Introduction to IMS application Development & Integration

White paper

By: IP-Solutions AB, 2008

Application servers execute and host services within IMS (IP Multimedia Subsystem). In order to integrate and develop application servers for IMS, there are some considerations that need to be taken into account, like the three main interfaces towards IMS: the Sh, Si and Isc interfaces. Depending on the type of service that the application server is hosting, it may run in two different modes: SIP UA (user agent) or B2BUA (back-to-back user agent). The application server may be located inside the home network or in a third-party network.

E-mail: info@ip-solutions.se, Tel: +46 8 615 08 60, web: www.ip-solutions.se

IMS Core functions

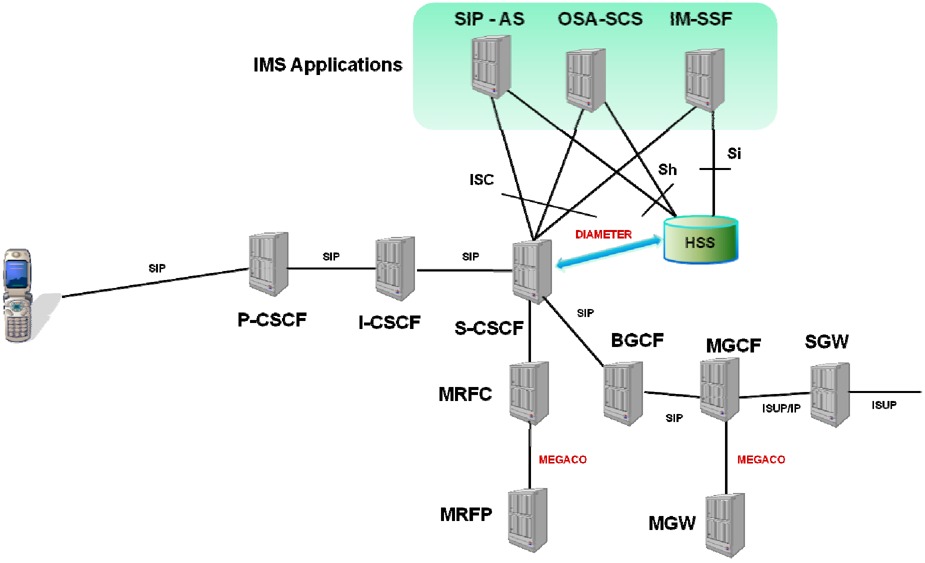

Figure 1 - The IMS core functions

The main functions in IMS consist of the P-CSCF, I-CSCF and S-CSCF. A CSCF (Call Session Control Function) processes the signalling in the IMS network.

P-CSCF (Proxy Call Session Control Function) is the *terminal's* entrance to the IMS network. It may be located in the home network or in the visited network in case of roaming. The terminal will discover the P-CSCF with either DHCP or PDP context activation (GPRS) or have a preconfigured P-CSCF address. The P-CSCF will not change during the lifetime of a registration.

I-CSCF (Interrogating Call Session Call Control Function) is placed at the edge of an IMS domain. The I-CSCF is the point of entry for (incoming) calls entering the network from another IMS domain, and its address is the one placed in the public DNS. The I-CSCF uses the DIAMETER based Cx interface (not shown in the figure) to communicate with the HSS.

S-CSCF (Serving Call Session Control Function) is the main function in IMS. It authorizes the user upon registration and will always be in the home network for any particular user. It uses the Cx interface to communicate with the HSS and DIAMETER is the protocol used. The S-CSCF authenticates the user and downloads (from the HSS) a **user profile** which includes information about different application servers used for the particular user. This is in fact one of its primary functions. The S-CSCF also enforces the policy of the network operator, i.e. it keeps track of which types of sessions that a user is authorized to initiate.

