13th ICCRTS: C2 for Complex Endeavors

"Using NFFI Web Services on the tactical level: An evaluation of compression techniques"

Topic 7: Network-Centric experimentation and Analysis, Topic 2: Networks and Networking, or Topic 9: Collaborative Technologies for Network-Centric Operations

Frank T. Johnsen and Trude Hafsøe
Norwegian Defence Research Establishment (FFI)
P.O. Box 25
NO-2027 Kjeller
Norway

Point of Contact:

Frank T. Johnsen
Norwegian Defence Research Establishment (FFI)
P.O. Box 25
NO-2027 Kjeller
Norway

+47 63 80 79 60
frank-trethan.johnsen@ffi.no

Using NFFI Web Services on the tactical level: An evaluation of compression techniques

Abstract

Blue force tracking is recognized as one of the most important aspects of the Network Enabled Capabilities (NEC) concept. In complex endeavors where several different nations take part, blue force tracking is important to avoid possible blue-on-blue situations. To facilitate interoperability between nations, NATO has specified a format for exchange of friendly force tracking information; NATO Friendly Force Information (NFFI). Part of the NFFI specification is an XML schema to allow the exchange of blue force tracking information using a Web service. To make systems interoperable at all levels, it is desirable to use XML encoded NFFI also at the tactical level. XML, while being a standardized way to structure data, leads to large text documents that need to be exchanged. At the tactical level bandwidth is scarce, and measures must be taken if one is to use an NFFI Web service. By compressing the XML document it requires less bandwidth to transmit the same amount of information over the network, and it becomes feasible to use NFFI also at the tactical level. We have evaluated several different compression techniques on a set of tracks encoded as NFFI XML documents. It is clear that NFFI is very compression friendly, and the compression rate increases with the number of tracks contained in the NFFI document. In this paper, we present the results from our compression technique evaluation.

Keywords: NFFI, XML, tactical level, compression

Introduction

The aim of NEC is to increase mission effectiveness by networking military entities, enhancing the sharing of information and situation awareness. The key prerequisite of shared situation awareness is increased access to (and sharing of) information. By using a Service-Oriented Architecture (SOA) [6] as a foundation for the information infrastructure, military resources may be made available as services that may be published and utilized over a communication infrastructure. The service itself is defined by using a well-defined interface that exposes the functionality and hides the underlying implementation details. Services may be aggregated, by either the service provider or service consumer, to create more advanced services. This modularization makes introduction of (and dynamic reconfiguration of) services easier.

Web services is a promising technology for implementing a SOA [7], allowing for dynamic information sharing between military units. Web services provide loose coupling of functional entities that allow for the dynamicity and flexibility required in NEC.

Web services technology is in widespread use on the Internet today, and COTS products are readily available. Thus, it makes sense to attempt to utilize this technology for

military purposes. This seems to be a general trend in the industry as the Network Centric Operations Industry Consortium [1] supports the WS standards.

In NEC there is an ambitious requirement for users at all operational levels to seamlessly exchange information. In order to achieve efficient information exchange between these users, the Web services solutions need to work with different types of information and communication systems. Systems and equipment used at the various levels are different, and the information exchange must be adapted to fit the capacity of the systems used. Data-rate constraints in tactical networks impose great challenges that have to be solved in order to fully deploy Web services supporting NEC.

Previously, we have performed experiments with Web services in a multi-national scenario at NATO Coalition Warrior Interoperability Demonstration (CWID) 2006. In these experiments, we showed that Web services could be used to exchange track data between nations. We used the object-oriented XML-version of the Command and Control Information Exchange Data Model (C2IEDM) from the Multilateral Interoperability Programme (MIP), and exchanged XML-based messages. Our experiments showed that the utilization of Web services in NEC is feasible, but it also revealed several challenges [2]. In those experiments Web services were used at the strategic level, where bandwidth is abundant (but even so, our uncompressed Web services traffic consumed a lot of the available bandwidth). In order to achieve full-fledged NEC the needs of tactical network users must be considered as well, and the experiments presented in this paper focus on those needs.

In our research following NATO CWID 2006 we have looked into measures for adapting Web services to tactical networks, and also given some specific suggestions for the use of C2IEDM in such networks [3]. One of the measures we suggested was that one should use data compression techniques in tactical networks to reduce bandwidth consumption. Recently, NATO has specified an alternative to C2IEDM for blue force tracking called NFFI. In this paper, we evaluate the gains of several different compression techniques applied to XML-encoded NFFI documents which we evaluated as part of our experiments at NATO CWID 2007.

