Towards an IP-Based Alert Message Delivery System

Pekka Sillberg
Tampere University of Technology (TUT), Pori
Finland
pekka.sillberg@tut.fi

Petri Rantanen
Tampere University of Technology (TUT), Pori
Finland
petri.rantanen@tut.fi

Mika Saari
Tampere University of Technology (TUT), Pori
Finland
mika.saari@tut.fi

Jari Leppäniemi
Tampere University of Technology (TUT), Pori
Finland
jari.leppaniemi@tut.fi

Jari Soini
Tampere University of Technology (TUT), Pori
Finland
jari.o.soini@tut.fi

Hannu Jaakkola
Tampere University of Technology (TUT), Pori
Finland
hannu.jaakkola@tut.fi

ABSTRACT

Advancements in technology have provided new opportunities for the delivery of emergency messages. However, some of the issues concerning data security and technical solutions are quite different from the problems of the traditional means of communication. The Internet poses its own set of challenges. This paper presents a few emergency messaging system proposals made by other researchers and also introduces a new proposition put forward by the authors of this paper. This will demonstrate how to use client-server architecture to deliver emergency alert messages in IP-based networks. The proposed system uses Atom feeds to deliver alert messages and also provides a feedback channel for client data. In this scenario clients could have any kind of device from mobile terminals to desktop computers.

Keywords

IP-based alert message, Mobile Emergency Announcement, Common Alerting Protocol, Atom feed, Information systems

INTRODUCTION

Mobile terminals have become an essential part of everyday life. The penetration level of mobile phones is approaching 100% in well-developed economies and even in less developed economies the number of phones is rising very quickly. From this aspect, the use of mobile phones for delivering emergency information is justified. In this paper the objective is how to utilize mobile technology to supply disaster information and especially communication by mobile phone. We have worked with a communication system that delivers alert messages. One main research area is how to send alert messages to mobile devices. This paper describes the communication mechanism which is needed to send and receive alert messages in mobile devices.

The major challenges of emergency management are to reduce the human and economic costs of the emergency. One of the most important elements to achieve this goal is to make the general public aware of and prepare them for an upcoming emergency situation. According to the Organization for the Advancement of Structured Information Standards (OASIS, 2005), the emergency management cycle consists of the four main phases shown in Figure 1.

Figure 1. The Disaster and Emergency cycle (Adapted from OASIS, 2005; Alexander, 2002)

The purpose of the Prevention and Mitigation phase is to lower the possibility of the disaster – or emergency – occurring and/or to minimize the damage caused. Actions done for this phase are generally long-term processes. The Preparation phase typically consists of actions taken when a disaster is anticipated or impending in order to ensure a rapid and more effective response. Actions taken during the Response phase are aimed at saving human life, the protection of assets, the supply of vital goods and services, and the protection of the environment. The Response phase begins as the disaster/emergency strikes and continues until the Recovery phase takes over. Recovery is the process by which communities return to a normal level of functioning. (OASIS, 2005)

One of the most important features of an emergency alert system – regardless of what medium/media it uses as its transfer layer – should be its ability to deliver the emergency alert effectively to as many of the people under threat as possible. According to a study made by Verma and Verma (2005), a good emergency alert system should have the properties of locality, automated operation, non-intrusiveness, spontaneity, ubiquity and support for second languages. These properties are not tied only to emergency alert systems working over the Internet, thus they may be applied to any emergency alert system. Locality means that the emergency alert should be available to the general population that is affected by the emergency. The automated operation of the emergency alert system means that it should be able to switch to alert mode from normal mode and vice versa without the need for manual intervention. Non-intrusiveness and spontaneity are especially aimed at emergency alert systems but may also be applied to "normal" emergency alert systems. Non-intrusiveness means that the user's activities should not be disrupted by the alert system. When the alert system is spontaneous, it can deliver and show alerts to the user without the user needing to do any predetermined action manually. Ubiquity by definition means "being everywhere," thus in an alert system context it means that alerts should be provided to everyone affected, and should not miss out any user in the affected area. The support for a second language simply means that the emergency alert system should be able to provide alerts in another or many other languages.

