Mobility and Intelligence in Telecom: How and Where to Handle it?

Lill Kristiansen

Dept. of Telematics, NTNU,
Norwegian University of Science and Technology
O.S. Bragstads Plass 2A, NO-7491Trondheim, Norway
lill.kristiansen@item.ntnu.no http://www.item.ntnu.no/~lillk

Abstract. We will look into handling of mobility and intelligence in new mobile systems. Our aim is to build telecom system(s) enabling many types of mobility, as well as many different types of 'rich' or 'intelligent' applications. We will look into 'beyond 3G systems' based on middleware. We will look back to the design of the 3GPP system UMTS IMS. We analyze different choices and solution in IMS with an eye to the needed *co-operation* (standardization) vs. the possibilities for *competitiveness.* I.e., we investigate the possibility to deploy services that shall not be standardized. Based on this we introduce some new refined mobility definitions aimed at middleware based systems. We also identify some further research issues to be solved before middleware can be fully introduced in new 'intelligent' telecom systems.

1 Introduction

Several research projects aiming at 'beyond 3G' plan to use middleware or some sort of platform for mobile code. We find it relevant to revisit TINA, but this time with a close eye also on the real time requirements from the telecom domain. TINA's slogan was: "A co-operative solution for a competitive world" [1]. This aim is still valid. Competition between operators and between operators and content provider etc. are becoming more important. We also revisit the design of the UMTS IMS system [2], because that design had some focus on competitive services.

TINA was based on the use of modern middleware platform. Based on Open Distributed Processing (ODP) TINA defined a Distributed Processing Environment (DPE). A well known ODP compliant architecture is CORBA, Common Object Request Broker Architecture, with its Object Request Broker (ORB) implementation TINA service architecture [3] introduced the notion of BAD (Business Administrative Domain) and related this to the DPE.

The paper is organized as follows: First we look at some trends relating to convergence and we look carefully into some mobility and middleware definitions in the context of this convergence. Then we 'make a step back' looking into the design of UMTS IMS (from around 1999). We identify important issues discussed during the design and standardization of that system. We then return to middleware, looking into

TINA, in the light of IMS. At the end we introduce some refined mobility definitions for middleware, and identify some related issues for further research in order to use middleware 'all over' in a modern telecom system.

2 Convergence

With convergence and the introduction of IP into telecom systems we also introduce some new separations. This means: Things that we former thought of as one thing, has now became several different entities. The user and terminal are now separated[1].We may illustrate the old systems and new convergent system as in Figure 1.

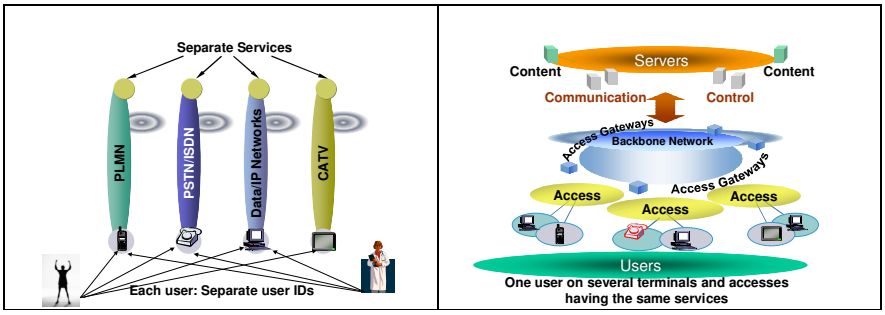


Figure 1 Evolution: From several monolithic systems (left) towards an access agnostic system with user mobility. The left part illustrates *today's situation* where the user has several different identities, and several rather similar services. The similar services may be either *messaging service* with its variants (Email, SMS, Instant Messaging (IM),…) or *voice telephony service* with its variants: (GSM voice telephony with call forwarding to mobile-voicemail, PSTN voice telephony with fixed-line-voicemail etc). The right part illustrates a *new situation*, where user and terminal is separated, and the same services may be accessed from several devices with the same user id. Figure is modified from Ericsson, see e.g. [4]

Another aspect of convergence is the combination of GSM / GPRS / UMTS technology (with its continuous handover) with more discrete and nomadic services on PCs and PDAs via IP and WLAN (e.g. web, with email and chat etc.). We may note that discrete terminal mobility may be supported also in fixed networks. We may also note that the separation of *discrete* and *continuous* terminal mobility corresponds to the well known separation of *call-related* and *non-call-related* (see more details in section 2.2).

[1] We have all seen drawings of the 'ultimate communication device' having clock, screen, loud speakers, printer etc. all build in. But specialized devices like phone, clock, etc. will still be around. One may also foresee that PCs will continue to exist as separate entity despite of devices like 'chat boards' attached to mobile phones and PDAs. Thus we will still have a multitude of terminal types, and it is important to handle that one user can use several of these devices, inside one common system.

2.1 The new Separation of User and Terminal

When we are separating the user and the terminal, we need to take care then we talk about terminal mobility and user mobility. We will return to this in section 3.2.

