late to make adjustments.

- How large is the overhead of the ATAM?
  The overhead of two days of ATAM-sessions is clearly less compared to the benefits of better documentation and a common view. Furthermore both cases showed that improvements can be made, which in general can save a lot of money because it is more expensive to adjust implementations later on in the development process.

- How well are the business drivers handled with the ATAM? Which aspects are not covered with the ATAM?
  The ATAM is developed from a technical viewpoint and that can be seen in the aspects which are not addressed. The Ericsson case showed that organizational aspects can have a large influence on the architectural process. The ATAM cannot deal with these influences. When there are more units involved, the emphasis should lie on the interfaces between them, these should be well defined. Another aspect that is not covered are the costs. Many decisions are made with costs in mind and are not based on the technical arguments.

- Can some metrics be given for quality attributes?
  Measuring the quality is very hard. Making a change in the architecture influences many times more than just one quality attribute. Each attribute has to be measured and a weighted sum has to be made. This means that quality attributes have to be prioritized in order to conclude if a change will have a positive effect or not. Metrics for a single quality attribute are already hard to find and certainly not usable in a generic way. The interaction between attributes is a whole different problem.
6. Comparison and analysis
Chapter 7

Conclusions

"Mere facts are for children only. As they begin to point towards conclusions they become food for men."

– Edmund Selous

Many difficulties in software engineering today arise not only from the ever growing complexity of systems but also from the important non-functional requirements. The implementation of functional requirements has, through good design and with the right methods which have evolved the last 30 years, been manageable. The non-functional requirements however are harder to fulfill. The techniques are young and have not matured yet. The architecting process tries to provide insight in the quality attributes of a system as early as possible and tries to get control of the non-functional requirements.

7.1 Cases

The two case studies showed how architecture and architecture evaluation works in practice. Both cases had some striking elements. In the Ericsson case the lack of business and organizational aspects in the architecture was clearly visible. In the case at CMG there was a clear need for a more quantified approach.

These aspects show that a generic method like the ATAM has to be used differently according to the specific setting of the system and business. It makes also clear that architecture is not just a technical artifact anymore. Many other specializations like competence management, project management and sales activities are involved. Making everyone work together in the architecting process is not easy. People have different views and the architect has to make them consistent within an architecture. Although the architecting process and use of the ATAM showed not easy, in both cases some recommendations for improvements could be proposed. The improvements demonstrate that the architectural approach and the used method are beneficial for the development of the systems and thus should lead to a better quality of the systems. In both cases the proposed improvements had a focus on performance. In the Ericsson case a change from a pipe- and filter style to a client-server
style should give better performance and result in lower latency. In the CMG case the change to a pipe- and filter style should result in lower latency. This shows that the use of architectural styles is very dependent on the situation and in one situation the effect will be different from another situation. The effects of an architectural approach are thus not generally predicatable, which makes the use of metrics also hard. One would like to measure the change of an architectural approach in terms of the quality attributes so that a clear decision can be made to adopt to the new approach or not. The experience from the cases learns that there exist so many uncertainties that only with a kind of prototype the real influences can be measured.

7.2 Options for future work

7.2.1 Business aspects

The architecting process and architecture analysis do not stand alone. There are always business and organizational decisions which play a role. Still almost all current methods take only the technical aspects into account. A lot of research needs to be done on how to integrate some of the business aspects in the current methods. When there is a product-line like in the Ericsson case one for example has to focus on the interfaces between components. Also a clear hierarchy is needed for solving disputes between different units. The best way to implement this and to integrate it in the architecting process is unclear.

7.2.2 Quantitative aspects

In some situations one would like to quantify some of the quality attributes in order to make a good decision about a sensitivity point. This became clear in the iTV case. Making a quantification is hard, each alternative has both some advantages and disadvantages. Moreover some decisions are influencing others. Creating a generic method for the modeling of the quality attributes of an architecture is work for the future. It will make an architecture analysis less abstract and probably more useful. It will also give better insight in the relations between attributes which are now mostly hidden.
Used terms and abbreviations