XML, described in further detail below, is often considered the base standard for Web services, as most Web Service standards use the encoding and format rules defined in the XML standards.

The remainder of this paper is organized as follows: First, we give a short overview of XML and NFFI. Then, we proceed to discuss various compression techniques, and present our evaluation of some of the available methods. Finally, we conclude the paper by summarizing our results.

Extensible Markup Language (XML)

XML is a simple, very flexible text format derived from SGML (ISO 8879). There are multiple XML related standards, with the two most important being XML itself, and XML Schema. The latter standard is a description of a type of XML document, typically

expressed in terms of constraints on the structure and content of documents of that type, above and beyond the basic syntax constraints imposed by XML itself.

One of the benefits of using XML is that an XML document contains metadata, that is, data about the data that are present in the document. An XML document consists of data that are surrounded by 'tags'. Tags describe the data they enclose. A tag may have other tags inside it, which allows for a nested structure. Such tags can be standardized, which allows for the exchange and understanding of data in a standardized, machine-readable way. An XML document can be defined according to an XML Schema, which enables validation of XML documents according to rules defined in the schema. NFFI, which we discuss below, defines such an XML schema, allowing track information to be represented in a standardized way for exchange.

In its basic form, XML can be seen as a structured, human readable way to organize data. However, in certain cases it is more serviceable to sacrifice human readability for more efficient encoding and transfer. In such cases a binary representation of the XML document should be used, i.e. so-called binary or efficient XML. So far there is no standard for efficient XML, even though there is a proprietary solution available from the company Agile Delta that is called Efficient XML[1] (EFX). However, a W3C working group called *Efficient XML Interchange* (EXI) is in the process of standardizing an efficient XML format [11]. The objective of the EXI Working Group is to develop a specification for an encoding format that allows efficient interchange of the XML Information Set, and to illustrate effective processor implementations of that encoding. Earlier this year the group released a working draft [12]. It is worth noting that Agile Delta is actively participating in the EXI work, and are continually adapting their EFX product to conform to the working draft. Thus, in this paper we evaluate several of the currently available compression techniques that can be employed while awaiting a standard from the EXI group.

NATO Friendly Force Information (NFFI)

The object oriented part of C2IEDM is very complex, and thus a not very efficient way of exchanging the needed information [4]. NATO developed an alternative data exchange model to C2IEDM for blue force tracking for use in Afghanistan, the NATO Friendly Force Information (NFFI) Afghanistan Force Tracking System. However, the NFFI format can be translated to C2IEDM if needed, as the standard specifies a mapping of the fields in NFFI to fields in C2IEDM.

The current version of NFFI is 1.3 as published in draft STANAG 5527. NFFI consists of a message definition and message protocols. The message format is defined by an XML schema containing both mandatory and optional fields. The position data is a mandatory part of the document, and contains information about position (longitude, latitude, altitude) and velocity. Identification data is also a mandatory part, and contains

[1] Agile Delta's efficient XML (EFX): http://www.agiledelta.com/product_efx.html

information about the object's name and a 15 character text string from APP-6A/Mil STD 2525B. Thus, the position and identification data contain all the information needed to draw a symbol on a map. Furthermore, a status field contains the operational status of the object. All the other fields are optional, and may contain contact information, telephone numbers, etc. Currently the format is used only to follow friendly forces, but it could be extended to encompass all units in an area.

Reducing communication overhead

The scarceness of resources on the tactical level, such as bandwidth and power, means that it is vital to keep communication overhead at a minimum. There are different means one can employ to reduce this overhead by:

- Using compression techniques that retain all the information but represents it using fewer bits and bytes.
- Discard some information that is of lesser or no importance to the recipient.
- Changing the way information is represented (e.g. the XML schema) [3]. By using the NFFI schema the friendly force information was represented in a more compact way than with C2IEDM.

This paper focuses on the first of these three techniques, by evaluating compression methods suitable for use in tactical communication systems.

Compression

There are two types of compression; *lossless compression* and *lossy compression* [5]. Lossless compression is used on data that needs to retain its exact representation when it is decompressed. Lossy compression is used on data that can tolerate some loss such as audio, pictures and video. Lossy compression can, since it is allowed to modify the data, achieve higher compression rates than lossless compression. For documents (in our case XML documents), however, we need all the information to be intact so lossless compression should be used.

