Optimising SIP for Narrow Band Signalling Channels

Raimo Kantola
Department of Communications and Networking
Helsinki University of Technology, TKK

Abstract:

This Chapter in Lecture Notes explores the problem of signalling efficiency of the Session Initiation Protocol in a cellular network. We present performance metrics on SIP and compare SIP signalling efficiency with ISDN Q.931. We present measurement results on SIP and SigComp performance. We explore alternatives to SigComp signalling compression such a TLV encoding for SIP/SDP (of our own definition since such a thing does not exist). As another alternative, we present an estimation of the efficiency of using zipping for compressing SIP/SDP messages.

1 Introduction

The Session Initiation Protocol has been chosen by the 3GPP as the future signalling protocol for advanced packet based communication services in cellular networks. With SIP, the Session Description Protocol is used to describe the session parameters. SIP and SDP use UTF-8 representation for information and are very general [5]. SIP for example supports both stateless and stateful proxies and thus must carry a lot of context information in each message. Adding all the cellular network specific requirements have lead to signalling messages that may be ca 1500 octets long and to session establishment procedures with many messages. As a result, in an UMTS network session establishment takes a long time and the signalling overhead for essentially narrow band services such as voice is quite high leading to inefficient use of the expensive bandwidth resource.

By taking a long time we mean that session establishment using 3G IP Multimedia Subsystem procedures takes much longer than a similar service takes in GSM. For example setting up a voice call using the 3GPP signalling sequences requires about 80 kbits of signalling information to be sent over the originating air interface. If a narrow band voice codec at 8 kbps is used, users can speak for some 11 seconds to generate the same amount of bits as were needed for signalling for the call.

Signalling compression has been suggested as the solution to this problem in RFCs 3320, 3321, 3485 [7, 8, 9]. However, an experimental study [2] shows that the achievable compression ratio without Huffman coding is at best in the order of 4 to 1, which is not enough to solve the problem. Moreover, the best compression ratio is achieved only for the lengthiest signalling flows with a heavy cost in processing cycles and memory. SigComp has the advantage that it is a generic solution to the problem of compressing protocol messages. On the downside, SigComp uses a lot of memory and processing cycles even without Huffman coding. With Huffman coding, the efficiency of SigComp can be improved probably to a compression ratio in the order of or close to 10 to 1. Also, in this case, the best ratios are achieved with only the lengthiest signalling flows.

This paper first explores the extent of the problem by presenting some performance metrics on SIP signalling and drawing some comparisons of SIP with ISDN Q.931. The paper analyses some design solutions in SIP and suggests optimisations that could be applied in a cellular access network without sacrificing the general nature of SIP. The analysis intends to demonstrate the causes of inefficiency in SIP. We carry out the first steps of optimisation in the text domain and then move to Type-Length-Value/Attribute-value pair (TLV/AVP) encoding. As an alternative we take a look at using zipping for compressing SIP/SDP. The paper estimates the signalling efficiency after each optimisation step.

2 Signalling Efficiency

2.1 What is signalling efficiency?

From the user's point of view, it is interesting that the post-dialling delay should be minimal. Post-dialling delay is the time that elapses from the point when the user has finished dialling till he/she hears the ringing tone.

However, using post-dialling delay to make judgements about a particular signalling system or dimension signalling channels is inconvenient. One needs to have a network model and possibly assume a combination of several signalling systems for the call. For the limited purpose of dimensioning signalling channels, or more generally the signalling network or for the purpose of making judgements about a particular signalling system, it is more convenient to use a different measure.

For digital signalling systems a good metric is look at the size of typical signalling flows created using the signalling system. Flow size can be measured in bits. Bits can be turned into time spent by the signalling flow by formula (1) (see LNotes on ISDN):

Transfer delay = size in bits/signaling channel speed. (1)

We will use this measure for SIP. First, however, let's look at some measurement results from a 3G test network.

2.2 Average performance of session establishment using SIP

In a study the delay of session establishment was measured in different access networks when the text based SIP was used [1]. Some of the results are presented in Table 1. In the UTRAN measurements the network allocated at least 64 kbps for signalling and the user device was capable of 128/384 kbps UL/DL speeds.