ALMA  Architecture level modifiability method
ATAM  Architecture trade-off analysis method
CBAM  Cost benefit analysis method
CORBA  Common object request broker architecture
CSI   Capability Set 1
CSI+  Ericsson superset of CSI
ETSI  European Telecommunications Standards Institute
FAAM  Family - architecture analysis method
GSA   Generic service adapter
GUI   Graphical user interface
HTML  Hypertext markup language
I&B   Information and business
ICC   Internet customer control
IN    Intelligent Networks
INM   Intelligent networks management interface
IP    Internet protocol
ISDN  Integrated services digital network
ISUP  ISDN user part
iTV   Interactive Television
IVR   Interactive voice response
IWS   Internet gateway webserver
MML   Man machine language
OpenMUX  Multiplexer which adds the application to the broadcast stream
USED TERMS AND ABBREVIATIONS

OpenTV Platform for applications on set-top boxes
OSI Open system interconnection
PBX Private branch exchange, company switch
PLEX Programming language used for making service scripts
RPC Remote procedure call
SAAM Software architecture analysis method
SAEM Software architecture evaluation model
SAF Service agent framework
SBAR Scenario-based architecture re-engineering
SCF Service control function
SCP Service control point
SDF Service data function
SDK Service development kit
Service script Combination of SIB’s which implements some functionality.
set-top box Device that enables a television set to become a user-interface for different applications broadcasted with a digital broadcast stream.

SGW Statistic gateway
SIB Service independent building block
SMA Service management application
SMAS Service management application server
SMS Service management system
SMS Short message service
SQL Structured query language
SSCP Service switching and control point
SSF Service switching function
SSP Service switching point
STB Set-top box

Switch A current telephony network is circuit switched. This means that for every call a channel is created. The switch routes the calls and creates the channels.
TCP  Transmission control protocol
UML  Unified modeling language
XML  Extensible markup language
USED TERMS AND ABBREVIATIONS
Bibliography

[METHOD] Liliana Dobrica and Eila Niemel"a,
A Survey on Software Architecture Analysis Methods,

[SEI] Software Engineering Institute,
http://www.sei.cmu.edu/

[UML] James Rumbaugh, Ivar Jacobson and Grady Booch,
The unified modeling language reference manual,
Addison-Wesley, 1999

[REVIEW] Lennart Söderberg,
Architecture for Intelligent Networks,
Ericsson Review 1999

[IEEE1471] IEEE Computer Society,
IEEE Recommended Practice for Architectural Description (IEEE Std 1471-2000),
September 2000

[Krucht95] Philippe Kruchten,
Architectural Blueprints - The “4+1” View Model of Software Architecture,

[Dinther] Yolanda van Dinther,
Architectural Modeling, introducing the Architecture MetaModel,

[GM1] Gerrit Muller,
Product Families Business Analysis and Definition,
http://www.extra.research.philips.com/natlab/sysarch/index.html,
BIBLIOGRAPHY

May 2000

[GM2] Gerrit Muller, 
Roadmapping, 
http://www.extra.research.philips.com/natlab/sysarch/index.html, 
May 2000

[GM3] Gerrit Muller, 
The System Architect; Meddler or Hero?, 
http://www.extra.research.philips.com/natlab/sysarch/research.html, 
October 2001

[CAN] Canvas, 
Ericsson Intranet - Canvas (SCSA intranet)

[Iperf] Iperf, Network performance measurement tool 
http://dast.nlanr.net/Projects/Iperf/, october 2002

[OpenTV] OpenTV, Information on the OpenTV middleware 
http://www.opentv.com/

[Shaw89] Mary Shaw, 
Larger Scale Systems Require Higher-Level Abstraction, 
1989, proceedings of fifth International workshop on software specifi- 
cation and design IEEE Computer Society, p 143-146

[Bosch01] Jan Bosch, 
Software Product Lines: Organizational Alternatives, 
May 2001, ACM international conference on software engineering 
proceedings p 91-100

[Hammer97] Dieter K. Hammer, 
The many Aspects of an IT-Architecture, 
1997, Proceedings of the 1997 Workshop on Engineering of Computer- 
Based systems

[Dolan01] Tom Dolan, 
Architecture Assessment of Information-System Families, 
2001, PhD thesis TU/e
[Nokia] Claudio Riva and Ynag Yaojin,
An Environment for Architecture Reconstruction,

[RFI] Martijn Zegers,
Request for information on throughput improvements for MVPN,
RFI 01610_PA9