The lossless compression techniques we can employ here come in two flavors; we can use a generic technique that can compress any kind of data, or we can utilize the structure of XML documents and use an XML-conscious compression technique. There exist a lot of compression techniques of both kinds, and it is beyond the scope of this paper to discuss them all. Instead, we choose to focus on a few that are particularly promising for use in tactical communication networks. Two of these, namely XMLPPM[2] and GZIP[3], have proved versatile and efficient in other studies; see [13] and [14] for further details.

---

[2] XMLPPM is freely available at Sourceforge: http://xmlppm.sourceforge.net/

[3] The package java.util.zip: http://java.sun.com/j2se/1.3/docs/api/java/util/zip/package-summary.html

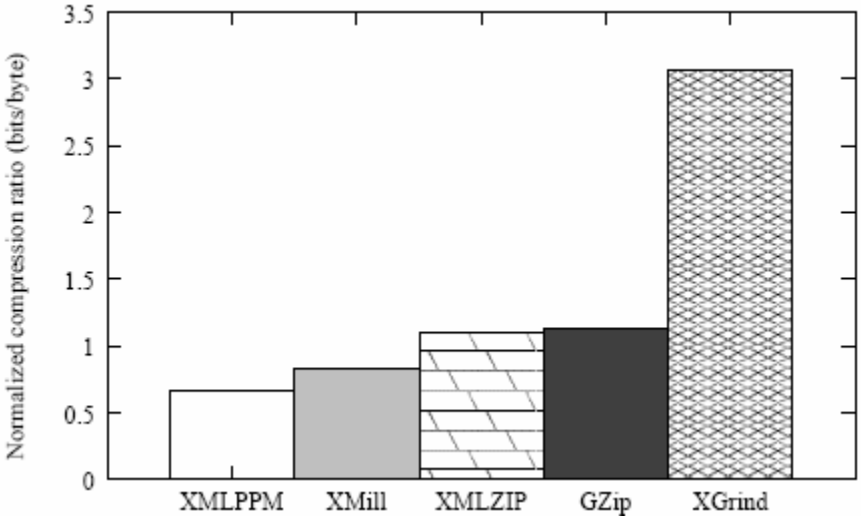

Figure 1 Average Compression Ratio(fetched from [13])

Figure 1 shows how a number of compression techniques performed in one of these studies. XMLPPM gave the best average compression ratios of the XML-conscious compression techniques, while GZIP was the best of the generic compression algorithms tested. Because these two techniques have been shown to be the best of their respective types, we chose to test these further in our evaluation.

Interoperability is a key challenge in NEC, so a standard based compression method is preferable. Since, as mentioned above, EFX is continually adapted to conform to the working drafts released by the EXI, we found it important to investigate EFX in the context of NFFI compression. The studies in [13] and [14] did not investigate EFX, since it is a rather new technique, but [14] mentioned its existence and that it should be evaluated in future work.

EFX can be used in one of two modes of operation; *generic* and *schema specific* compression. The generic option can compress any valid XML document without knowledge of the schema. The schema specific option needs to have access to the XML schema when it performs compression and decompression, thereby sacrificing generality for a very slight increase in compression rate. We used the generic option in our experiments enabling us to compare EFX directly to XMLPPM (which provides only non-schema specific XML compression). When evaluating the efficiency of the algorithms we focused on compression results and not resource usage during compression (memory and CPU usage). The reason for this is that for our intended use, i.e. in tactical networks, the bandwidth is the most limiting resource. Power consumption is also an issue when using battery powered communication equipment. However, in an earlier study [4], we have shown that the difference in computation time between the various techniques is in the millisecond range. Transmission of data also requires power, and by using a compression technique with a high compression ratio, we can reduce the transmission time by several seconds. It is reasonable to assume that the reduction in power consumption caused by reduced transmission time greatly outweighs the benefits of saving a few milliseconds when performing compression and decompression.

Filtering optional NFFI fields

NFFI has some mandatory and a lot of optional fields. We removed all optional fields and kept only the mandatory fields of each track. The tracks contained in the NFFI message would, as a result of this removal of optional information, be very uniform (e.g. all the same XML tags used in all tracks) and only the data differing. This made the NFFI message as compression friendly as possible, an important aspect for transmission in low bandwidth networks. For example, we could compress a message with optional fields removed to about 5% of its original size (when we used EFX with its built in compression and had 15 tracks or more in the message).