Table 1: Average session establishment delay using SIP

	IETF signalling UTRAN	3GPP signalling UTRAN	IETF signalling GPRS
Delay measured between GGSN and IMS	1.4s	8.8s	Not measured
Delay measured at end user	3.3s	11.5s	8.0

If a user is not registered in the IMS but wants to start a session, the overall delay experienced by the end user consists of the following parts: the IMS registration, session establishment and call acceptance. With 3GPP SIP signalling over UTRAN, this takes on average 27.9s [1]. IETF signalling drops the delay to 20.2s [1] which is still high compared to GSM. The call setup delays according to Foster et al. [3] for Mobile Originated and Mobile Terminated GSM calls are about 2 seconds. When UMTS release 99 circuit switched call procedures are used, the delays are a bit shorter than in GSM [3].

Another study [2] implemented a rather comprehensive SigComp prototype except for Huffman coding and tested its performance extensively. SigComp achieved a compression ratio of less than 4:1 in the study. Table 2 shows the 3G SIP signalling flow summary for 3GPP release 5 [4]. Naturally, the length of the messages depends on the length of the names that are used. However, shortening the names defeats the main argument of human readability for text based signalling.

Let us study the 3GPP release 5 signalling flows found in [4] for both Mobile Originated and Mobile Terminated session setup. Assuming that the UTF-8 encoding is used like in [4], the total length of the MO session setup sequence in Table 2 is 79000 bits. The Mobile Terminated session setup procedure is even lengthier and totals about 96000 bits. Table 2 shows the number of UTF-8 characters in each of the MO and MT messages in columns two and four respectively and the cumulative bits in the flows in columns three and five.

Table 2 SIP messages in Session Setup in 3G IMS

	MO char	Cumulative MO bits	MT Char	Cumulative MT Bits
INVITE	1427	11416	1687	13496
100 Trying	249	13408	534	17768
183 Session Progress	1390	24528	1605	30608
PRACK	1310	35008	1185	40088
200 OK	902	42224	1206	49736
UPDATE	1283	52488	1146	58904
200 OK	854	59320	1153	68128
180 Ringing	558	63784	898	75312
PRACK	674	69176	571	79880
200 OK	255	71216	558	84344
200 OK	523	75400	876	91352
ACK	447	78976	547	95728

Let us consider the components of the session setup delay. One of the components is the message transfer delay (*d*) on a narrowband signalling channel. This delay follows formula 1.

$$d = \text{size / speed of the signalling channel} \qquad (1)$$

Considering a mobile to mobile call, the total delay caused by transferring signalling over the two air interfaces is almost the sum of the delays according to (1) excluding messages that appear on the two interfaces truly in parallel. The 180 Ringing is likely to be such a message. We assume that it may appear in parallel on the MT interface while the MO interface is transferring the previous OK –message. In human to human communications there are no other messages that are likely to benefit from a similar parallelism in the message flow. Therefore, for our reasoning we will assume that the total delay on the two air interfaces in a mobile to mobile call is the sum of the MO and MT delays according to (1) minus the delay caused by the Ringing message on the MT side. Note that the Ringing message is the shorter of the two parallel messages.

In Figure 1, we present the accumulation of the Mobile Originated SIP message transfer delay on 16 kbps, 64 kbps and 128 kbps signalling channels using formula 1 and the message sizes in Table 2. The Figure shows that the delays experienced by such long messages on a reasonable signalling channel are significant. In a 3G Mobile to Mobile call the total transfer delay for call setup is about 10.5 seconds assuming that 16 kbps signalling channels are used.