[SEP] Joris Bajjens,
System Evolution Plan for I&B,
ELN/Y-01 067 Uen PA2

[Sarch98] Len Bass, Paul Clements and Rick Kazman,
Software Architecture in Practice,
Addison Wesley 1998

[ATAM] Paul Clements, Rick Kazman and Mark Klein,
ATAM: Method for Architecture Evaluation,
Carnegie Mellon Software Engineering Institute, Technical report
CMU/SEI-2000-TR-004

[ESA02] Paul Clements, Rick Kazman and Mark Klein,
Evaluating Software Architectures, methods and Case Studies,
Addison Wesley 2002

[Gilb88] Tom Gilb,
Principles of software engineering management,
Addison Wesley 1988

[Royce70] Royce, Winston W.,
Managing the Development of Large Software Systems,
IEEE 1970

[Perdeck01] Michiel Perdeck,
Kritieke succesfactoren voor werken onder architectuur,
Architectuur & Infrastructuur, nummer 2, 2001

[Moore01] Geoffrey Moore,
Inside the Tornado, Marketing strategies from Scicicon Valley’s Cutting Edge,
november 2001
Appendix A

System Evolution Plan

This appendix shows the system evolution plan of I&B.
A. System Evolution Plan

I&B SEP

Information and Business architecture document/System evolution plan.

1. Management summary
   2

2. Introduction
   2

2.1 Architecture
   2

2.2 Purpose
   2

2.3 I&B service
   2

2.3.1 Freephone
   2

2.3.2 Premium Rate
   3

2.3.3 Universal Access Number
   3

3. Acronyms and abbreviations
   3

4. Conceptual Architecture
   3

4.1 General IN-architecture
   3

4.2 Current architecture
   4

4.3 SCF and Service Scripts
   5

4.4 GSA
   6

4.5 SMS
   6

4.6 I&B SMA
   6

4.7 IWS
   6

4.8 ICC
   6

5. Deployment
   7

5.1 Interaction
   9

5.2 Network Requirements
   11

6. Components
   12

6.1 SMA Base
   12

6.2 SMS and GSA
   14

6.3 IWS, ICC
   14

6.4 SCP and service scripts
   14

6.5 SSP
   15

7. Changes
   16

7.1 Version 3.1
   16

7.2 Long term
   17

7.3 Document changes
   17

8. Revision History
   17

9. References
   17

Appendix:

User guide for making architecture diagrams
1 MANAGEMENT SUMMARY

The Information and Business Service is an umbrella service covering services known as: Freephone, Premium Rate and Universal Access number. The Ericsson implementation of these services is called Information & Business. It contains a large number of features like Queue-handling and Originating User Prompter and components like Internet Customer Control.

I&B is targeted at companies/businesses which have some kind of telephone contact with their customers. There for a lot of flexibility is incorporated in the service to fulfill the different customer requirements.

This document describes the architecture of I&B and tries to define the evolution of the product. With the purpose to accommodate the product creation process and maintenance.

2 INTRODUCTION

2.1 ARCHITECTURE

Architecture can be defined as: "The structures of the system, which comprises the components, the externally visible static and dynamic properties of those components and the relationships among them". Architectures exist on different levels and this document will concentrate on the high-level system architecture of I&B (Information and Business). Which includes the services itself and the management of it but also a picture is shown of the whole environment in which I&B operates. I&B 3.0 is used as a starting point for this document. From there on the future architecture will be defined, thus making this a living document.

2.2 PURPOSE

Drivers for the development of the I&B architecture and IN architecture in general are the quality aspects and further development. The enormous complexity of the product and rapid deployment and change has slowly degraded the overall quality. A good architecture could enhance the quality by identifying weak points and provide the means for proper further development. Thus as a starting point for improvements.

Apart from describing the architecture also the evolution is described. It set directions for the long-term development of the service. This means that this document must be consulted before each change of the service and that it has to be updated.

2.3 I&B SERVICE

The I&B service is compatible with both fixed and mobile platforms, the I&B service is a multi-purpose service with a flexible and modular design. The flexible structure allows the service provider to choose the features for a subscription that can best satisfy the customer. Furthermore, subscribers can subscribe to the Internet customer control option. This enables subscribers to view, modify, and delete subscription data to certain I&B features on the Internet.