The motivation for removing the optional fields was that they were of no importance on the tactical level. In fact, the experimental tactical soldier system we used in our experiments, NORMANS[4], would not be able to use these fields anyway. The NORMANS visualization is simple, being designed to run on a Windows CE PDA. As such, the NFFI message contained more than enough information after the optional fields had been removed for the application to function (in fact, even some of the mandatory information in NFFI will not be visualized, since NORMANS makes a distinction only between friend and enemy units, and does not show the type of unit). For further details about our experiments with NORMANS at NATO CWID 2007, see [8].

Evaluation

When considering the results, it is important to remember that we used NFFI-tracks without optional fields, as described above. Each document contained a number of NFFI-tracks, ranging from one to 570, and the corresponding size of the documents ranging from 776 up to 393,066 bytes. We noted the size of each original document, as well as the size of each corresponding compressed document, using the compression methods we have identified as promising for use in tactical networks.

Some XML-conscious compression methods seek to retain the structure of the XML document during compression in order to make it possible to perform computations on the documents without having to decompress them first. The compression ratio achieved by these compression methods is lower than other types of compression due to this tradeoff. By performing a two step compression, where an XML-conscious technique is applied first, followed by a generic compression, the effects of this tradeoff can be reduced. Due to this we have, in addition to the stand-alone tests of each compression technique, also performed tests were we applied a generic compression on the results of the XML-conscious techniques.

In Table 1 we show document sizes, both before and after compression, for some of the documents used in the experiment. The results show that all compression techniques

[4] FFI develops concept, requirements and technology for the future network enabled soldier. An overview of the Norwegian Modular Network Soldier (NORMANS) is given in [9], and the C2I system is presented in [10].

perform well for large documents, typically with hundreds of NFFI-tracks in them. XMLPPM performs better than GZIP for small documents, while the opposite is true for documents containing many tracks. EFX alone does not perform very well, but combining it a generic compression technique improves its performance. AgileDelta's own ZIP-variant does this better than GZIP. Another interesting observation is that XMLPPM should not be combined with GZIP, as this resulted in *increased* document sizes (rightmost column of the table) compared to using either of the two techniques separately.

#tracks	original size	EFX	XMLPPM	GZIP	EFX + GZIP	EFX own ZIP	XMLPPM + GZIP
1	776	310	275	367	465	286	444
2	1429	353	315	392	522	313	496
3	2082	396	340	412	546	339	525
4	2738	460	385	460	611	372	581
5	3394	514	411	484	664	384	614
6	4041	560	430	500	715	397	638
7	4697	614	449	514	774	409	665
8	5353	668	468	529	839	417	686
9	6000	714	485	540	893	423	706
10	6656	768	502	554	915	431	730
11	7304	824	540	593	971	463	777
12	7960	879	558	615	994	472	797
13	8607	924	576	632	1017	479	818
14	9263	986	597	647	1074	489	852
15	9915	1028	620	673	1103	504	884
...							
570	393066	26396	12931	11691	14447	7203	16368

Table 1 Size (in bytes) of different NFFI-documents, before and after compression

It should also be noted that Table 1 only shows the payload that is being transmitted. After having created the compressed document, a packet header is needed for transmission over the network, which increases the message size with a fixed number of bytes.

Figure 2 shows a graphic representation of how the four best techniques performed over the entire data set. All the documents are reduced to less than ten percent of the original size for large documents, illustrating how important compression is when bandwidth is scarce. Regardless of compression method, none of the compressed documents were larger than 26 kB.