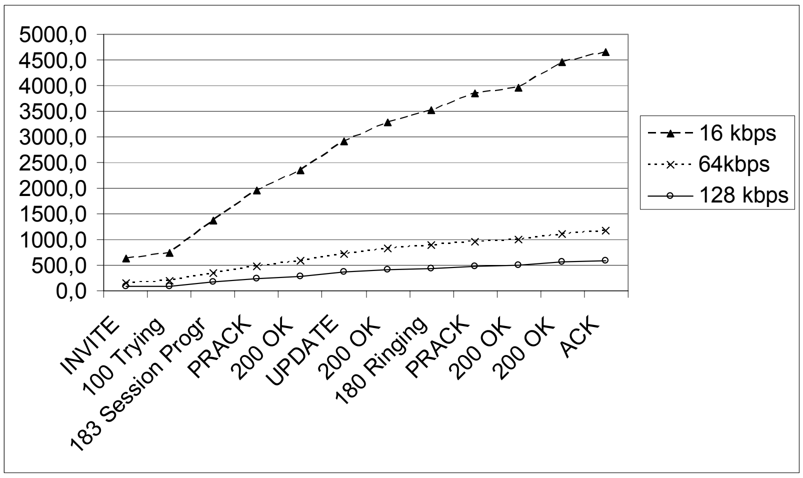

Figure 1: MO SIP message transfer delay on a signalling channel in ms.

UMTS networks can be configured to use signalling channels of different speed. However, it is reasonable to claim that a performance criterion for a signalling system is its capability to use a signalling channel that takes no more radio capacity than the service that is being established and that only for a period of time that is not significantly longer than what current signalling systems need. Most of the service use in 3G will consist of essentially narrow band services such as voice, messaging or data without graphics. An example of a narrow band data service is presence. Using a higher capacity signalling channel would be justified if the signalling channel were shared, which is not the case. This would help with the delays but can not help wasting air interface capacity for non-chargeable bits. We should also note that Figure 1 shows only one major component of the session setup delay. Any current version of 3GPP SIP can be expected to become more feature rich as time goes on. Figure 1 shows that in order to achieve reasonable performance, the access for IMS services needs a 128 kbps signalling channel. This is also the recommendation proposed in [1].

Let us assume that packet voice uses a coder at 4.75 (or 8) kbps. It takes 18 (11) seconds of speech to produce as many bits in a Mobile Originated call as are needed for the corresponding session setup and release on the same interface. Taking into account the lower layers of signalling would present an even more dismal picture.

From the analysis we can conclude that text based SIP in IMS is inefficient in using the cellular radio capacity. If the signalling channel is configured for low speed, it leads to a significantly lengthier session setup than in GSM or ISDN. It is reasonable to expect that users will not be happy about such services. To avoid introducing unusable services to the market, the operators are forced to configure 3G signalling channels to use at least 128 kbps for IMS signalling. This is a clear waste of precious radio capacity for something that from the users' perspective is pure overhead and non-chargeable for the operators.

Naturally, time and Moore's law tend to offer a cure. The High Speed Packet Access brings higher speeds to the air interface allowing configuring higher speeds for signalling. When cells become smaller, one cell is used by fewer users at any given time. These things will help but the air interface will still remain a shared precious resource for a long time.

2.3 Signalling message delays in ISDN

We captured an ISDN signalling sequence of Q.931 and Q.921 using a basic rate interface with 16 kbps for signalling. With 8 digits for both the calling and the called number, call setup creates a signalling transfer delay of 44 ms and the whole call flow uses 62 ms for transferring layer 3 and layer 2 signalling bits. Results were presented in Lecture Notes on ISDN.

2.4 Conclusions on signalling delays

In our analysis of the 3G signalling we have ignored the lower layers completely and concentrated on SIP alone. In case of ISDN we consider the combined message transfer delay created at layers two and three. Even so, the comparison shows that SIP and SDP, as specified for 3G and presented in 3G documents [4], takes 100 times more time than ISDN signalling on the same signalling channel. On the other hand it is easy to see from Figure 1 that the large size of SIP messages does not matter much in a broadband environment such as WLAN where signalling speed is calculated in megabits per second. The comparison also shows that besides the overhead due to text representation instead of binary, SIP is much more verbose than ISDN signalling.

3 Optimisation of SIP for cellular access

In this section we take a critical look at the 3G signalling sequences as they have been presented in [4] and suggest an optimisation process for narrow band SIP. Taking the optimisation step by step we can see the sources of inefficiency in SIP. Our target is a compression ratio of better than 10:1 which is probably better than what SigComp can achieve. Such a compression ratio would allow signalling channels of ca 12 to 16 kbps instead of 128 kbps.