SSMC/DDKM - The Project

The starting point of this study is the ongoing SSMC/DDKM (Seamless Services and Mobile Connectivity in Distributed Disaster Knowledge Management) research project, coordinated by the Tampere University of Technology (TUT). The general goal of this two-year research project (Soini, Leppäniemi and Jaakkola, 2008) is to study and develop the methods, processes and technologies to support improved knowledge management in disasters and accidents. The ability to build a reliable situational model based on the history and current knowledge of the situation and on good practices concerning comparable accidents/disasters is important in minimizing the consequences of an accident. The project is funded 70% by Tekes (the Finnish Funding Agency for Technology and Innovation) and 30% by a consortium consisting of two Finnish ICT companies and the Finnish Emergency Response Centre Agency. The SSMC/DDKM project includes two dimensions: *international* and *national.* The project is based on *international research* co-operation between several organizations. The leading forces are NICT (National Research Institute on Information and Communication Technology) and Keio University (SFC) in Japan.

The Japanese project has been given government financing for a 5-year period. The goal of this international part is to develop a distributed disaster knowledge management system, which supports the connectivity of separate knowledge sources (NICT, 2007a, 2007b). In the *national dimension* the project focuses on investigating current collaboration – taking into account both technical and social aspects – in accident and disaster situations between the Finnish authorities who are responsible for producing, disseminating and utilizing information in these situations.

The structure of this paper is the following: first, the background of the subject is introduced in the section entitled "Emergency alert systems on the Internet." The standards used are also mentioned. In the next section "Towards an IP-based alert Message Delivery System," the basic construction of our proposed emergency message alert system is described. At the beginning we describe the protocol used in client - server communication. The section entitled "Communication flow between server and client" includes two cases for messaging between client and server: a basic case and an extended case. The basic case is demonstrated by a sequence diagram of message transportation. The extended case demonstrates the use of a feedback channel. Finally the study is summarized in the "Conclusion".

EMERGENCY ALERT SYSTEMS ON THE INTERNET

What makes IP-based (such as Internet) emergency alert systems plausible are the many possibilities the Internet has to offer. For example more and more of the mobile devices sold nowadays have the option of connecting to the Internet using wireless networks – or other high bandwidth networks – and can also run programs made to support emergency messaging. This can enable services where one could report immediately from the disaster site thus giving first hand knowledge for use in emergency management. The service provided could easily be expanded to giving alerts, warnings or instructions before and after the emergency incident. IP-based alerting systems also allow other devices accessing the network to use the service, such as desktop computers and personal digital assistants (PDA). This is a great advantage over emergency alert systems designed solely for mobile phones (for example SMS messages).

When sending or receiving information over the Internet or any other IP-based network architecture, there are basically two possible implementations: The first one is to make a custom – free or proprietary – protocol and use it over the existing network infrastructure – and the second option is to use a standardized protocol. In many cases the first kind of implementation (proprietary protocol) may be the faster way of transmitting information and more optimized regarding bandwidth usage. However, the custom structure of the chosen protocol could cause problems when communicating with third party software or when interfacing with other systems. Therefore, especially with emergency messaging, this is not a desirable outcome. The interoperability with multiple systems, devices and software should be one of the prime goals of development.

To overcome the interoperability problems in emergency information transfer the second option is preferable, i.e. a commonly known and standardized protocol. For emergency messages the Common Alerting Protocol (CAP) of the Organization for the Advancement of Structured Information Standards (OASIS, 2008) is a good choice because of its standardized format (CAP v1.1, 2005; Botterell, 2006) and completely free and open nature. CAP has already been used in many official systems and especially in Atom feeds (CAP Cookbook, 2008; RFC 4287, 2005). CAP itself is a part of a larger protocol family known as Emergency Data Exchange Language (EDXL). EDXL has standardized means, for example, for sending messages using eXtensible Markup Language (XML) with and without the XML-based SOAP protocol (EDXL-DE v1.0, 2006). In conclusion, using standard means and protocols is the second and recommended implementation option. Both the Atom feed-based approach and the SOAP protocol are used in the emergency message system proposition presented in this paper. Next we will examine a few other systems that have also been developed for delivering emergency alerts.

There have been many previous propositions that use CAP or Atom or both for sending and receiving emergency messages. X-Capatom (IBM developerWorks, 2007) is quite similar to our proposal and there are also similarities with Art Botterell's Advanced EAS Relay Network (AERN) proposition's "Secure server" approach (Botterell, 2003). Botterell (2003) describes many other means for emergency message delivery and primarily these are meant to improve the United States' existing Emergency Message System (EAS), but the same ideas can be applied to other systems. When referring to AERN in this paper, the "Secure server" approach is meant. The AERN architecture is free and open for use for anyone, as is X-Capatom.

Both X-Capatom and AERN represent a system which has one or multiple secure servers and the content of these servers is updated by some authorized party. Since only selected individuals or parties can add or modify content on the server, the information can be solely trusted as long as the server is not compromised. This is the basic idea of the approach. X-Capatom does not especially mention the "Secure server" approach by name, but it can be

considered to belong to this category. AERN describes the architecture more and does not go too deeply into technical, code-level details.