HSS (Home Subscriber server) is a central repository of data related to the subscribers in the system. HSS is the successor of the HLR, and it also incorporates the AUC, as it holds the shared secrets used for authenticating users. Information included in the HSS is:

- Public Id
- Private Id
- Service Profile
- Initial filter criteria
- Security information
- Assigned S-CSCF

Simple call flow

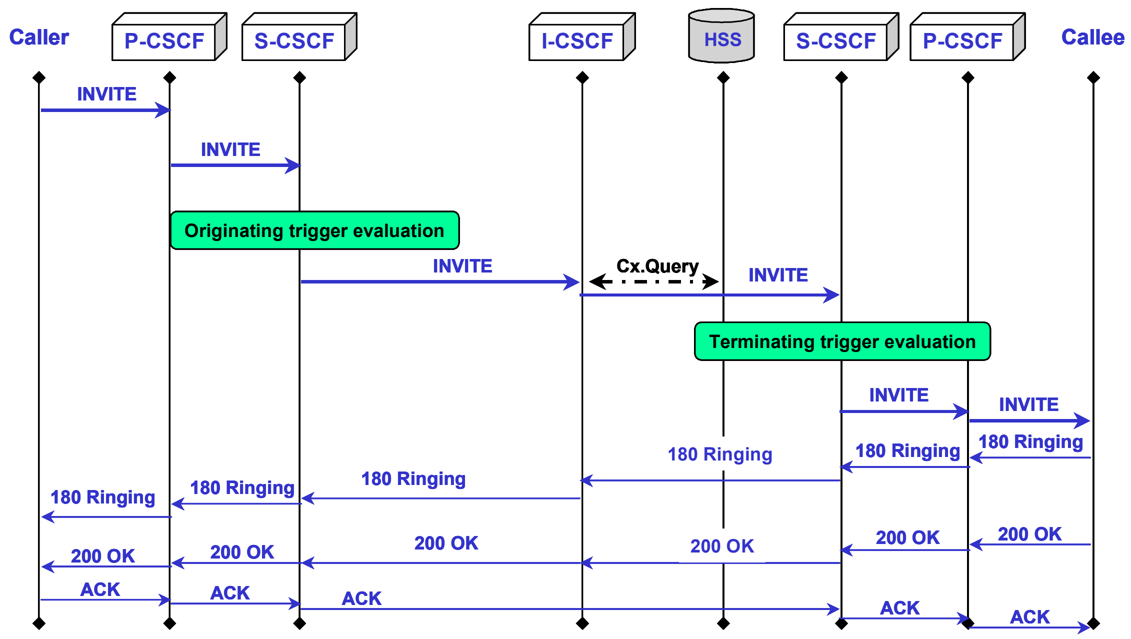

Figure 2 - Simple Call Flow

A simple call flow scenario is shown in Figure 2. The caller sends SIP INVITE to the P-CSCF (Proxy) found prior to registration. The P-CSCF forwards the INVITE to the S-CSCF that is assigned to the caller. The S-CSCF checks if there are any originating triggers for the caller, which is not the case in this example, and forwards the request to the I-CSCF in the terminating network. The I-CSCF uses the Cx interface (DIAMETER) to query the HSS for which S-CSCF the callee is assigned to, and forwards the INVITE to this particular S-CSCF. The S-CSCF checks for terminating triggers of the callee, which is not the case in this scenario, and then forwards the request to the P-CSCF of the callee. When the callee picks up the phone a 200 OK is routed the same way back as the INVITE came. There may be provisional messages (and corresponding acknowledgements) sent that are not shown in this call flow. Note, however, that these are not inherently necessary for a functioning system.

Media

Figure 1 also shows the

- MGCF (Media Gateway Control Function) – Translating between PSTN *signalling* and SIP
- MGW (Media Gateway) – Translating between PSTN *traffic* and IP (RTP)
- MRFC (Media Resource Function Controller) and MRFP (Media Resource Function Processor) – Playing, recording and mixing media streams

An ordinary call between two parties does not involve any media processing of any of these functions.

Push-To-Talk Over Cellular (PoC)

PoC is the cellular way of creating a walkie-talkie service. The PoC client uses SIP as signalling protocol and XCAP to maintain user specific PoC preferences in an XDMS (XML Data Management server). OMA (Open Mobile Alliance) is responsible for standardising PoC and Presence, and was set up in order to provide standards for services that can handle interoperability. The integration of PoC server and XDMS is shown in Figure 3. The XDMS holds user preferences and group lists. To inform the IMS network that a terminal supports push-to-talk, it sends a REGISTER with **+g.talkburst** and **+g.talkburst.groupadd** feature tags in the Contact header.