First, we can assume that P-CSCF, the first proxy in the IMS always has state and can form a virtual connection with the UE. The SIP/SDP flows consist of lines of text and contain a lot of redundancy, the same information is sent many times. Due to the assumption of stateful P-CSCF and UE, we can remove most of the redundancy by introducing a reference to a line that has appeared previously in the session. The receiver can append the removed signalling message text using its state. One of the things that will happen is that we remove headers for session identification since the corresponding headers are repeated in every message.

Central to the optimisations is session identification. When we remove the lengthy Call-Id, To – and From header based session identification, we must put in place an alternative mechanism that will preserve session integrity and is robust. This can be done based on the source IP address and port and a session reference value. The session reference value can work at least in one of two ways. The first option is like in Q.931. A session id is allocated by the initiator and a flag having two values says who has allocated the session reference (UE or proxy). The second option is that both the Initiator and the Responder allocate their own reference values and the peer will always use at least the receiver's value in the reminder of the flow. Learning each others reference values takes place during the first two messages that are exchanged between a UE and P-CSCF. Naturally, this approach assumes that there can be no proxies between the UE and the P-CSCF.

Moreover, we can remove all headers in SIP messages and all elements in the SDP that are there for the stateless proxy option. This means that we remove the headers and parameters with address information originating from the UE. Unnecessary address information coming from the user opens a door for spoofing and that we do not need in telephony. On the other hand, if such information is needed, it can be registered and properly verified in the process of registration rather than complicating the session setup itself. We also remove the headers or parts of headers that describe the signalling path. This is justified because in IMS the path from the UE to the P-CSCF is fixed and because a user is not really interested to know what path the signalling message took on its way through different proxies.

3.1 Optimisation process for narrow band SIP/SDP

Let us use the following process for demonstrating and verifying the efficiency of different optimisations. Let us first reconstruct a full Mobile Originated session setup and release flow from [4] so that all lines of text in both SIP and SDP are in place. Let us place the flow in a spreadsheet (I used Excel). The selection of the MO flow is arbitrary and done for the sake of limiting the amount of processing. The process has the following steps.

1. Instrument the lines of text in SIP/SDP that should not be removed. These are the start lines.
2. Add line numbers to the flow. Since both the sender and the receiver see the same messages, they can also number the lines identically and independently.
3. Sort the lines in an ascending order by content.
4. Replace replicate lines with line references to the first line in a sequence of identical lines. For our purpose and without loss of generality we will assume that a reference has a form “=12” or “=123” or “=12 23 34 56 123”.
5. Restore the flow into the original order.
6. Add session reference values. Simplify the protocol: Tempting headers to be removed are the To –header because the information in it is not used by the network, the Call-ID –header just because it seems pointless to carry a global call identifier in every message, the Max-Forwards and Via –headers because our communicating parties are connected in a point to point manner over a PDP context and finally all references to SigComp because we are not going to use SigComp. Note also that the above modifications lead to removing lines from the protocol. As a result, we also need to remove the line references in the flow to the removed lines.
7. Identify long valued repeated parameters by placing the parameter values on their own lines in a spreadsheet, adding a new numbering to the lines, sorting the lines in an alphabetical order, replacing replicate parameters with parameter references pointing to the first parameter value in a sequence of identical values and restoring the whole flow back to the original order.
8. Introduce an option for delegating the responsibility of routing SIP messages to the proxy CSCF. Such an option could be requested by the user device in the INVITE message. In the remaining message no ROUTE or ROUTE RECORD headers would be needed.
9. Introduce TLV encoding for the remaining headers and SDP lines. Usually compound TLV elements correspond to lines and nested TLV elements to parameters. All compound and nested TLV elements can be numbered in the order of appearance of types in a session. Thus the references to lines and parameters are replaced by references to TLV elements by introducing two types for that purpose. One will contain a list of one byte references as values and the other a list of two byte values as references.