IMS is 'access antagonistic', and this means that also fixed accesses should fit in. See Figure 1 again. We will list some issues that must be looked into when designing such a system:

Where/on which terminal(s) do you want your call to be delivered? How/where to handle other special call treatment? End-point or network? What if the only available terminal and network is a 2G network (e.g. PSTN)? How to have access to your address book and more at all times and from all devices? How to treat hand-over and call transfer between different networks which may vary in both network capabilities and costs? Should A or B pay for B's mobility? [2]

Some of this issues are mostly related to the fact that one user is moving between several terminals, while others are mostly dealing with the relations between the home network domain, the visited (or access) network (and the endpoint). We will return to some of these questions later (e.g. in section 4).

2.2 Some Session Definitions Relating to Real Time

We have already indicated that log-on procedures to fixed line terminals should be possible within a future mobile system. Hence we have discrete and continuous mobility in our system. In telecom we traditionally have the following concepts:

- Call session and the so-called *call related services***.** This is related to the normal telecom concept of 'soft real time'. A typical example is execution of call forwarding or call transfer. (Handover of call falls into the same category)
- Registration procedures and other *non-call-related services*. This is related to an interactive process involving the user and will have some timing requirement. Turning on the mobile and activating call forwarding are typical examples
- For the sake of completeness we should also mention *management services*. These services may be interactive or may even be run as a batch job.

TINA however ignored these differences in their modeling, and did not separate session data from persistent data. This might have caused some problems when implementing a services like 'multimedia call' in a real time and scalable way.

3 Some Definitions Relating to Mobility and Middleware

We must be careful with our definitions. They may come from several sources, and implicitly have different contexts and assumptions surrounding them.

---

[2] In GSM the caller (A) pays for a national mobile call (i.e. A pays for some of B's mobility). This is possible because A knows that he calls a mobile number with a specific tariff. B pays for the roaming leg in case he is moved to abroad, since this is unknown to caller A. In US mobile systems, no separate number range was allocated for mobile calls, so in all cases the called party B paid. This resulted in a situation where many customers did not give out the mobile number, because of fear for the extra cost. This was partly a reason for the slow take up of mobile systems in the US. Note also that fixed telephony has different charging models in Europe and US, and this influence usage patterns.

There are many notions of middleware. The GSM system may be seen as supporting location and migration transparencies to the applications in the GSM terminals. This view might work for some distributed applications, (typically those using socket programming). However, for building new distributed applications in general we may need to look more closely into the call setup procedures (sometimes called session control).

3.1 Initial Remark: Data and Code

In a von Neumann architecture we separate data and code. However, data may in fact be code. The 'classical example' is a Universal Turing machine taking a description of another Turing machine as input. A less abstract example is a compiler taking code as input.

We still find it useful to separate between *code* and *data*, not the least for performance reasons. Thus we may have *.exe* files and *.dat* files as well as (small or big) (control) data being communicated ('moved') over the network and we find it sometimes useful to distinguish them.

Code and *data* are just two ways of looking at the same thing. A relevant analogy is in physics where we can consider light either as *waves* or as *particles*. We can chose the one or the other approach depending on what properties we are aiming at. Reasoning about real time and performance are best done when code and data are seen as different entities.

However, other properties may easiest be deduced by playing with the 'code=data' concept. E.g. some properties of service interaction problems can be shown to be unsolvable by using these techniques and relating it to the halting problem results of Universal Turing machines. (See Gaarder and Audestad [5]).

3.2 Mobility Definitions

We start this section with definitions of terminal mobility and user mobility in a way that support the separations described in section 2.1.

Terminal Mobility is the ability of a terminal to change physical location. This includes terminals which can continue to support services while moving, and those that cannot. (From TINA [6])

Similar definitions are given in standardization documents describing currently existing mobile systems. The process of changing network access point is called a handover (or handoff) in the continuously case.

User mobility is defined as the ability for a user to connect to, or use, different terminals or terminal types for the purpose of communication. (From Ericsson [7])

This definition does not specify exactly the phrase 'for the sake of communication', nor does it specify exactly what communication services we are talking about. The next definition talks about all services (subscribed to by the user), and is hence somewhat more specific, as well as more difficult to fulfill.

Personal Mobility is the ability of a user to access [all[3]] services from any terminal and any location, (including invitations[4] to join sessions). This ability may be

---

[3] All is added here for clarity.

[4] 'Invitation to join sessions' are the TINA term corresponding to SIP INVITE (or a call setup)

restricted due to contract agreements between the Consumer and Retailer, and due to user system capabilities. (From TINA [6])

This definition is pretty close to 3GPPs definition of Virtual Home Environment (VHE) defined in 3GPP [8]. It is also what we illustrate in Figure 1 (right part).

Both these TINA definitions and the VHE definition of 3GPP are definitions of abstract concepts, not saying anything about how they should be implemented. (In fact TINA and 3GPP chose to implement them differently).