2.3.1 Freephone
Freephone services allow service users to call service subscribers free of charge. The service subscriber pays the cost of the call. The Freephone service is widely used by companies that wish to encourage customers to call them, such as travel agencies and order departments for all kinds of consumer-product retailers. Many countries have a special number series reserved for freephone services; for example, 0800- is used in The Netherlands and 020- is used in Sweden.

2.3.2 Premium Rate

Premium Rate services allow service subscribers to provide service users with information services at charging rates that are higher than ordinary calls. The cost of the call is proportional to the duration of the call. The revenues generated by the calls are divided between the service subscriber and the network operator.

2.3.3 Universal Access Number

The Universal Access Number allows a customer to be identified with only one number for all his subscriptions. Thereby easing the administrative tasks for the operator.

3 ACRONYMS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSA</td>
<td>Generic service adapter</td>
</tr>
<tr>
<td>HLS</td>
<td>High Level Script</td>
</tr>
<tr>
<td>I&amp;B</td>
<td>Information and business</td>
</tr>
<tr>
<td>ICC</td>
<td>Internet customer control</td>
</tr>
<tr>
<td>INM</td>
<td>Intelligent networks management interface</td>
</tr>
<tr>
<td>ISUP</td>
<td>ISDN user part</td>
</tr>
<tr>
<td>IWS</td>
<td>Internet gateway workstation</td>
</tr>
<tr>
<td>MML</td>
<td>Man Machine Language</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote procedure call</td>
</tr>
<tr>
<td>SAF</td>
<td>Service Agent Framework</td>
</tr>
<tr>
<td>SCF</td>
<td>Service control point</td>
</tr>
<tr>
<td>SDK</td>
<td>Service development kit</td>
</tr>
<tr>
<td>SGW</td>
<td>Statistic gateway</td>
</tr>
<tr>
<td>SIB</td>
<td>Service Independent Building Block</td>
</tr>
<tr>
<td>SMA</td>
<td>Service management application</td>
</tr>
<tr>
<td>SMAS</td>
<td>Service management application system</td>
</tr>
<tr>
<td>SMS</td>
<td>Service management system</td>
</tr>
<tr>
<td>SSF</td>
<td>Service switching point</td>
</tr>
</tbody>
</table>

4 CONCEPTUAL ARCHITECTURE

4.1 GENERAL IN-ARCHITECTURE
The I&B architecture is embedded in the total IN architecture or network architecture. Two parts can be distinguished the traffic related part and the management part. The traffic part has two layers, the Connectivity layer and on top of that the service agent layer (Figure 1). These two layers take care of all the calls in the network. The management part has two sub-parts, the service creation component and the service management component. The service creation component will not be discussed here because it is not part of I&B. The service management component is a large part of the I&B application and therefore will be discussed.

The connectivity layer is just the physical connection between telephones and has no other functionality than the simple routing of calls. The service agent layer adds more functionality to the network making it an Intelligent Network. The layer can be divided into the Service Switching Function and the Service Control Function. These functions can be located anywhere in the network and are highly centralized. A call that needs a specific service is now routed to a service switching point, which is connected to a service control point where the service is executed. The separation of the connectivity and service agent layer gives the operator a lot of flexibility to add or change services in his network. A service in the SCF consists of a number of scripts called SIBs, service independent blocks. The SCP is a runtime environment for these scripts. An operator only has to change the scripts to change a service. This is done via the service management component. The management component is also used for adding and changing subscribers and subscriptions of the service. The service creation component allows for easy creation of new IN-services.

4.2 CURRENT ARCHITECTURE

The current architecture of I&B is described in the following functional/logical view.
A. System Evolution Plan

Figure 2 functional overview

Figure 2 shows the functions that are involved or functions that implement parts of the I&B service. Other IN-functions that do not affect I&B are left out. The architecture of those can be found in their own SEP.

4.3 SCF AND SERVICE SCRIPTS

The service control function provides a runtime environment for the Service Scripts. These scripts define the actual functionality of the I&B service. They define how a call is handled. The SCF takes care of the communication between an instantiation of a service script and other functional entities like the SSF.

The I&B scripts are split into 8 groups. The scripts can be combined in several ways, which will be described in section 6.4. The possible feature scripts are shown in Table 1. By combining several features a specific services can be created.