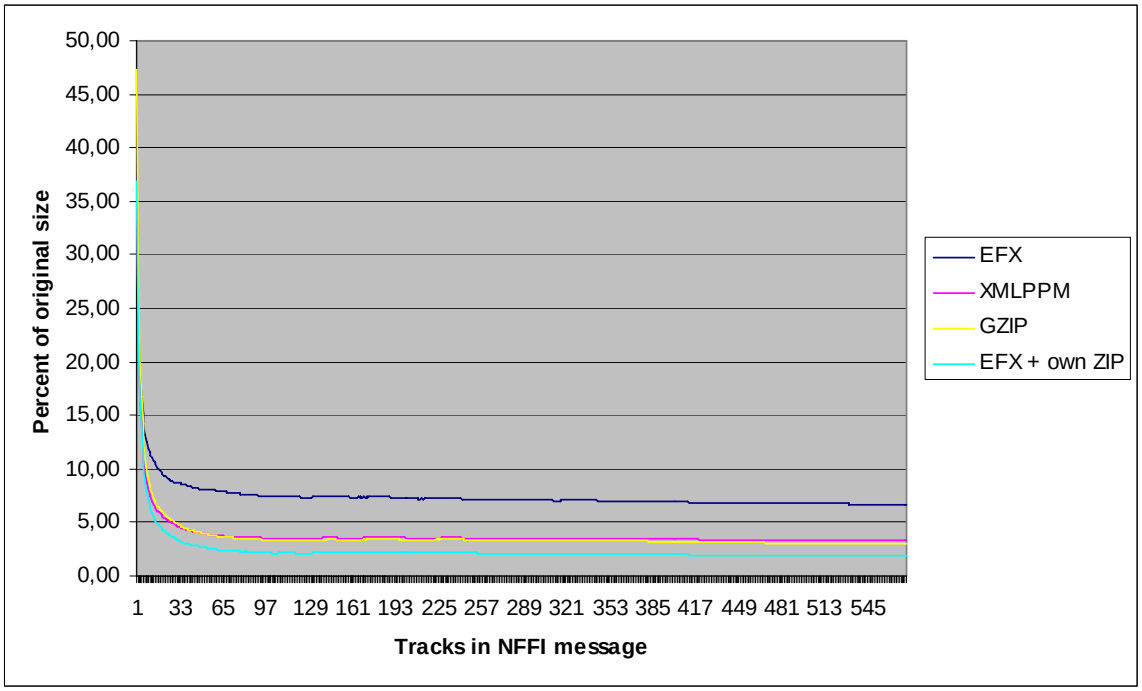

Figure 2 Percent of original size of the XML document after compression compared with the number of tracks in the NFFI message

Figure 2 also shows that when the number of tracks in the documents reaches a certain level, EFX combined with its own built-in ZIP always performs best. However, when there are fewer tracks to report, the differences between the compression techniques are larger. In a disadvantaged grid, every byte saved can have a noticeable impact on the transmission time, and it is therefore important to take the individual differences between the techniques into account. Figure 3 shows the results for the 15 smallest documents. For the smallest documents with only one or two tracks, XMLPPM does the best job. EFX alone does not perform very well, but combining it with Agile Delta's own ZIP-variant makes it out-perform all the other techniques as soon as the documents start growing in size.

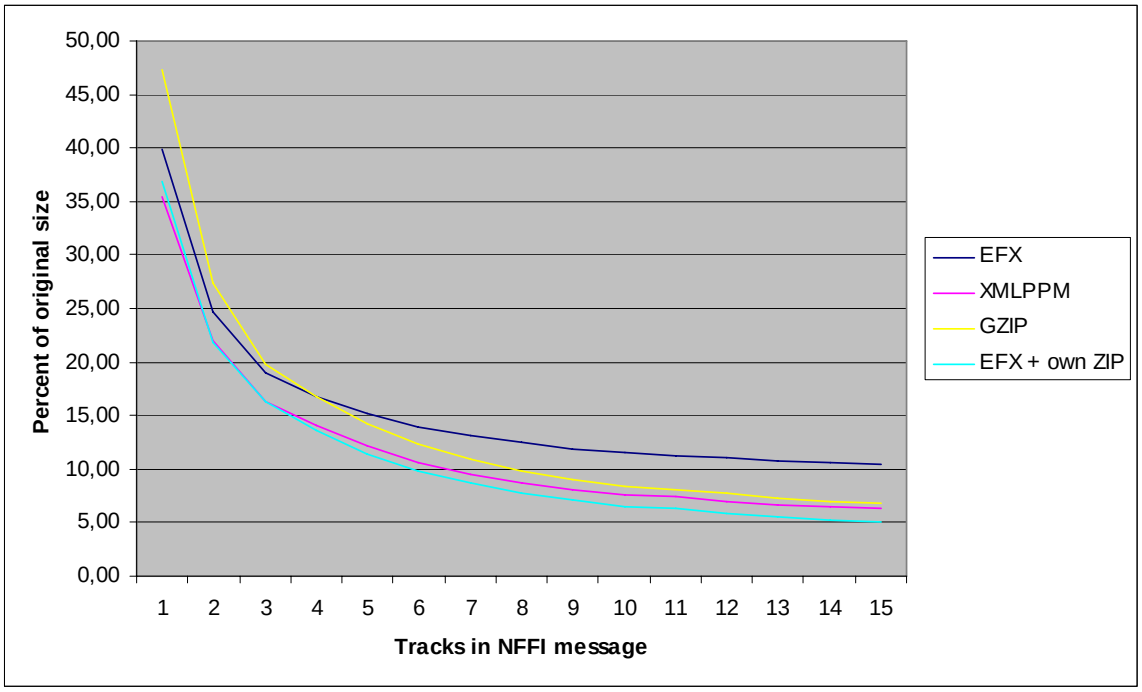

Figure 3 Percent of original size of the XML document after compression compared with the number of tracks in the NFFI message