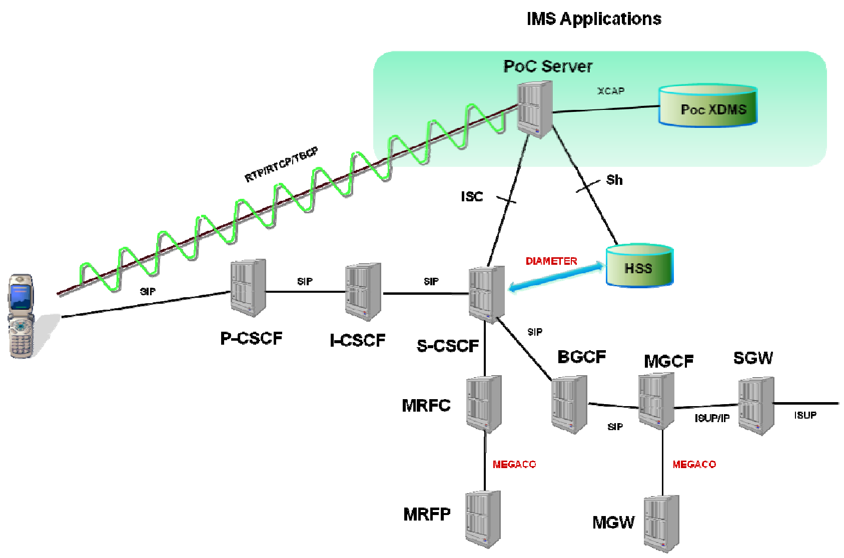

Figure 3 – Integration of PoC server and XDMS in IMS

The PoC server handles SIP signalling, media buffering and transportation. The PoC server also handles the floor control messages using TBCP packets, carried as RTCP (application dependent) packets. When two subscribers are exchanging information, the SIP messages will go through the PoC Servers on the originating side as well as on the terminating side. Trigger matching will be performed on each side causing the SIP message to be forwarded to the PoC servers.

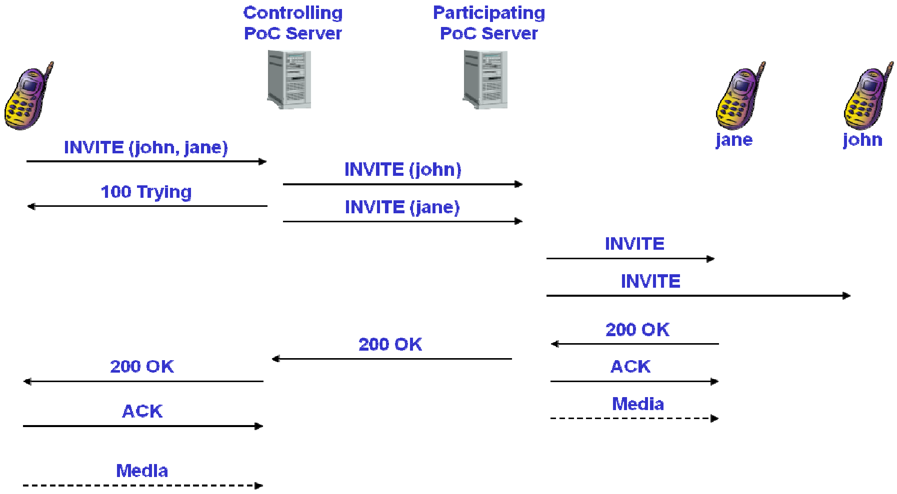

Figure 4 - Controlling and Participating PoC servers

PoC servers can operate in either controlling or participating mode or both. The controlling PoC server performs centralised session handling, like media mixing, floor control and policy enforcement.

How to Integrate applications into IMS

An application server is a SIP server that executes services within IMS. An application server can act as a SIP User Agent Client, SIP User Agent server or SIP B2BUA (Back-To-Back-User-Agent). Application servers are interfacing the S-CSCF that forwards requests to the specific application server. Since there could be several application servers in IMS the HSS contains filter criteria that select (if they match) an application server to contact based on the particular content of an incoming request. The interface from the S-CSCF to the application server is called **Isc** and is defined in 3GPP TS 23.228. Figures 5 through 8 show the different roles an application server may have and the different outcomes in terms of SIP signalling.