<p>| 0 | Identification (ID), Statistics (STAT), Universal International Subscription Number (UIN) |</p>
<table>
<thead>
<tr>
<th></th>
<th>Feature Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Call Screening (CS), Authentication (AUTC), Emergency Routing Plan (ERP)</td>
</tr>
<tr>
<td>2</td>
<td>Ordinary Call Transfer (OCT), Time Dependent Routing (TDR), Origin Dependent Routing (ABR, EBR, RBR, TBR and LDR), Very Important Person (VIP), Call Distribution (CD), Originating User Prompter (OUP), Interactive Voice Response (IVR), Call Prompter Database (CPD)</td>
</tr>
<tr>
<td>3</td>
<td>Call Limiter (LIM)</td>
</tr>
<tr>
<td>4</td>
<td>Call Forwarding (CFC), Call Hold with Announcement (CHAQUE)</td>
</tr>
<tr>
<td>5</td>
<td>En Route Announcement (ERA)</td>
</tr>
<tr>
<td>6</td>
<td>Global Outgoing Call Screening (GOCSS)</td>
</tr>
<tr>
<td>7</td>
<td>Post Answer Redirection (PAR)</td>
</tr>
</tbody>
</table>

Table 1 IS&B features

4.4 GSA
The generic service adapter serves as an abstraction layer for SMS. It gives an open interface to SMS for other components.

4.5 SMS
The system management application server (SMS) provides an interface for other management components to the SCP and thus to the service. SMS has the possibility to change data direct in the SCP. This also includes the service scripts.

4.6 IS&B SMA
The IS&B SMA allows for the provisioning of the service, adding and changing subscriptions and subscribers. It is based on a 2-tier client server model and developed on SMABase, a framework that provides functionality like language support, authentication and session handling.

4.7 IWS
The Internet gateway webserver is a platform for Java/html based web-applications for IN-services. The ICC function is implemented on this platform.

4.8 ICC
ICC allows web-based provisioning for the end-user of the service (thus not the operators). It is a web-based counterpart of the old DTMF customer control. The use of the web allows for easier and more powerful customer control. A standard browser can be used to access this function which is now implemented as a Java-
applet. Although this function is currently implemented in the management component and not like DTMF in the service agent layer the implementation is different from the SMA. Therefore there is no guarantee that they behave the same or check the same constraints.

5

DEPLOYMENT

Deployment of the I&B service is showed in Figure 3.

The mapping of the different functions of the functional overview to the components is shown here. Furthermore the functions are mapped to specific Ericsson systems which implement those functions.

The management network has the following products: SMAS, SMABase, IWS and ICC. The GSA-server is incorporated in the SMAS product. Further are the I&B SMA client an server based on the SMABase framework.

In the service agent layer I&B uses the Service Control Point. This context point is connected to one or more Service Switching Points for the routing of the calls.
Figure 3 Product deployment view
5.1 INTERACTION

The interaction of the whole IN-network is based upon the request-response style. All the components in the network are generally passive; e.g., they start processing upon request not on their own. The requests are triggered by a user through one of the different the user-interfaces.

![Diagram of user interface interaction]

Figure 4 User interface interaction.

The user-interface in the link-layer is just a telephone or any other type of terminal. The user in this case is somebody who wants to make a call. An IN-call, a call that uses an IN-service is routed to a SSP. The SSP sends a connect-signal (CONN) to the SCP where the service script is started. The script will further handle the call (connecting to the right number, changing, playing announcements, etc).
Figure 5 Service agent layer interaction

The management part of the network has two user-interfaces, the 1&B SMA-client and ICC. A user triggers the interface, which then sends requests to other components. Through SMABase or IWS and GBA the requests are send to SMS where the information is processed and changes in the SCP can be made. The pipe-and-filter-style of this process can easily be seen from Figure 5 Management component interaction where the interaction from the SMA is showed.
5.2 NETWORK REQUIREMENTS

Today I&B is based on IN 3.0 and GSM R8 and uses the products shown in table 2.

The PAR-feature is a bit of an exception. This feature uses Ericsson INAP and therefore it can not be used in mobile networks. All the other features are supported both in fixed and mobile.

<table>
<thead>
<tr>
<th>Fixed</th>
<th>IN 3.0 (L7, TL4, TG6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>GSM R8 *</td>
</tr>
<tr>
<td>HLS</td>
<td>2.3 R2C</td>
</tr>
<tr>
<td>SMABase</td>
<td>3.1</td>
</tr>
</tbody>
</table>
6 COMPONENTS

Each of the components from the deployment view will be further described.