3.3 Agent Definitions

From Nwana [9] we have the following definitions:

- Agents may be classified by their mobility, i.e. by their ability to move around some network. This yields the classes of *static agent* or *mobile agent*.
- Secondly, they may be classed as either *deliberative agent* or *reactive agent.* Deliberative agents derive from the deliberative thinking paradigm: the agents possess an internal symbolic, reasoning model and they engage in planning and negotiation in order to achieve coordination with other agents.

We may also note that so far nothing is said about the communication aspects of these agents. In the light of the remarks in 3.1 we see that the definition of *static* and *mobile* may depend on how you choose to look at the system. Objects communicating via data exchange, may sometimes be seen as moving objects between them.

Much work on agents is based on deliberations and artificial intelligence (AI) technology. We will not go into that here. We will however illustrate some technology for mobile agents that do *not* use AI technology, as well as one well known static approach.

3.3.1 TACOMA Mobile Agents

We quote from [10]: An *agent* in TACOMA is a piece of code that can be installed and executed on a remote computer. Such an agent may explicitly migrate to other hosts in the network during execution. The TACOMA project focuses on operating system support for agents and how agents can be used to solve problems traditionally addressed by other distributed computing paradigms, e.g. the client/server model.

The TACOMA concept was used in the StormCast project [11] for reporting e.g. weather conditions from the fishing boats in the Artic Sea via satellite to the mainland Norway. Due to bad satellite links, (especially in bad weather, when the most needed), the code was sent from the mainland, and executed locally, and then the results were return to the mainland. We will return briefly to StormCast in Figure 7.3).

3.3.2 SDL processes as Static, Reactive Agents

It is a long tradition in telecom for modeling in SDL. SDL uses *static reactive agents* according to the definitions above. We may note that in telecom we are almost always using the asynchronous communication scheme. We may also note that static agents can implement a mobile system. (GSM is a prime example).

3.4 ODP and Middleware Definitions

A simple picture of an ORB and the CORBA concepts is given in Figure 2.1). The objects at the top are ‘mobile’, e.g. they can be accessed independently of their

physical location. The object services, common facilities and the ORB itself will locate the object and perform the binding. Then the communication between the objects can take place. A TINA view of the DPE is given in Figure 2.2).

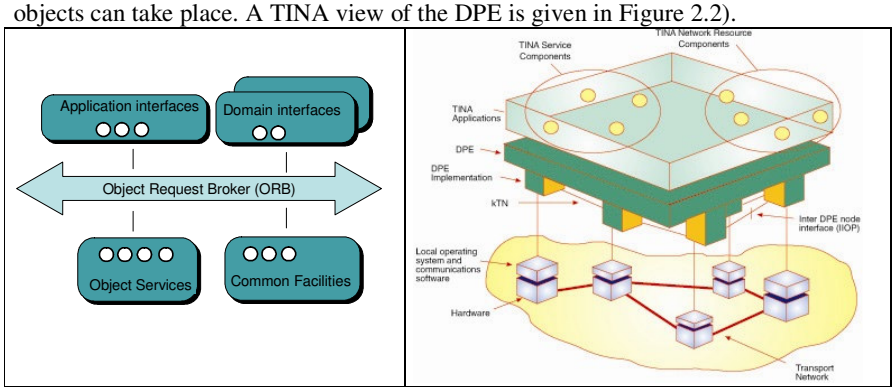


Figure 2 Illustration of two different views of midleware. From a computer science view the network is often not shown (left). Quite naturally coming from the telecom side, TINA shows the networks: Transport Network (TN) and kernel TN (kTN). 2.1) Left: OMG-like from [12]. 2.2) Right: TINA-like, see e.g. [13]

We have the following important ODP transparencies. They will be offered by the middleware platform (the ORB/DPE and below).

Table 1 Some important ODP transparencies (from [14])

Access transparency	Hides the heterogeneity of the implementation language (by using Interface Definition Language IDL)
Location transparency	Hides the details of the location (node) of one object from the communicating other objects
Migration transparency	Hides the details of a migration function from the application objects

CORBA offers object services as well, such as Life cycle service, Naming service, Trading service, Notification service, etc.

There is a separation between migration and relocation in ODP. We will quote from Gavras [14]: 'The relocation function facilitates an uninterrupted client-server interaction during system changes'. Whether 'uninterrupted client-service interaction' means for real time in a telecom sense or 'just' for RPC over TCP is a bit unclear. Migration need not offer uninterrupted service.

We see that CORBA separates 2 types of timing constraints. They most probably correspond to the distinction between *non-call-related* and *management* in 2.2. Call related services (with stronger real time requirements) are most likely not covered by the ODP relocation concept.

4 Lessons Learnt from Designing the IMS System

Consider the question *'Where to place the 'intelligence'?* First we must define what we mean by 'intelligence'. In this paper we do not mean 'artificial intelligence', rather plain, simple functionality, like a call forwarding service, or a no-reply-functionality. We do however foresee new variants of call forwarding, or no-reply, using more complex information.