We show the process of optimisation in steps 1 to 9 in detail in [6]. Alternatively, we can take the original MO flow or the optimised flow after step 8 and use zipping for compression. The value of the above step by step process is in showing the sources of data redundancy in the flow and in preparing a base for an efficient TLV encoding. We will use both approaches and compare the results.

3.2 Optimisation results

The size of the reconstructed MO flow including session setup and release is 10810 UTF-8 encoded characters. We will present the results after steps 5 to 9. In our evaluation we compare the optimised SIP flow with the original flow using the compression ratio that is the relation of the original number of bits in the flow to the number of bits in the optimised flow.

After step 5, we have removed 63% from the reconstructed MO setup and release sequence and have 4004 characters left in the remaining lines. The line references to the removed lines take 695 characters of the remaining total. The compression ratio after step 5 is 2.7. This is a little less than the SigComp compression implemented in [2] could achieve. In the process we have removed session identification from all messages but INVITE, so the resulting protocol will not work. We fix this in the next step.

In step 6 we remove 332 characters and the compression ratio grows to 2.9. We assume that a user created session reference is of the form "Sf: U12" and a P-CSCF created network session reference is of the form " Sf: N12345".

In step 7, we look for repeated parameters with long values and replace them by parameter references. There are 23 cases where we can replace a long parameter value with a reference to the same value since the value appears earlier in the flow. Typical long values are names, addresses and telephone numbers. After these modifications, the flow is 3302 bytes and the compression ratio is 3.3 compared to the original. This shows that SigComp is unlikely to work on any arbitrary chucks of text but seems to be replacing redundant lines and parameters. This result can possibly be used to create a compression method integrated with SIP such that instead of parsing the message again, the method would work directly on the binary SIP/SDP syntax tree saving memory and processing cycles.

In step 8 we introduce the option of delegating the routing responsibility to the network. A user device could ask for such an option in the INVITE message and in the remaining messages no ROUTE or ROUTE RECORD headers will appear. The result is an optimised UTF-8 flow of 3062 bytes and the compression ratio is 3.5. It is noteworthy, that the remaining flow has 730 bytes of references to lines and long valued parameters. The share of references is so high because the references are also in UTF-8 encoded text. Removing all those references would increase the compression ratio to 4.6.

At step 9 we review the remaining text line by line in order to estimate the efficiency of TLV encoding for SIP/SDP. First we create a common header for all SIP messages. It is presented in Figure 2.

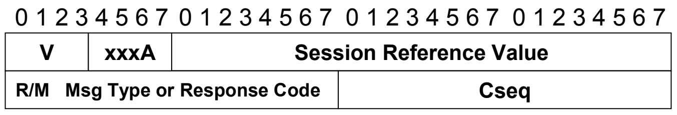

Figure 2: Proposed common header for all TLV encoded SIP messages

The common header has fields for session and transaction identification, for the SIP version, for Message types that distinguish request messages and for response codes that appear in responses. For the reference value we also need a flag to indicate whether it was allocated by the UE or the proxy or alternatively by the sender or the receiver.

For network allocated session references at least 3 bytes are needed because it is easy to imagine a system that can handle more than 65 000 simultaneous sessions. On the other hand the need to be able to identify more than 16 Million simultaneous sessions locally is not clear. Therefore, 3 bytes would seem to be just right. For sequence numbers for ensuring the order of messages in a session it would seem that 16 bits would be quite enough in a cellular context. One could even argue that one byte is enough.

Response codes can be encoded with 10 bits (response codes are between 100 and 699). By using one bit to distinguish between response codes and message types, we can encode both the message type and the response code in the same two byte field since they do not appear in the same messages.

We assign new types as new information items appear on the lines and use new and existing types to estimate the space needed for the Lengths and the Values in the TLV elements. After step 9 of our optimisation procedure, we have 53 types that are needed for encoding the MO session setup and release and have used 1186 bytes for the whole flow without routing delegation and only 1056 bytes with routing delegation. The compression ratio is now 9.1 to 1 without routing delegation and 10.2 to 1 with routing delegation. We present the result of this and other intermediate steps in [6].