6.1 SMA BASE

SMA Base 3.1 is a set of tools and applications to give service providers a common platform for service provisioning and end user control of IN services. It consists of the two parts SMA Core and SMA Developer's Kit.

SMAs consist normally of a server part and a client part. The server part contains the core of the SMA, which is the same regardless of type of client, while the client part is specific for the type of client, e.g. a Graphical User Interface or a Machine-Machine Interface. With this partitioning, the same server part of a SMA can be used by different clients, thus providing consistent behaviour regardless of client access.

SMA Core also includes functionality that is common for all SMAs. It provides a uniform handling of vital core functions like the logon procedure, security handling and management of SMA users independent of the service.

The server part of the SMAs executes in an Application Server environment, the Sybase Enterprise Application Server (Jaguar) in combination with SAF. Their interfaces are described in IDL (Interface Description Language). Clients and servers can interact over CORBA IIOP, HTTP, CAI, and GSA-RPC. The GSA-RPC and CAI interfaces are implemented in SAF. The IIOP and http interfaces are implemented in the application server. This is showed in Figure 7 SMA Core. The service agent framework specific takes care of the routing of the requests where the application server executes the business logic and application.

I&B today uses the CORBA interface of SMARBase. The SMA client is a thick client made in powerbuilder. It has a bad design and is a point for improvement.
A. System Evolution Plan

Figure 7 SMA Core

SMA Developer's Kit is a development environment that provides tools and documentation for development of service specific Service Management Applications (SMAs) for deployment in SMA Core. Server side SMAs can be developed in either PowerBuilder or Java or a combination of both, while the SMA clients can be developed using PowerBuilder, Java or HTML techniques for web based clients. The SDK consists out of base-classes, which can be used. These classes can be inherited and methods can be overridden by an application. Figure 8 shows how the different components of the SMABase product are deployed into the server and client applications.

Figure 8 SMA client and server
6.2 SMS AND GSA

The SMA and ICC will send RPC requests to GSA-servers. These generic service adapters will transform the requests into SQL commands and will execute them on the SMS database. This can be done in two ways: synchronous and a-synchronous. In the a-synchronous case there is a queue which is polled at regular intervals.

6.3 IWS, ICC

IWS and ICC provide a web-based customer control interface. The ICC application runs on the IWS platform, which is communicating, to GSA via RPC.

In future releases of I&B, the functionality of the Internet gateway webserver (NW) will be integrated into SMABase (version 3.2). ICC will then be an application on the application server of SMABase. This will unify the management component and therefore ICC will have the same behaviour as the SMA.

6.4 SCP AND SERVICE SCRIPTS

I&B provides not only one service but is very flexible and provides a whole range of possible services. This flexibility is achieved by using little features, which serve as building blocks for those services. The features that exist in I&B are shown in Table 1. The making of a service now consists of combing the features through the feature selector script. The feature selector script defines the next feature to be executed and in this way features can be chained. Not all features can be chained together and therefore the features are categorized into 8 groups. Starting from basic features to more specific ones. The possible combinations are shown in Figure 9.
Figure 5 combining scripts

Most of the features are rarely used in practice. The most common combination is
the combination of ID, STAT and OCT. This combination is used for the popular
0800 and 0900 numbers.

6.5 SSP

The service switching point is not specific for IN. It’s used by all IN-services. It
routes the calls and takes care of the charging. In mobile networks the function is
mostly combined with the Mobile Switching Centre (MSC). The IN service uses
only the plain CS1 protocol to talk to the SSP and therefore there is no difference in
the service between mobile and fixed networks. Because both have CS1 implemented. The only exception is the PAR feature. This feature uses Ericsson INAP, which is more powerful and can handle three or more parties in a call what is needed for PAR. Because Ericsson INAP is not implemented in a mobile network this feature can not be used there.

7

CHANGES

This document is base on I&B 3.0, which is sold now. The requirement phase of I&B 3.1 is already started. This chapter will focus on the changes that are foreseen in version 3.1 and following versions.

7.1

VERSION 3.1

In the management component quite some changes are foreseen. Largely because an upgrade is done to SMABase 3.2 (v2). The new version of SMABase will have more functionality. A web-server will be included. This will phase out the IWS. ICC therefor has to be changed to use SMABase. Also the I&B SMA-client has to be changed from a thick client, which incorporates a lot of functionality to a thin client. This will ensure that the same constraints and checks are used both on an ICC and the usual SMA-client through the unified business logic in SMABase.