X-Capatom in turn concentrates more on describing the basic functions and technical details of the system. IBM's X-Capatom pages (IBM developerWorks, 2007) even have a Java™ based reference implementation. As the name implies, X-Capatom uses a combination of Atom feeds and CAP to deliver messages. This approach has the major advantage of being able to deliver messages to many programs which are not originally meant to read emergency message information. Atom feeds can be read using many of the basic Really Simple Syndication (RSS) readers and by normal web browsers. It is also possible to make customized third party software that interprets the feeds in some specific and predefined way and displays the information to end users, but this is optional. X-Capatom uses Atom Publishing Protocol (RFC 5023, 2007) for modifying, deleting and adding new information to Atom feeds. The reference implementation does not have any kind of user authorization or user control even though the X-Capatom recommends adding such features when using the system. The feeds themselves are available for everyone to read.

In summary, the techniques and methods presented earlier in this section share many common elements and basic ideas with the emergency messaging system proposed by the authors of this paper. In next section the new proposal and implementation example are described in more detail.

TOWARDS AN IP-BASED ALERT MESSAGE DELIVERY SYSTEM

The system that we are introducing in this paper is fundamentally based on the integration of two different messaging channels working together in a single system. The diagram (Figure 2) below illustrates the basic operation of the system. In our example the *feed channel* uses Atom feeds as a transfer medium. The feeds are compiled by the server using data it has collected from various sources. These information sources could be basically any type. These feeds contain the elements of Atom and CAP – or any other XML-based format – elements with the necessary processing information for use by the client software. Feeds can be used as standalone feeds in any RSS/Atom reader and can still offer sufficient information about the incident for the user. In cases where the user does have an enhanced reader program, the processing information in the feed can offer extended features. For example, one feature could be an automatically initiated alert if a predefined threshold value is exceeded. Another example could be opening a map program showing the location of the coordinates included in the feed.

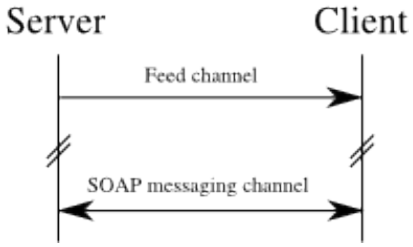

Figure 2. Simple communication sequence

The Atom feeds, i.e. the alert messages, should be constructed in such a way that every reader (normal RSS/Atom readers or enhanced clients) is capable of understanding all the Atom-specific elements of the feeds and seeing the crucial alert information. This means that processing information meant for enhanced clients should only include meta-information of the information already provided in the common part of the Atom feed. In this way different kinds of clients/devices receive the same information, and only the layout and the presentation vary. In case of the previously mentioned example of opening the map program, if the client device cannot show maps, the user can still see the coordinates and verbal descriptions of the location. This kind of difference in the visual presentation of the information may affect the intelligibility of the alert messages (e.g. it might be difficult for the user to figure out the precise geographical location of the coordinates).

In our example the extended messaging channel is using SOAP encapsulated messages. This channel allows users to initiate the data transfer between the server and client as well as the other way around. In other words, use of this kind of approach enables a feedback channel as well as a wide range of services and features. But this approach is

not completely free of problems and/or drawbacks. For instance, the server needs to know where its users are (IP addresses etc.) and some network applications such as Network Address Translators (NAT) and firewalls may interfere with connection attempts made by the server. Also, client devices running software for listening to a server, in order to make a connection, may make them vulnerable to security threats.

The emergency message delivery system proposal presented in this paper follows the ”Secure server” approach and the ”Client software” approach (Verma and Verma, 2005). The feed-based system was chosen because it enables normal users, that is, users who are not known to the system in any way to read emergency messages using whatever software they prefer. Atom feeds were chosen purely because of Atom's status as the Internet Engineering Task Force (IETF) standard (RFC 4287, 2005). RSS feeds may be used more often, but there is no common agreement about what RSS feeds can or cannot include. In some cases this kind of freedom can be a good thing, but when a standardized format is required it is not. Additionally, unlike Atom, RSS does not state what to do with XML elements and this is left to the XML processing programs to decide. Atom especially defines that unknown elements, fields or attributes should *not* be processed in anyway. This means that any software working as a client cannot start guessing what to do with this kind of unknown data. In emergency messaging it is crucial that unknown messages are not interpreted wrongfully or “guessed.” All client – and of course server-side – programs should process the messages similarly and notify the user in the case of unclear messages. This can be done with RSS, but it is *not required* by any standard. It should be noted that nothing guarantees that programs using Atom do not just ignore the unknown data and continue normally, but in well-made programs this kind of behavior is not desirable. Another gain in using Atom compared to RSS is its ability to clearly state whether the feed content is just a collection of elements from some other message (for example a CAP message) or the complete original message in the form of a link pointing to another destination (Westfall, 2007).