The old call forwarding services are simple in two ways: The treat all callers the same way, and they are based no simple call states such as:

- CFU Call Forwarding Unconditional
- CFBS Call Forwarding Busy Subscriber
- CFNR Call Forwarding No Reply

Now we foresee that presence and status information from an IM system may be used, as well as information from a calendar service. Some new possibilities of such services are given in [15].

We want to build new 'advanced' services. We are now ready to revisit the question *'Where to place the 'intelligence'?'*. We will do so by looking into related issues in IMS and how they were solved.

4.1 Issues in the Design of IMS

In IETF, SIP [16] is chosen as the standard for session initiation (and hence for call setup in multi media IP telephony). 3GPP has chosen SIP as well for their IMS system [2], but with some deviations in the system architecture and the business model. IMS is an access antagonistic system aiming at a convergent system, as illustrated in the right part of Figure 1.

4.1.1 Real Time and Availability Issues

The telecom companies seems to prefer a layered approach to services, i.e. to build APIs from the call layer to a separate service layer. This is advocated in 3GPP [8] . IETF advocated an other approach with extensions, some of that history is documented on the web page [17].

One reason why the telecom companies wants an API might be that in this way the call layer may be more stable, and less affected by new features and value added services. Some real time protocols may be reused, (as in IN). Business aspects may affect the choice as well. We will not study this in more detail in this paper.

IMS is not based on mobile code. This again has something to do with the telecom requirements. It is simply much harder to guarantee the real time aspects if new foreign code starts running on your machine! It will also be much harder to guarantee 'the 5 nines' of service operation. Today we expect to be able to pick up the phone to the IT department when the computer network is down. This tell us something about the wanted availability of a modern phone system (also when it is based on IP!).

We may note that IMS is in some aspects a 'traditional' telecom system. It is based on proven technology (no mobile code), as well as the use of 'call flows'(MSCs). SDL may well be used in the internal design (though that is not formalized).

4.1.2 Issues Relating to Competition and Standardization

In the design of the IMS system some major questions were: I. Should the intelligence be in the endpoint or in the network? II. Should the intelligence be in the visited domain or in the home domain? One important issue before determining solutions to these questions was: III. What will enable us to deliver new advanced services to the market in a fast way, and at the same time obtain interoperability as well as possibilities for new competitive services?

The answer to III in UMTS IMS was: Keep the Serving call server (S-CSCF or S-CS for short) in the home domain. This allow for a 'minimal' need of standardizing the basic capabilities and features in a call setup. The wish was to avoid the standardization of value added services. Instead each domains can develop non-standard services via e.g. OSA or a separate service layer.

In such a traditional telecom system without mobile code we may place the functional entities ('boxes') in a matrix according to the layer and the domain they are in. Each of the 12 'boxes' are both software and hardware run by the same domain. This principle is illustrated by Figure 3:

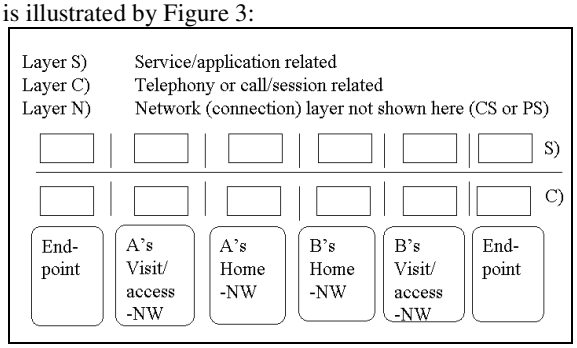


Figure 3 Reference figure. Such a figure may be used to analyze business relations in a system. It can also be used to compare the architecture of GSM and IMS. (For more details, see [18])

Here we placeIMS according to this reference figure.

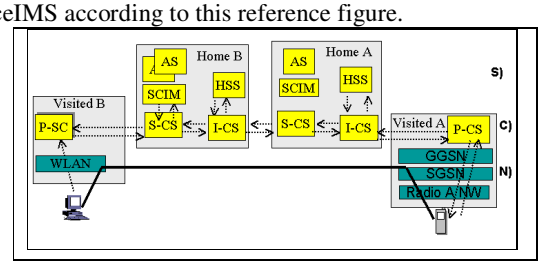


Figure 4 IMS placed according to Figure 3. For a similar figure of GSM and OSA, see [18]

4.2 Relating the IMS Architecture to its 'Predecessor' IPT from Ericsson

Ericsson introduced some refinements of the H.323 system. These refinements were implemented in IPT II product in 1999-2000. Some of them were adopted as a formal standard (H.323 Annex K [19]). Others were proposed to ETSI Tiphon [20]

but not accepted at that time (spring 1999). We may however say that even though the splitting was not accepted by Tiphon, the architecture and thinking behind the proposal was in the end accepted by 3GPP. (See also Ericsson press release [21]). The similarity with the 3GPP architecture is explained below:

Figure 5 shows the H.323 compliant system IPT II from Ericsson. We see that the gatekeeper (GK) has been split in 2 parts, the N-GK and the U-GK. This was in accordance with the proposal [20]. These two gatekeeper components correspond roughly to the IMS Call State Control Functions in IMS depicted in Figure 4. The correspondence is as follows:

- N-GK corresponds roughly to Proxy-CSFC in IMS
- U-GK corresponds roughly to Serving-CSCF in IMS (and partly to HSS)

In the 'classical' case of a PC to PSTN call, the called user (on the plain phone) would be treated by the PSTN system only. Here this is not the case. Instead the terminating half call is handled by the new IPT system, and the called party is an IPT-user. The IPT system supports the user mobility in an access antagonistic way, just like IMS. Figure 5 shows how a PSTN user (B) will execute the new web-based services, and deliver the information to the calling user (A).

Since the access is this case is fixed, continuous terminal mobility will not be supported. However, if user B later registers on a wireless terminal (WLAN or UTRAN access), then both terminal and user mobility will be supported.

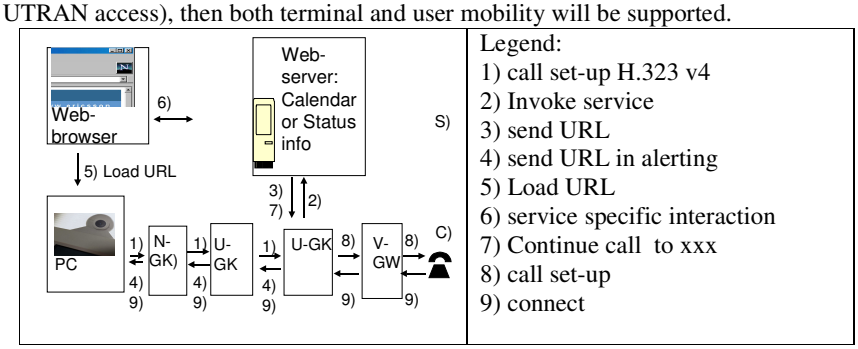


Figure 5 Multimedia features in a Multi Media over IP system (based on Ericsson [22]). The entities are placed in accordance with Figure 3

As in IMS the network centric solution with a 'user representative' in the home network enables personal mobility as well. The 'user representative' (S-CSCF and U-GK respectively) corresponds roughly to TINA's User Agent. In IPT as well as in IMS the main reason to place this 'user representative' in the home domain, was that the visiting domain(s) may be competitor(s) of the home domain.

4.3 Concluding Remarks on the IMS System

We may notice that the system architecture in IMS is different from the GSM system architecture. Figure 3 is useful when doing this comparison. In GSM the user profile (treating the supplementary services) is moved (as data) during registration from the home domain to the VLR/MSC in the visiting domain. Also the Camel API in GSM is different from the API approach proposed in IMS, as explained in [18].

When GSM was designed there was not much competition between the operators, in fact each operator was mainly national. We may notice that today the GSM operators are competitors. E.g. today Telia (Sweden) owns the 2nd largest GSM operator in Norway. This user profile transfer may thus reveal important user information to a competitor.

The solution in GSM requires that all the supplementary services (like CFU, CFNR etc.) are *fully standardized.* In that way the VLR / MSC would be able to treat the user profile data, and use it during its own execution of its (static) code. (Look back to section 3.1 for more on mobile data versus mobile code)

We will now return to TINA service architecture and middleware. Based on the experiences from IMS, we will revisit TINA and have an even closer look at TINA's slogan: "A co-operative solution for a competitive world". In particular we will look at business administrative domains. They will act as competitors. They will also use middleware from competing vendors. These middleware platform must co-operate, but they may also add extra functionality.

5 TINA Service Architecture

We will first give some basics of TINA. TINA Service Architecture was developed in several versions, the latest being [3] from 1997. TINA Consortium was a combination of telecom and computer (IT) companies. TINA was based on the use of a Distributed Processing Environment (DPE) and heavily influenced by ODP / CORBA.

In Figure 2.2) we see a middleware platform where all objects 'float' freely around at the top. However, in the TINA Service Architecture we illustrate which domain each of the objects belongs. In Figure 6 we see a typical example of domain boarders.

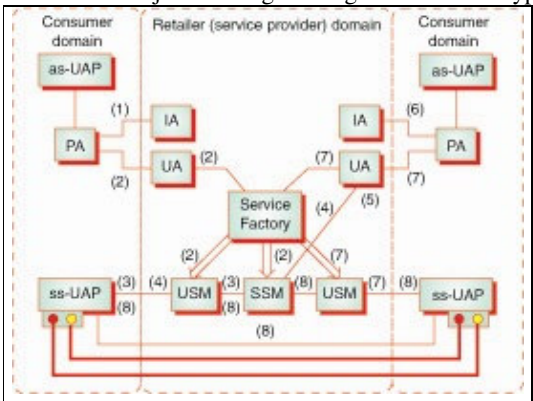


Figure 6 TINA service session initiation with domains and multimedia streams shown. Figure from [12]. The numbers are referring to method invocations explained in TINA [3]

TINA chose to build the service session control in the applications *instead* of using the build in ORB and the common facilities. In this way we *may* obtain more control of the real time aspects, and make it easier to build call-related value added services.