Furthermore the unified business logic will enable the development of an open interface for the operators to make a client of their own. This interface is called Machine Machine Interface (MMI) and will be based upon an IDL specification.

Figure 10 shows the new management network.

Figure 10 Desired management network

In the service agent layer no big changes in functionality are foreseen. More flexibility in combinations of features especially of group 2 features is desired. Also co-operation with number portability and possibly other services is desired. This
A. System Evolution Plan

should be done by chaining the service scripts of the different services together just like the feature scripts of I&B are chained.

7.2 LONG TERM

In the long term a switch has to be made to a new platform, the Jambala application platform to use the service in a 3G network. Because the Jambala platform is based on COTS components and Java the whole service functionality has to be redesigned. No details are yet known.

7.3 DOCUMENT CHANGES

This section will list all changes to this document so that changes are traceable. Future decisions that change the architecture will also be listed and explained why these are made. This will serve as a sort of knowledge base.

8 REVISION HISTORY

<table>
<thead>
<tr>
<th>Number</th>
<th>Responsible</th>
<th>Date</th>
<th>Revision</th>
<th>Affected section</th>
<th>Reason for update</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joris Bajens</td>
<td>2001-05-12</td>
<td>PA1</td>
<td>All</td>
<td>First draft</td>
</tr>
<tr>
<td>2</td>
<td>Joris Bajens</td>
<td>2001-06-28</td>
<td>PA2</td>
<td>All</td>
<td>Received comments</td>
</tr>
</tbody>
</table>

9 REFERENCES

SEP IN

SCSA Product Catalogue

Administration Function Specification 1/155 17 FAY 103 148

Implementation Proposal (IP) SMABASE V2 1/15941-FCP1010515 Uen

ELN/Y-01:067 Uen
Appendix B

Latency experiments

This appendix shows the experiments held in order to be able to quantify the performance and latency of two alternatives.

B.1 First alternative

The first alternative and the approach of the current architecture is to store all the data first in a MySQL database. An application-server polls the database at a specific interval for the data (Figure B.1).

![Diagram of data flow](image)

**Figure B.1:** The data-flow in the current architecture

The prototype consisted of three parts. First a client-application which inserts data in the database, the database and an application which queries the database. The database was deployed on another server than the two other applications. The measurements consisted of a number of clients which try to insert 100.000 records each as fast as possible in the database. Concurrent the other application queries the database a 1 second interval. In Table B.1 the measurements of \(T_2\) are shown. This is the time needed for the insertion of all the records in the database. The query-time of the database, \(T_3\) is dependent on the amount of records in the database and the amount of servers which are concurrently inserting records. When there are no other servers inserting records and with 1.000.000 records in the database, the application has to wait 3.9 seconds for the query to return. When there are concurrently servers inserting records the query-time increases to almost twice the time.
B. Latency experiments

<table>
<thead>
<tr>
<th># clients</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>47 sec.</td>
</tr>
<tr>
<td>5</td>
<td>2m. 50 sec.</td>
</tr>
<tr>
<td>10</td>
<td>5m. 48 sec.</td>
</tr>
</tbody>
</table>

Table B.1: Time measurements of $T_2$

B.2 Second alternative

The second alternative makes a direct link between the application-server and the return-servers as shown in Figure B.2.

![Figure B.2: The data-flow in the alternative architecture](image)

The database in between is gone away. The data can afterwards be inserted in the database from the application-server for persistency. The experiment was held with two applications, a client application which sends data to the application-server and an application-server which receives data from its clients. The two applications were deployed on two different servers. The application-server did two things, calculate some statistics, the same as in the first alternative, and writing the data to a file so that afterwards the data can be inserted in the database. The clients do the same thing as in the first alternative, they try to send 100,000 records each as fast as possible. In Table B.2 the measurements of $T'_2$ are shown. As can be seen, the application-server has the date much faster than in the first alternative. This will bring the loop-latency down quite a bit.