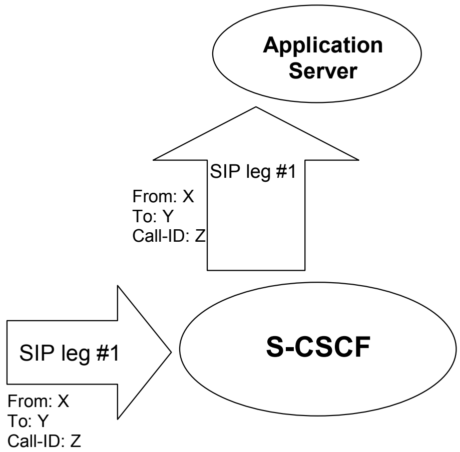

Figure 5 - Application server acting as terminating UA, or redirect server

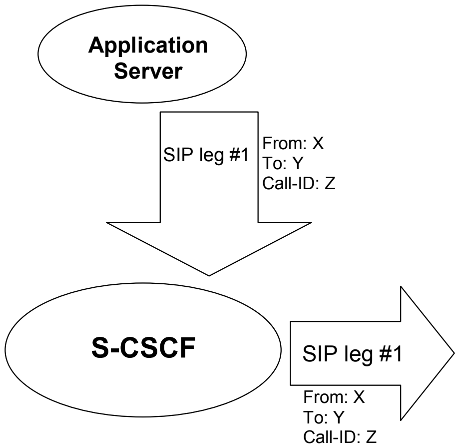

Figure 6 - Application server acting as originating UA

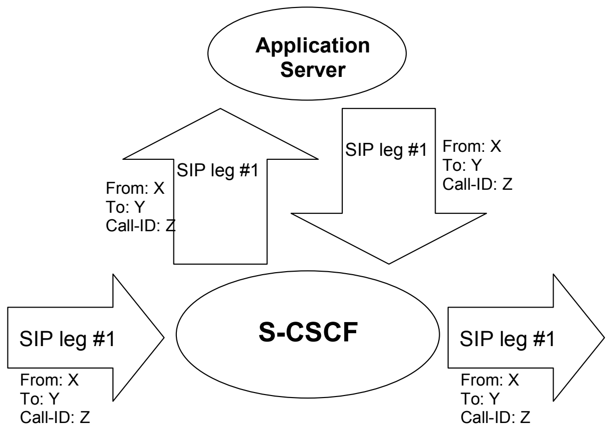

Figure 7 - Application server acting as a SIP proxy

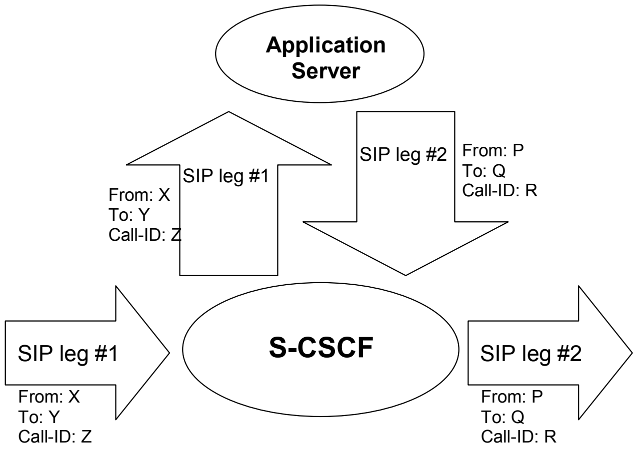

Figure 8 - Application server performing 3rd party call control

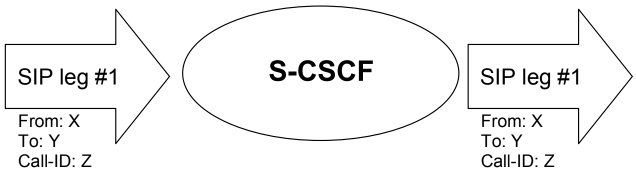

Figure 9 - A SIP leg is passed through the S-CSCF without passing an application Server

In general there are three different types of application servers but they all act as a SIP application server towards the IMS network (S-CSCF). The different types of application servers are:

SIP AS

This is a native IMS application server, based on SIP. It could be implemented in Java, C++ or any programming language. Today there are SIP stacks available written in Java (JAIN) that could be used in order to implement IMS services in a fast and controlled manner. This type of application server is the most likely candidate for implementations of new services. Having Java APIs as a platform and creating services based on SIP/IP with open frameworks like Java is even believed to create an avalanche effect in the number of services developed in the future.