However TINA did not separate between call related services and management services. TINA did not separate between 'states' and persistent objects at the modeling level neither.

In general we may say that still today there are few telecom properties in most ORBs. Inside one domain (or one server farm), we may however use a telecom-enabled version of an ORB, such as e.g. TelORB [23]. But we are still far away from a situation where we have ORBs 'all over'.

In Figure 6 the domain boarders are illustrated at the (application) object level. However, the details of how these objects maps to nodes are not shown, but it is obvious that the consumer domain is at the terminal (endpoint).

As we see in Figure 6 TINA did not have a separate access provider. In order to support personal mobility (virtual home environment) it would be natural to place the User Agent in the home domain. But there was something missing in the TINA definitions: The end user was supposed to log on to his home domain User Agent. There were some discussions on how this binding should take place ([24]), since it actually involved the DPE of 2 (or 3) domains. See also Kristiansen [4], for some more on this issue. We may, however, note that several research projects working with TINA and mobility in the 90'ties actually introduced an 'access network' is some form, (see e.g. Thanh [25]). For newer information (2004) on CORBA and mobile endpoints, access network and home network, see OMG [26].

Today we have global operators (e.g. of the size of Vodafone). We can imagine one such operator which we call *worldwideltel.net* that offer wireless and fixed accesses. We may have the case that *Worldwideltel.net* is present both in the subscriber's home country, and in the visiting country. The User Agent (UA) may thus be migrated to a node in the visited country, but still be in the same company.

In fact maybe *worldwideltel.net* in UK and *worldwideltel.net* in US are (legally) separate entities, but still with much cooperation between them. (Much more cooperation than between real competitors). We may e.g. assume that they have the middleware in common and use their own internal network between their ORB nodes. To move objects between nodes is part of the job of an ORB. In this case we may see this as being done within one business administrative domain (BAD).

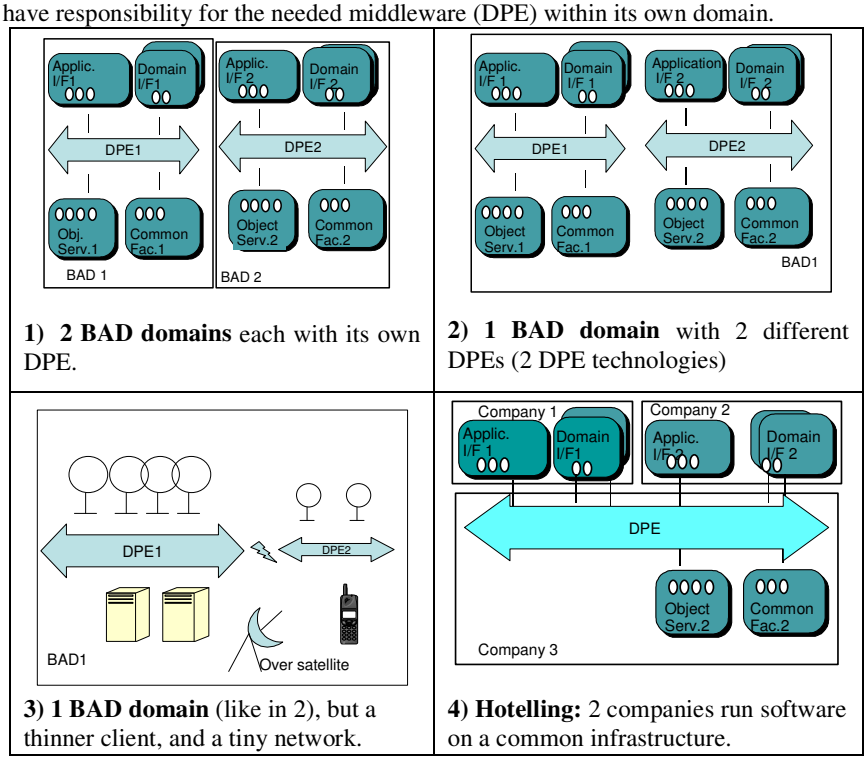
Another question is if it really matters to move the UA to a more local node. In a global world distance is of less importance, though is may of course reduce the latency somewhat to use a local node. We may note that in any case this can be one during registration phase (non-call related). To move objects during call setup phase is far more demanding, as indicated in section 2.2.

When we move object between domains we must take care. In particular, mobility between the endpoint and network needs a special type of care. This is because the sending of data over e.g. GPRS (even best effort) is not for free. Thus it is important to control when data are sent locally between objects (applications) inside the node (terminal) and when data is sent over the (billable) network.

6 Refinements of the DPE: New Types of Mobility

In this section we will use DPE instead of ORB, since ORB is often related more directly to existing CORBA compliant products.