Figure 3 summarises the effects of different optimisation steps. The TLV encoded result with routing delegation still has about 20 percent of UTF-8 encoded text in values. It has many references to popular lines. Therefore, if we are ready to throw computing cycles into compressing the result further, we could try using a Huffman code for a part of the TLV encoded result. This may help for sessions if the code is agreed in the Registration procedure.

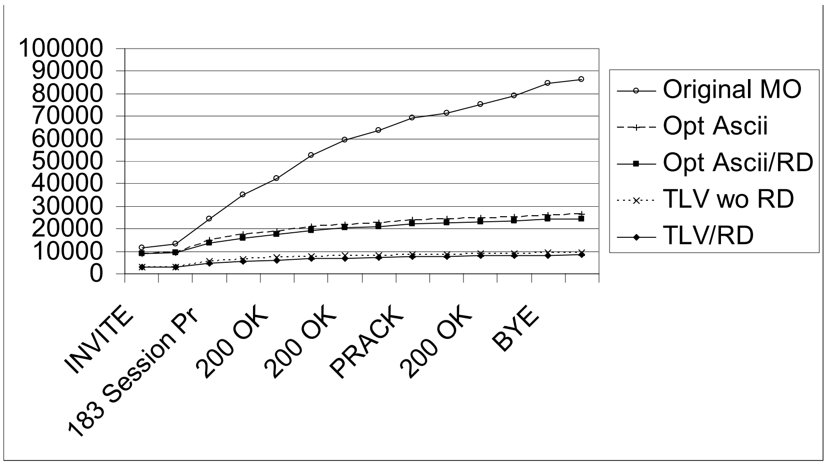

Figure 3: Comparison of different optimisations in accumulated bits

The Figure demonstrates that the TLV encoding works well also for short flows while optimisation in the text domain tends to bite seriously only from the fourth message onwards.

3.3 Zipping as an optimisation method

Zipping is available in well known tools such as gzip, zlib and WinZip. They first find repeated strings of bytes, replace them by distance-length pairs (LZ77) and then can use Huffman coding to encode most frequent symbols with short codes and less frequent with longer codes using a prefix code. Typically, the tree of codes is stored with the resulting file. We use the WinZip program to compress each of the original and optimised text messages individually. In addition we create all sub flows of the original MO flow and the optimised UTF-8 encoded text flow. We zip the sub flows to model the use of this compression method incrementally during a session.

The results show that zipping each message individually gives an overall gain for the whole MO flow of 1.7 to 1 while incremental use of WinZip on the best optimised MO flow gives a compression of 8.7 to 1 compared to the original flow. The results also show that WinZip is good at finding repeated lines and parameters in the source and removing them manually does not pay off. Figure 4 presents the scatter diagram of the results showing compressed size as a function of the original SIP/SDP size. The Figure shows that roughly speaking until the size of about 2000 bytes WinZip gradually achieves a compression ratio of about 2:1 and for the reminder of a flow the ratio is about 25:1, i.e. some 400 zipped bytes are generated from about 10 000 bytes of SIP/SDP material. The gzip tool seems to create results that are a bit less compact but the difference is within 10% of WinZip. Quite naturally, for message sizes of just a few hundred bytes, the compression ratio is significantly less than 2:1.