OSA-SCS (Open Service Access – Service Capability Server)

The existing OSA framework application servers need to be integrated into IMS. To fulfill the requirements in IMS, the OSA-SCS acts as an interface between IMS and the OSA application servers. In other words, it uses SIP on one side (as an AS, interfacing S-CSCF) and the OSA application Programming Interface on the other side, towards the OSA application server.

IM-SSF (IP Multimedia Service Switching Function)

Today there is already an existing base of GSM Services implemented using CAMEL (Customized Applications for Mobile network Enhanced Logic). Analogous to the OSA case above, the CAMEL applications need an interface towards the IMS, in order to remain usable. The IM-SSF allows gsmSCF (GSM Service Control Functions) to control an IMS session by acting as an intermediate interface. On one side it acts as an application server with SIP/IP towards the S-CSCF and on the other side as an SCF (Service Switching Function), interfacing the gsmSCF with a protocol based on CAP (CAMEL application Part) defined in 3GPP TS 29.278.

Interfaces Towards the HSS

Apart from the **Isc** interface (towards the S-CSCF), which is a SIP based interface, the application servers have interfaces used for accessing the HSS. The **Sh** interface is responsible for transferring information between the HSS and the SIP AS and between the HSS and the OSA-SCS, e.g. user related information and service related data (MSISDN, user location, cell-ID). The Sh interface is DIAMETER-based and is an intra-operator interface, i.e. it can not be reached from any other operator. The Sh interface is not used between the HSS and the IM-SSF, which instead employs the **Si** interface. The Si interface transports CAMEL subscription information including triggers for use by CAMEL based applications.

Service Profiles

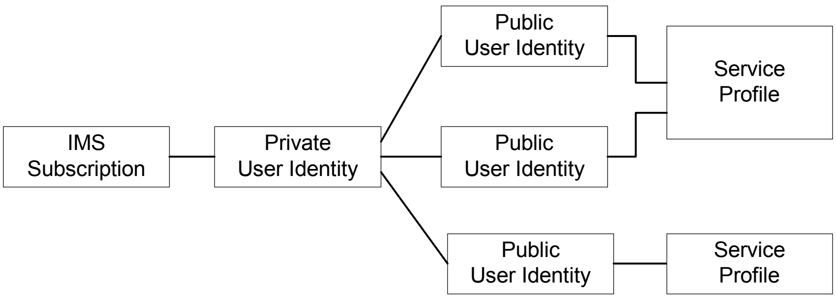

Figure 10 - Private and Public identities in IMS release 7

One of the foundations for all application servers is that the subscriber information stored in the HSS needs to be aware of the types of applications/services that are available for a certain user. Each user has one or several associated *service profiles*, stored in the HSS. When a user registers with the IMS network, it may use "implicit registration", i.e. several public identities associated with the registered private identity are registered. When registering a public identity, the associated service profile is copied from the HSS to the assigned S-CSCF.

E-mail: info@ip-solutions.se, Tel: +46 8 615 08 60, web: www.ip-solutions.se

An UML diagram of the service profile is shown in Figure 11.