For all four compression methods, the compression ratio continues to increase with increasing number of NFFI-tracks in the documents. The best ratio we measured was using EFX with Agile Delta's ZIP on the largest document with 570 NFFI-tracks, where we achieved a compressed document size of less than 2% of the original size, as is shown in Table 2.

#tracks	EFX	XMLPPM	GZIP	EFX + GZIP	EFX own ZIP	XMLPPM + GZIP
1	39.95	35.44	47.29	59.92	36.86	57.22
10	11.54	7.54	8.32	13.75	6.48	10.97
50	8.11	3.98	4.01	6.30	2.69	5.31
100	7.42	3.52	3.34	4.64	2.19	4.58
200	7.24	3.56	3.38	4.73	2.19	4.55
300	7.10	3.52	3.36	4.46	2.11	4.48
400	6.92	3.43	3.22	4.07	2.00	4.36
500	6.74	3.34	3.04	3.84	1.89	4.24
570	6.72	3.29	2.97	3.68	1.83	4.16

Table 2 Size (in percent of the original, uncompressed XML document) of different NFFI-documents after compression

Compression ratio can be expressed in a number of different way, and we have used the formula from [13] (see Figure 4) to calculate the compression ratio in terms of number of bits per byte. This measurement expresses the number of bits after compression that is needed to represent each byte in the uncompressed data format. This in turn means that with a compression ratio of 1 bit/byte, the compressed document contains one bit for each byte in the original document, in effect reducing the size of the document to 1/8th of the original size.

$$CR_1 = \frac{sizeof(compressed\ file) \times 8}{sizeof(original\ file)}\ bits/byte$$

Figure 4 Compression ratio in terms of bits/byte (fetched from [13]).