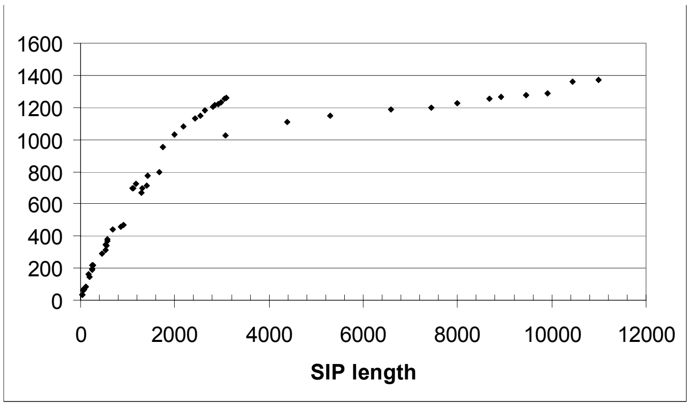

Figure 4: WinZipped length of messages and sub flows of the MO session

Let us now consider how efficient a real working solution based on LZ77 and Huffman coding could be for SIP and SDP in cellular networks provided that the solution allows agreeing on the Huffman code in the registration procedure and possibly creates a seed string for later use in compressing of sessions. By looking at Figure 4, we can guess that due to the fact that no Huffman tree information needs to be transferred, it may be possible that for call sessions this approach could also achieve the compression ratio of 10:1 like we have demonstrated with TLV –encoding especially if we do some protocol simplifications such as we have suggested in this paper. However, this remains to be proven with a prototype. Also we expect that due to the incremental application of zipping, this approach is less efficient for short flows than for long flows.

4 A new architecture for SIP optimised for cellular access

We present an architecture that adds a TLV representation to SIP as an option that can be efficiently used for narrow band services in narrow band networks. Concerning IMS the justification for the introduction of such a SIP variant is twofold: (1) we achieve signalling efficiency that meets user needs as we have shown and (2) 3G already has many protocols, one new protocol variant does not hurt. It is reasonable to expect that a TLV encoded SIP would also be more efficient in using processing cycles than the combination of SIP and SigComp by two to three orders of magnitude. In the use of memory the gain is likely to be significant but less than in cycles. The proposed architecture is presented in Figure 5.

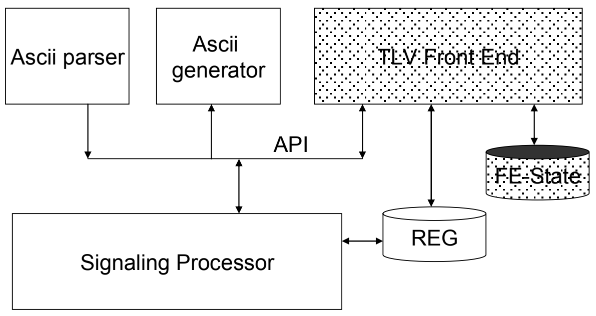

Figure 5: A new SIP architecture

There are essentially three new elements in the architecture. The first is the TLV Front End. The second is the database for the state used solely by the Front End. Third, the Application Programming Interface between the Signalling Processor and the parsing and generating SIP messages has been made explicit. The API is based on binary data structures that carry the content of SIP messages in a format that can be directly processed by the signalling processor. A SIP implementation always needs either the text parser and the signalling text generator or the front end with its database or both of them. The Signalling Processor and the Registrar stay compulsory just as they are in the present architecture.

Following the architecture in Figure 5 we may create inexpensive mobile devices with packet services that only speak TLV SIP but do not understand the text encoded SIP at all. A network based implementation or a high end mobile device would typically have either text and TLV capabilities or the text capabilities only. This allows carrying out the new feature specification first for the text version and introducing TLV encoding for the features in a separate effort if so desired by the industry.

The task of the TLV Front End is to read and generate TLV encoded SIP messages and use all the optimisations suggested in Section 3. It identifies sessions based on IP addresses, port numbers and session reference values. A new standard port needs to be allocated for TLV SIP. The TLV FE replaces references to previously seen compound TLV elements representing lines of UTF-8 SIP and references to nested TLV elements representing SIP parameters by their content. The TLV FE uses default values as registered by the user to generate regular SIP messages. The Front End stores SIP and SDP data into its database so that it can benefit from the pieces of text that it has seen in a signalling flow or that has been generated during the signalling flow.

5 Conclusion

This paper has demonstrated the inefficiency of using text based signalling in a narrow band network for narrow band services. The efficiency of 3G SIP/SDP signalling flows was shown to relate to the efficiency of ISDN signalling as 1:100. We brought out measurement results showing that SigComp without Huffman coding can only cut the length of a 3G SIP flow to one fourth at best. We did not bring out facts but pointed out the issue of the efficiency in terms of processing power or memory usage. In our experience the combined efficiency of SIP and SigComp relates to ISDN CPU and memory use as something in the order of 100:1 to 1000:1. This paper explored the possibility of eliminating some of the information in 3G IMS SIP messages. We were able to show the sources of improvements in signalling efficiency by performing the optimisation step by step. We assumed that P-CSCF and UE always keep state for a session and redesigned the SIP session identification mechanism. The elimination of repeated lines and long parameter values in addition to the suggested protocol simplifications seem to give about the same gain as using SigComp without Huffman coding.