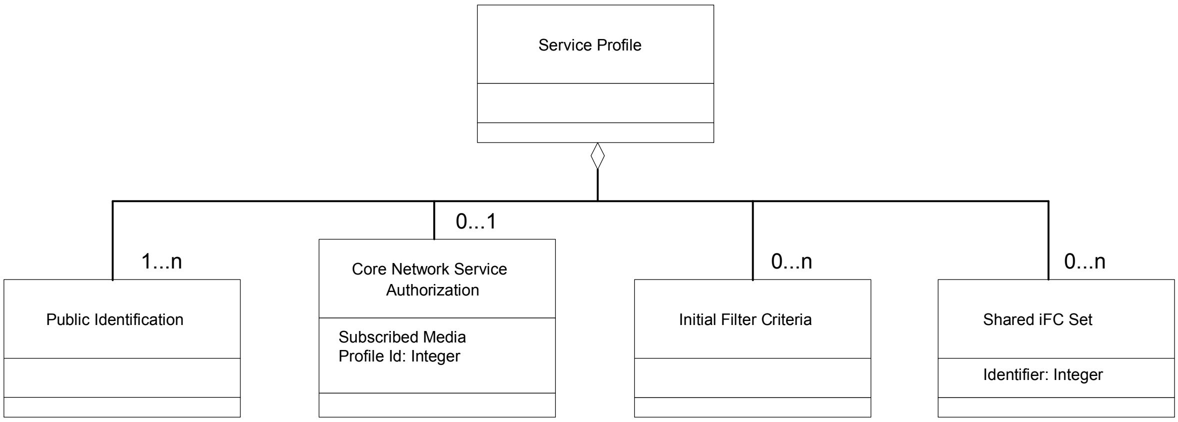

Figure 11 - UML diagram of the Service Profile

The initial filter criteria contains one or zero trigger points. The trigger point consists of a set of rules, or *service point triggers*, that need to be true in order to forward the request to an application server. Each Service Point Trigger is formulated as a boolean expression constructed by AND, OR and NOT in conjunctive or disjunctive form.

An example trigger:

```
(Method="INVITE" OR Method = "MESSAGE" OR Method="SUBSCRIBE") AND
(Method="INVITE" OR Method = "MESSAGE" OR (NOT Header = "from" Content =
"alice"))
```

The Service Point Trigger is then stored in the HSS in XML format and is fetched by the S-CSCF during registration (along with all other user related information).

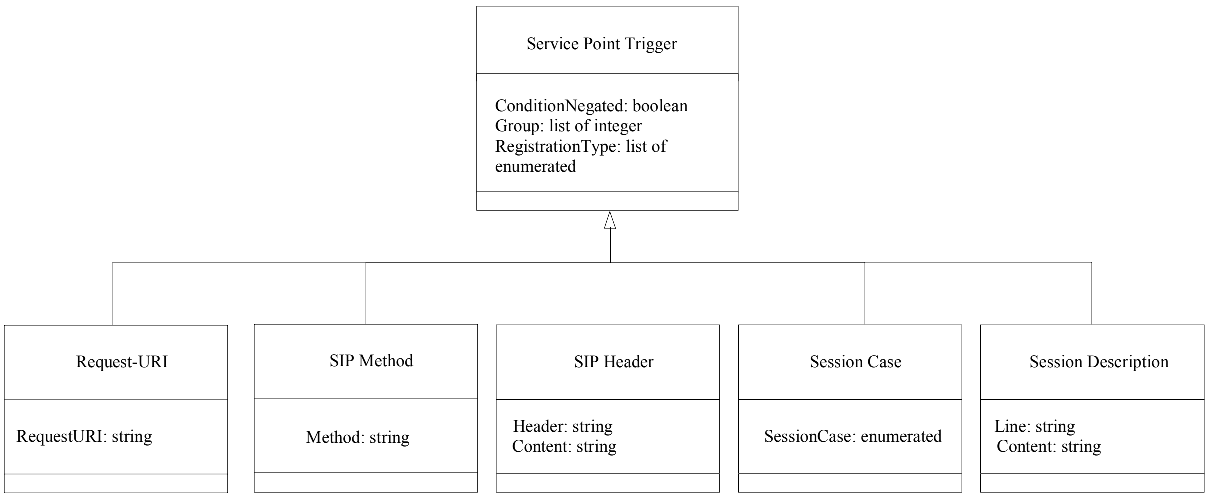

Figure 12 - Service Point Trigger

Development of IMS applications

IMS is based on open standards such as IETF RFCs and 3GPP Technical Specifications and as a base the SIP protocol is used. SIP itself is a very flexible and extensible protocol. It is imperative to have a consistent and exact interpretation of these specifications, in terms of signalling, and such interpretations are much appreciated by developers. If the developers are able to test and debug their applications during the implementation phase without large overhead, it will decrease the overall development phase significantly. Since IMS is a complex network architecture, it is difficult to supply one IMS network for each developer. Effort should be made, however, to support the developers with a good testing and debugging environment.