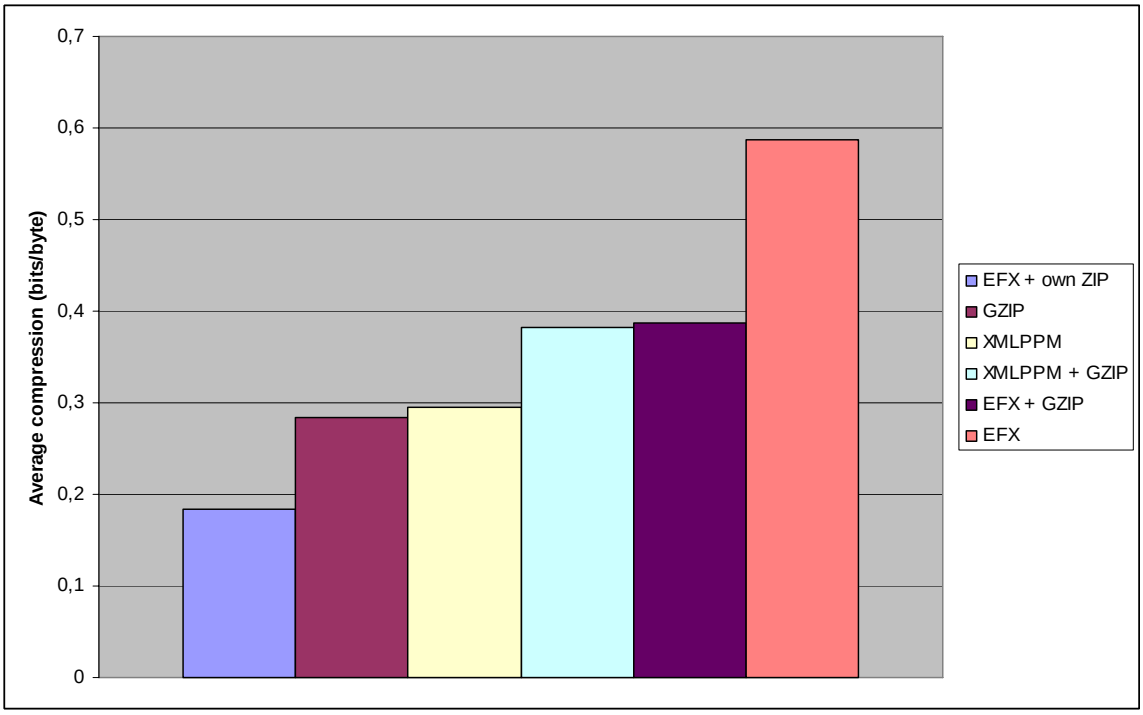

Figure 5 Average compression ratio

The exact compression ratio one can achieve will depend on the original document. How well the various compression techniques compress NFFI formatted data, can be expressed using the average compression ratio in bits/byte, shown in Figure 5. Comparing these results with those in Figure 1, we see that the NFFI XML documents achieve a far better average compression ratio than that. This is easy to explain; XML documents are highly regular in structure and compress well [13]. We see that the solution achieving the best results is EFX with its built in proprietary ZIP enabled. XMLPPM and GZIP are comparable, with GZIP being marginally better than XMLPPM in this figure. However, when we look at Table 1, we see that XMLPPM is better than GZIP when there are few

tracks in the NFFI message. In fact, when we investigated this we found that the inflection point was at 54 tracks in our experiments. Up to and including 54 tracks in the NFFI message XMLPPM performed best, after that GZIP was slightly better. The study from [13] found that XMLPPM was slightly better than GZIP (see Figure 1), because they used small documents in their study (all documents were below 40 Kbyte except for one which was 100 Kbyte) for which XMLPPM performs better.

XMLPPM should not be used together with GZIP, as this yields much worse results than using either GZIP or XMLPPM alone. Pure EFX yields the worst compression, but this is expected since EFX is a binary XML format and not compression as such. Thus, there are further gains by using GZIP, or even better, Agile Delta's own ZIP, with EFX.

Table 3 shows the theoretical minimum time to transmit an NFFI message over a 2.4 Kbit/sec link. For simplicity we calculate the transmission time of the message only, thus finding the theoretical minimum transmission time. In a real network the actual transmission time would be higher, since there would be transport protocol headers added to the message. How much more delay this incurs would depend on the transport protocol chosen and other packets traversing the same link. Here we assume that the link is available entirely to our disposal.

#tracks	Uncompressed	EFX	XMLPPM	GZIP	EFX + GZIP	EFX + own ZIP	XMLPPM + GZIP
1	2.59	1.03	0.92	1.22	1.55	0.95	1.48
10	22.19	2.56	1.67	1.85	3.05	1.44	2.43
50	109.22	8.86	4.34	4.38	6.88	2.94	5.80
100	217.98	16.17	7.67	7.29	10.11	4.78	9.98
200	451.53	32.71	16.08	15.26	21.37	9.90	20.53
300	687.24	48.78	24.19	23.06	30.64	14.50	30.76
400	927.76	64.25	31.86	29.88	37.79	18.55	40.46
500	1157.84	78.05	38.71	35.20	44.44	21.83	49.07
570	1310.22	87.99	43.10	38.97	48.16	24.01	54.56

Table 3 Theoretical minimum time, in seconds, to transmit an NFFI message over a 2.4 Kbit/sec link.

As the table shows, we can save a lot of time (and thus bandwidth) by employing compression. Whereas it would be infeasible to send much more than 10 uncompressed tracks in a 30 second interval, one could easily send ten times that provided compression was used. In fact, provided one has a dedicated channel of 2.4 Kbit/sec, it should be feasible to send all 570 tracks in the allotted 30 second interval provided EFX with its proprietary ZIP compression is used.

In practice, however, this turned out to be a different story. In the NATO CWID trial we only had 24 tracks to disseminate from the HQ to the tactical unit. The NFFI messages

were distributed using a push Web service, so each compressed NFFI message was wrapped in uncompressed SOAP headers, something which added to the number of bytes needing to be transmitted over the network. Table 4 illustrates this. We see that for the trial NFFI message we achieved an NFFI message compression rate of 0.61, which was worse than the results achieved with the NFFI messages evaluated in Figure 5. Adding SOAP headers further diminishes the compression ratio, yielding 1. Still, we see that even considering this NFFI is very compression friendly when comparing with the results from another study of generic XML documents in Figure 1.