TINA in 1997 introduced *business administrative domains* (here we use *BAD* for short). Though not formally defined in TINA [3], it was assumed that each BAD will have responsibility for the needed middleware (DPE) within its own domain.



1) 2 BAD domains each with its own DPE.

2) 1 BAD domain with 2 different DPEs (2 DPE technologies)

3) 1 BAD domain (like in 2), but a thinner client, and a tiny network.

4) Hotelling: 2 companies run software on a common infrastructure.

Figure 7 Refinements taking business domains into account

From TINA [3] we quote: 'The overall objective of the service architecture is to support the most general case of business administrative domains, interacting with one another over a DPE, in order to offer business objects or applications for commercial gain'. We may note that the term 'a DPE' in this quote makes this statement somewhat unclear.

The intention was that each BAD should have software objects running on a DPE that suited their own needs. Then we need to distinguish between the following meanings:

- 'A DPE' meaning 'one common DPE used by all BADs.'
- 'Several DPEs' meaning 'each BAD choosing a DPE that suits their own requirements and with some basic interworking between the DPEs to ensure interoperability).'

TINA chose the interpretation given in second bullet. We may notice that also within one BAD there may be a need for several types of DPE, some supporting high availability and real time, and others that do so to a lesser degree. This is illustrated in

Figure 7.2) Between the domains we will need protocols and Service Level Agreements (SLAs).

Figure 7.3) may be an illustration of the StormCast project as described briefly in section 3.2. In this case the server on the mainland and the satellite phones may be regarded as belonging to *one* domain[5]. Hence we call this concept for IntraDomain mobility, since the mobility is only internal to one (business) domain. In Figure 7.2) and Figure 7.3) there is one business domain, and 2 DPE technology domains.

If we look into mobility across these domain boarders a refinement will be needed. Then we will have two domains like in Figure 7.1), but the mobility will now be across the business domains. We will call this case InterDomain mobility.

Figure 7.4) illustrates a case where Company 3 acts as a *hotel host* or *infrastructure provider* for Company1. The TINA concept of BAD is not valid any more because we are breaking the assumption of relations between DPE and BAD. To handle this case we will need even more refined definitions. The case of course gets further complicated by the fact that Company1 and Company2 share the same DPE. To formulate Service Level Agreements (SLAs) in this complex setting is for further study.

We will now introduce several refined notions of mobility support of objects (or code), i.e. extending beyond the definitions of the transparencies in section 3.4 and Table 1. We will first look into the 'vertical' approach illustrated in Figure 7.1). This corresponds to TINA's notion of BAD in the simplest case. This figure could also be refined with the notion of kTN provider and TN provider (i.e. like in Figure 2.2).The new definitions will now be given. They all assume that each DPE is located inside one BAD, as explained earlier.

IntraDPE mobility (DPE mobility): Mobility of objects between nodes in one DPE domain. This definition is illustrated in Figure 7.1) and Figure 7.2). In this case the mobility of the objects are within DPE1 or within DPE2, but not across the DPE boarders. Implicitly this means that the objects stay inside one BAD.

IntraDomain mobility (**IntraBAD mobility**): Mobility of objects (between possibly different DPEs) within the same BAD. This definition is illustrated in Figure 7.2) and Figure 7.3). To support such mobility, the concept of *design portability* from TINA [6] may be needed.

InterDomain mobility (**InterBAD mobility**): Mobility of objects between business administrative domains. This corresponds to Figure 7.1) when we allow full mobility. The issue of 'foreign code' will now pop up. (See 4.1.1)

We may also refine these 3 definitions with the real time concepts from section 2.2. Thus we may have objects that are mobile between different DPEs in the same BAD domain, but not in real time. This may e.g. correspond to a *planned* software upgrade or a planned migration of code, and this is different from handling the mobility in real time. This has similarities with the two ODP concepts migration and relocation. However, as we saw in 3.4, the ODP concepts may not distinguish between call related services demanding real time, and other services requiring some 'reasonable' response times.

---

[5] In a general telecom setting we will regard the endpoint as a separate domain, like in Figure 6.

7 Identification of Future Research Issues

We have seen that many of the requirements in telecom related to real time. A CORBA based middleware may be OK to use inside one domain. Here products like TelORB [23] is relevant. Across domains there are still some problems with such platform concepts and real time applications.

We will now look back to Figure 2 and Figure 7 and revisit the question: *Where should we place the ('intelligent') functionality?* In this case the question refines to: In the platform? In the DPE kernel? As common facilities? As object services? Or should we make the intelligence explicitly as a distributed application?

We will not answer these questions in detail here. We may however conclude briefly as follows: The more 'intelligence' or functionality we place down in the platform, the more we need to standardize (and the more time it takes).

There are also several other aspects to the questions above. To place something inside the DPE or below the DPE may be seen as a way of choosing the level of abstraction. Will this require that the domains agree on a common abstraction of their modeling? This question is again related to the design portability definition from TINA [6].