Testing your application server

Testing can be done at different levels. One level is to ensure that your application server is doing what it is supposed to be doing function-wise, i.e. acting according to the use cases. Both positive and negative test cases should be used for this purpose. Another level, and a more difficult one, is to investigate how your application server acts during stress. Both levels should be part of the test phase. A development project will greatly benefit from having automated test cases with scheduled scripts that may run at any given time and deliver detailed reports. This will most likely also increase the quality of the application significantly. If a certain number of automated scripts have been tested and performed on one version, they could be reused for later releases in regression tests. The main principles to keep in mind are:

- Use automated testing, i.e. scripted test cases
    - Makes your testing more streamlined
- Scripted test cases for regression tests
    - Makes it easy to ensure that previous functionality has not been affected
- Run function tests to ensure that your application behaves in accordance with both successful and unsuccessful use cases
- Run stress tests, with successful as well as unsuccessful use cases
    - Check memory consumption and network performance and behaviour

Performance considerations

IMS applications are prone to putting high demands on processing power. This is due to the inherent properties of the SIP protocol as well as accompanying formats such as XML. XML is a highly flexible format and very easy to implement, but the drawback is that XML parsers and validators typically put CPUs to the test. Bottlenecks should be taken into account when developing IMS applications, and some of the major ones are:

- SIP parsers and XML parsers
  - String management done with reference counters? Threadsafe string operations?
  - XML parsers and validators differ in speed. Parsers could be used in a way to reduce the use for validators in some cases. Is validating done more than once?
- Thread usage and locking
  - Normally, for any given system, the number of threads used should be equal to the number of CPUs if I/O wait is not taken into account.
  - Locking between threads should be carefully considered. The more locks that are used, the more scheduling needs to be done by the system which leads to more waste of CPU time. Several threads and locks can also lead to deadlocks that may be very difficult to break.
- Network processing
  - Asynchronous I/O processing is highly recommended. Socket operations that are performed in blocking mode might start to fail, which in turn forces us to use some other kind of mechanism for reissuing the socket operation. This is a highly unwanted situation and will consume extra CPU and application intelligence to handle one single operation. If there is a queue for doing other operations on the socket, it will sooner or later clog up and create an avalanche effect of outbound operations and the application will sooner or later fail.
  - For building a high performance application on Windows based operating systems non-blocking methods for socket operations in conjunction with IOCompletion ports should be used. Some of the non-blocking operations are WSASend, WSARecv, WSAAccept, WSAGetOverlappedResult, WSAClose, WSAConnect and WSAWaitForMultipleEvents.
  - In Linux based operating systems, there are currently non blocking libraries under development and these will be used in Linux kernels 2.6.9 and later. These are called AIO (Asynchronous IO).
  - Java supports asynchronous socket operations in the java.nio packages. Tests show that asynchronous processing in native C++ is about 27 % faster than Java NIO (JDK

E-mail: info@ip-solutions.se, Tel: +46 8 615 08 60, web: www.ip-solutions.se

v1.4) on an XP based computer and about 10% faster on Linux based computers. Microsofts .NET package in C# falls short, being 44% slower than the C++ native asynchronous processing. In other words, on a Windows based computer, the best alternative is C++ native asynchronous IO, then Java NIO packages and lastly .NET C#. In Linux based systems, Java is quite close to native C++ with only a 10% margin.

References

- 3GPP TS 23.228 – IP Multimedia Subsystem (IMS); Stage 2
- 3GPP TS 29.228 – IMS Cx and Dx Interfaces; Signalling flows and message contents
- 3GPP TS 29.278 – CAMEL Application Part (CAP) specification for IMS
- 3GPP TS 32.240 – Telecommunication management; Charging management; Charging architecture and principles
- 3GPP TS 32.260 – Telecommunication management; Charging management; IMS charging
- RFC 3588 – DIAMETER
- Async IO on Linux – http://lse.sourceforge.net/io/aio.html (Last visited 2008-01-30)
- Comparing two high performance patterns – http://www.artima.com/articles/io_design_patterns3.html (Last visited 2008-01-30)