<table>
<thead>
<tr>
<th># clients</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 sec.</td>
</tr>
<tr>
<td>5</td>
<td>10 sec.</td>
</tr>
<tr>
<td>10</td>
<td>21 sec.</td>
</tr>
</tbody>
</table>

Table B.2: Time measurements of $T'_2$

B.2.1 setting

The tests were run in the following setting.
### B.2. Second alternative

<table>
<thead>
<tr>
<th>Machine 1:</th>
<th>Machine 2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>600Mhz intel celeron processor</td>
<td>700Mhz intel celeron processor</td>
</tr>
<tr>
<td>128MB RAM</td>
<td>128MB RAM</td>
</tr>
<tr>
<td>10 Mbit Lan</td>
<td>10 Mbit Lan</td>
</tr>
<tr>
<td>Linux Mandrake 8.0 (2.4.3-20 kernel)</td>
<td>Linux RedHat 7.1 (2.4.9-31 kernel)</td>
</tr>
<tr>
<td>MySQL 3.23.36 database</td>
<td></td>
</tr>
</tbody>
</table>

The setting is given such that the figures can be reproduced.
B. Latency experiments
Appendix C

TCP/UDP experiments

This appendix shows the experiments held in order to quantify the performance difference between TCP and UDP.

C.1 TCP

In the current architecture TCP-connections are used for communication between set-top boxes and return-servers. In a pilot project a Windows-server could handle a maximum of 2000 connections. This means that many return-servers are needed to support a larger group of set-top boxes. Also additional load-balancing and fail-over solutions are needed. The costs of this solution will be very high. In order to see if the performance can be improved we tested the amount of connections a Linux-server, which is expected to have a more efficient network-implementation, can handle.

For the tests a simple server-application is written. It opens a socket and listens for incoming connections. New connections are accepted and messages can arrive, which are then written to a file. The client application just tries to make as much connections as possible. This application runs on more machines.

A standard Linux configuration has to be tuned for such a specific use, else the performance of the Windows machine cannot even be achieved. The experiments were held with the values as shown in table C.1. These values showed good results, but with more elaborated tuning better results may be possible. The Linux-server was the same server used in the other tests as described in Appendix B.2.1.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximum open files</td>
<td>16384</td>
</tr>
<tr>
<td>stack size</td>
<td>16384</td>
</tr>
<tr>
<td>tcp max syn backlog</td>
<td>512</td>
</tr>
</tbody>
</table>

Table C.1: Configured values used for better network performance

The tests show that the Linux server is capable of handling more than 10,000 connections. This with the use of a low-end machine. With more
C. TCP/UDP experiments

memory, a faster processor and more tuning the results can be even much better. These results shows that the Windows-implementation does not perform well and that the use of TCP should not be as expensive as expected. A better implementation can change the performance and related costs considerable.

To test the performance of the setup, also tests were run which tried to send messages as fast as possible. For each message a different connection was made, which was also immediately closed after sending. A client was able to send 10,000 messages within 20 seconds. When more clients were started simultaneous, the time to send the same amount of messages increased to 23 seconds. The clients did not send the messages equally fast for an unclear reason.

C.2 UDP

UDP is a connectionless network protocol. This gives the advantage over TCP that the return-servers do not have the overhead of establishing and managing the connections. A disadvantage is the reliability, UDP does not guarantee the arrival of packets, which TCP does.

In the test the performance of UDP is measured. The tests were executed on the same local network with the same Linux machines as the latency tests, see Appendix B.2.1 for details. For the tests a tool called Iperf [Iperf] was used. This is a TCP/UDP bandwidth measurement tool. The tool sends in this case UDP packets to the other server and calculates how fast this can be done. The tool is used with the options shown in Table C.2.

<table>
<thead>
<tr>
<th>options server-side:</th>
<th>iperf -s -u</th>
</tr>
</thead>
<tbody>
<tr>
<td>options client-side:</td>
<td>iperf -c &lt;IP-number&gt; -u -b 10M</td>
</tr>
</tbody>
</table>

Table C.2: Iperf options used for measuring

The tool measured that with the simple LAN-connection a bandwidth of 9.1 Mbits/sec was achieved. This means that with UDP the network connection can be fully utilized. The processor was not even heavily loaded. This shows that one return-server can handle much more set-top boxes than with TCP-connections. The possible loss of packets is not problematic because packets are only dropped when there is not enough bandwidth at each node. If TCP was used in the same case packets would have to be resent many times, which results in a high latency and possibly the dropping of connections.