Original NFFI message size	Compressed NFFI message size	Message size on the wire, i.e. with SOAP headers	Compression rate of NFFI message in bits/byte	Compression rate when considering SOAP headers
22072 bytes	1679 bytes	2291 bytes	0.61	1

Table 4 NATO CWID NFFI message with 24 tracks

At NATO CWID we used a link emulator[5] to achieve a link speed of 2.4 Kbit/second, which is representative of a typical speech channel. We sent track updates every 30 seconds. By using compression we were able to send much more data over the link than if we had just used plain XML. However, we did not achieve as good results as the theoretical minimum time from Table 3. This was not expected either, since the theoretical minimum transmission time takes neither application level headers (i.e. SOAP) nor transport level headers into account. Furthermore, in Table 3 it is assumed that the link is used exclusively, whereas in the actual trial the link was being used for two way communication. Thus, for the message presented in Table 4 we achieved an average transmission time of 14.9 seconds, with a standard deviation of 4.5 seconds.

Conclusion

As we have shown in this paper, there are significant gains when using compression of XML data. The NFFI documents containing track information that we compressed had their size reduced to such an extent that using XML encoded NFFI at the tactical level should become feasible, even when one considers the bandwidth constraints.

The main issue here is that compression of some form should be employed, but which algorithm one chooses is of lesser importance as they all give a significant reduction in document size. The choice of compression technique should be made collectively, and must be agreed upon in NATO to facilitate interoperability. If Web services indeed become the fundament for realizing NEC, then using the emerging standard for XML compression as defined by the EXI working group would probably be a good idea. One

[5] We used the NIST Net network emulator package for emulating a tactical link in our experiments. The NIST Net software is freely available, and can be downloaded from "http://www-x.antd.nist.gov/nistnet/". We used version 3.0a with SuSe Linux 10.0.

will then eventually have standard based COTS products available. An extra benefit gained from choosing a binary XML format compared to just compressing the XML document with for example GZIP, is that the computer can work with the binary format directly, with no need to decompress first. This saves both computational time and memory requirements.

References

[1] Network Centric Operations Industry Consortium (NCOIC) http://www.ncoic.org/home

[2] Haakseth, R., Hadzic D., Lund, K., Eggen, A., Rasmussen, R. E., "Experiences from implementing dynamic and secure Web services", 11th Coalition Command and Control in the Networked Era (ICCRTS), September 2006, Cambridge, UK.

[3] Hafsøe, Trude, Johnsen, Frank T., Lund, Ketil, and Eggen, Anders "Adapting Web Services for Limited Bandwidth Tactical Networks" 12th International Command and Control Research and Technology Symposium, (ICCRTS), June 2007, Newport, RI, USA.

[4] Hadzic, D. et al, "Web Services in networks with limited data rate" (in Norwegian), FFI-Report 2006/03886, ISBN 978-82-464-1049-4

[5] Salomon, David. Springer 2000. "Data Compression – The Complete Reference, 2nd edition", ISBN 0-387-95045-1

[6] Erl, Thomas. Prentice hall 2005. "Service-Oriented Architecture – Concepts, Technology, and Design", ISBN 0-13-185858-0

[7] Erl, Thomas. Prentice hall 2004. "Service-Oriented Architecture – A Field Guide to Integrating XML and Web Services", ISBN 0-13-142898-5

[8] Johnsen, Frank T., Hafsøe Trude, and Lund, Ketil, "NATO CWID 2007 disadvantaged grids experiments", FFI-Notat 2007/02063

[9] R. Lausund, and S. Martini, "Norwegian Modular Network Soldier (NORMANS)", FFI Facts, November 2006, http://www.mil.no/multimedia/archive/00086/FFI-FACTS-NORMANS-EN_86447a.pdf

[10] J. Flathagen and L. Olsen, "NORMANS KKI" (in Norwegian), FFI Facts, November 2006, http://www.mil.no/multimedia/archive/00086/Faktark-NORMANS-KKI-_86445a.pdf

[11] Efficient XML Interchange Working Group http://www.w3.org/XML/EXI/

[12] Efficient XML Interchange (EXI) Format 1.0, W3C Working Draft July 2007 http://www.w3.org/TR/2007/WD-exi-20070716/

[13] W. Ng, W.-Y. Lam, and J. Cheng, "Comparative Analysis of XML Compression Technologies", World Wide Web 9(1), pages 5-33, Kluwer Academic Publishers, March 2006.

[14] C.J. Augeri, B.E. Mullins, L.C. Baird, D.A. Bulutoglu, and R.O. Baldwin, “An Analysis of XML Compression Efficiency”, In Proceedings of the 2007 Workshop on Experimental Computer Science (ExpCS '07), 2007.