Some service creation methodologies assume that the same (design) methodology is used 'all over'. This is not realistic. Inside each domain the business itself should be able to use what they conceive as the best tools available to move from service creation, and down to running code. We need to allow for plain protocol interworking between domains.

8 Conclusion

We have studied the notions of platform and mobility. In particular we separate between sessions and objects that are mobile *in real time* (call related) and *not in real time*. We also separate objects being mobile *within* a business domain or *between* business domains. We separate object mobility *within* a DPE technology domain and *between* DPE technologies as well.

Questions like: *Where to place the intelligence?,* may have several answers. Based on the experienced from the design of the new multimedia IMS system in UMTS, we find it useful to consider this along two axes:

- A horizontal axis: from endpoint, via access network to core network.
- A vertical axis: from application layer to call layer to middleware layer.

We have given some new some definitions of mobility in a DPE setting. We have also identified some new research issues that needs to be answered.

It is important to analyze the need for standardization versus the need for competitive services and a quick time to market. It is also important to take the real time and other telecom requirements into account.

Acknowledgments

The author wants to thank several former co-workers in TINA core-team and in Ericsson (in Norway, Sweden, the Netherlands and elsewhere). Børge Nilsen then at Ericsson IPT (Norway) should be mentioned in particular.

References

[1] TINA Consortium web site: www.tinac.com (Accessed July 2004)
[2] 3GPP, IP Multimedia Subsystem (IMS); Stage 2 (release 5), TS 23.228,
[3] TINA, Service Architecture Version 5.0, Baseline document, (1997)
[4] Kristiansen, L.: A 'user centric' appraoch to multi media services, TINA Conference, Paris, (2000) http://www.tinac.com/conference/Paris2000_14.htm
[5] Gaarder, K. and Audestad, J. A.: Feature Interaction Policies and the undecidability of a General Feature Interaction Problem, TINA Workshop, L'Aquila, Italy, (1993)
[6] TINA-C Glossary of Terms Version: 2.0, (1997)
[7] Ericsson, L.M.: Mobility and roaming technologies, ETSI Tiphon contribution 12TD088, (1999) http://www.item.ntnu.no/~lillk/docs/12td088.doc
[8] 3GPP, Virtual Home Environment/Open Service Access, Doc. no. TS 23.127
[9] Nwana, H. S.: Software Agents: An Overview, Knowledge Engineering Review, Vol. 11, No 3, (1996) 1-40, http://agents.umbc.edu/introduction/ao/
[10] TACOMA, http://www.tacoma.cs.uit.no/overview.html (Accessed April 2004)
[11] StormCast http://www.cs.uit.no/forskning/DOS/StormCast/ (Accessed April 2004)
[12] Braek, R.: Middleware lecture, tttm4160, NTNU ((Accessed April 2004) http://www.item.ntnu.no/fag/ttm4160/Middleware/Middleware.pdf
[13] Handegaard, T. and Kristiansen, L.: The TINA Architecture, Telektronikk Vol. 94 No.1, ISSN 0085-7130, (1998), 95-106,
[14] Gavras, A.: Distributed Platforms for Telecom Applications, Telektronikk Vol.96, 4 ISSN 0085-7130, (2000) 137-145
[15] Bolstad, K.: Personalized options on no answer conditions; introducing HTTP based menus into IP multimedia telephony, M.Sc. thesis, Ifi, Univ. of Oslo, May 2001. Available http://www.item.ntnu.no/~lillk/docs/NoReply.pdf (Accessed July 2004)
[16] IETF Network Working Group, SIP: Session Initiation Protocol, RFC: 3261 http://www.ietf.org/rfc/rfc3261.txt
[17] Schulzrinne, H.: SIP Drafts: Extensions to Base Specification , web-page http://www.cs.columbia.edu/sip/drafts_base.html (Accessed July 2004)
[18] Kristiansen, L.: An Open Service Architecture With Location Aware Calls And Services, Proceeding of WOCN conference, Oman , (2004)
[19] ITU-T, H.323 Ver.4, Annex K
[20] L.M. Ericsson, Splitting the Gatekeeper, and some reasons to do so, ETSI Tiphon contribution 13TD104, (1999)
[21] L.M. Ericsson, IPT II press release, June 2000 http://www.ericsson.com/press/archive/2000Q2/20000607-0099.html
[22] Ericsson, L.M.: Services in H.323, Protocol Update, Presentation at VoN, Atlanta, Nov. 1999, http://www.item.ntnu.no/~lillk/presentations/Von-99-atlantawernerE.ppt
[23] Telorb, web page http://www.telorb.com (Accessed July 2004)
[24] TINA, Internal discussions on mobility with Hans Hegermann, core team member, 1996
[25] Thanh, D.V.: Mobility as an Open Distributed Processing transparency, Dr. Scient Thesis, Ifi, Univ.of Oslo, ISBN 82-7368-162-9, (1997)
[26] OMG, Wireless Access and Terminal Mobility in CORBA, Version 1.1, (2004)