Looking for a more efficient compression method, we drafted a TLV encoding for SIP/SDP and suggested a new architecture for SIP applicable in narrow band networks for narrow band services. We showed that our mixed Type-Length-Value/Attribute Value Pair encoding achieves a compression ratio of better than 10:1 compared to the original Mobile Originated session flow on the UE to P-CSCF interface. This results a significant improvement over the release 5 text based SIP/SDP that requires a signalling channel of 128 kbps.

We also explored an alternative optimisation approach using the well known LZ77 and Huffman coding that are used in many zipping programs. Also this approach may be able to achieve a compression ratio of 10 to 1 for call sessions but this remains to be proven by a prototype. We expect that zipping is less efficient for short flows than for long flows while TLV encoding is efficient for all flows. The advantage of zipping would be that SIP remains independent of the compression method. The disadvantage is that like SigComp this approach uses a lot of memory and computing cycles. All compression methods that work best for long flows would benefit if some initial state can be created during Registration.

SigComp leads to wasting a lot of processing capacity for converting the initially binary information to text and from text to binary. The receiver receives binary, converts it to text, and parses the text converting it to binary. This requires a lot of memory and finally is inefficient in terms of the achievable compression ratio. The processing cycles and the memory consumption can not and should not be ignored when all this is implemented in a mobile device or thousands of sessions need to be processed in parallel.

We further argue that TLV and Attribute Value Pair encoding is easy to process, does not require heavy data conversions and provides a reasonable compression ratio as compared to the current text based SIP/SDP. Alternatively, we should agree on a fixed compression algorithm such as zipping (LZ77 combined with Huffman coding) and allow for agreeing on the code and creating a seed string for further compression in the registration procedure.

Our analysis also shows that solutions exist that achieve even better signalling efficiency than the proposed SIP/SDP variant. We, however, argue that a significantly more efficient protocol would have to be based on a monolithic communicating state machine approach. Moreover, such protocols are being designed by porting the circuit switched GSM signalling on top of an IP infrastructure. Or instead of SIP we could use H.323. Therefore, there are limits after which further optimization will not produce anything new.

6 Further study

This paper analysed a problem and a possible solution on the level of requirements. It remains to be proven that a TLV/Attribute Value Pair encoded representation can be added to the SIP architecture so that the flexibility and openness of SIP for new services is preserved. Naturally, precise definition of the representation and prototyping is needed. Trying to generalise the result is important. We should also consider using the resulting binary protocol everywhere in the IMS for cellular network services. In parallel the efficiency of using LZ77 and Huffman coding should be verified.

[1] Hanna-Pauliina Heiskanen, Quality of Person-to-Person Services in the 3GPP architecture, Master thesis, TKK, June 2005.

[2] Jouni Mäenpää, Performance of Signaling Compression in the 3G Mobile Network, Master thesis, TKK, June 2005.

[3] Foster G., et al. Performance Estimation of Efficient UMTS Packet Voice Call Control, Motorola & Cork Institute of Technology, 2002.

[4] TS 24.228, v5.12.0 Signaling flows for the IP multimedia call control based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3, 3GPP.

[5] J. Rosenberg, et. al, RFC 3261 – SIP: Session Initiation Protocol.

[6] R. Kantola, Steps of Optimising SIP for narrow band signalling channels, Technical Report, Networking Laboratory, TKK, 2005.

[7] Price R. et. al., 2003, Signaling Compression (SigComp), RFC 3320.

[8] Hannu H. et. al., 2003, Signaling Compression (SigComp) – Extended Operations, RFC 3321.

[9] Garcia-Martin M. et. al., 2003, Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Static Dictionary for Signaling Compression (SigComp), RFC 3485.