System overview

The picture below (Figure 3) shows a generalized overview of our proposal i.e. the system described in this paper. In essence, the purpose of the system is to relay messages. The system is used to relay emergency alert information, but the same basic model could be used with any kind of information

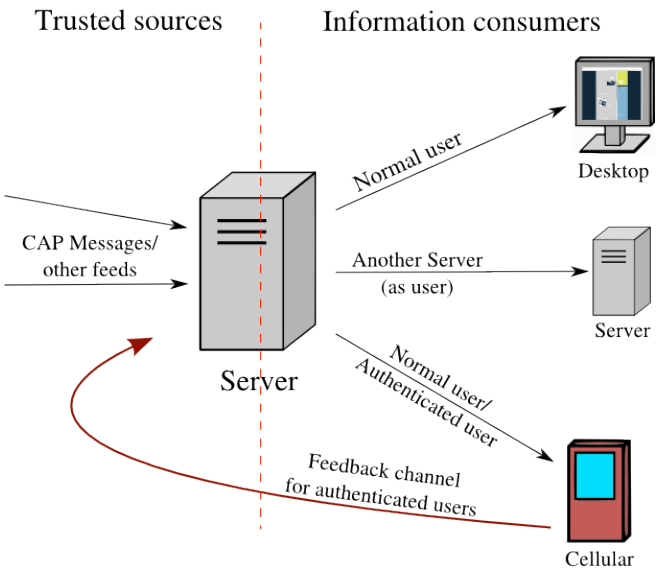

Figure 3. Overview of system

The “Server” is the connecting part between two sides, the trusted sources – which could be thought of as the providers of the information – and the information consumers. The major difference between this model and the X-Capatom is that in the basic usage scenario there is no need for user management or authorization. This is because no information is needed about the “normal users” reading the feeds – this is also true with X-Capatom. The providers of the information should also be trusted automatically. These trusted sources could be anything from CAP messages offered by official sources or other Atom feeds – RSS feeds are also possible despite their

nonstandard nature. This allows some servers to work as information providers for other servers, or in other words, they could behave like a client of a server.

All information coming from trusted sources should be transmitted over secure communication lines to guarantee that the messages are authentic. The secure communication could be anything from Virtual Private Networks (VPN) to Secure Sockets Layer / Transport Layer Security (SSL/TLS) using certificates (for example X.509 certificates). It is also possible to use certificates and unencrypted data when only message integrity is required. SSL/TLS is also a good choice for message authentication checks in Atom feeds because it requires very little changes in the operation of the web server, though it should be noted that using encryption causes extra workload on the server side. Unencrypted data and X.509 certificates are often enough in feeds because the information is meant to be read by anyone and there is no risk of information leaks to unwanted parties or individuals. If the system is used to relay confidential information that often means that the information is meant for a relatively small group of individuals and in that case the extra workload caused by data encryption is smaller.

The figure 3 (above) also shows the possibility of an authenticated user. Because the server can receive messages from the trusted sources side over an SSL/TLS connection, the same functionality enables it to receive messages from "Normal users." These users must be known to the server and authenticated before any information can be received. Connection to authenticated users can be thought of as a secondary means of information retrieval and can be used for example to transmit real-time, on-location data between different authorities (for example between the police and fire departments). Knowing certain clients and collecting information about them – such as IP addresses and/or geographic locations – also makes it possible to connect to these clients directly. In this paper's emergency message system the direct connection to clients and the feed-based information relay are separated to different usage cases and these cases are described below (Figure 4) in more detail. Direct connection includes both sending information to clients and receiving information from them.

Communication flow between server and client

The differences to existing systems are pointed out using a couple of example cases (*basic* and *extended)*. In principle most World Wide Web (WWW) pages (such as Web portals and news pages) use the basic approach where the client is solely responsible for fetching (i.e. pulling) the information. This is also how many other messaging services work. Extended cases use the push approach for transmitting information from sender to receiver, that is from client to server or from server to client.

Communication between the server and the client is quite simple and straightforward. The diagram below (Figure 4) describes the most common use cases offered by the system as a whole.

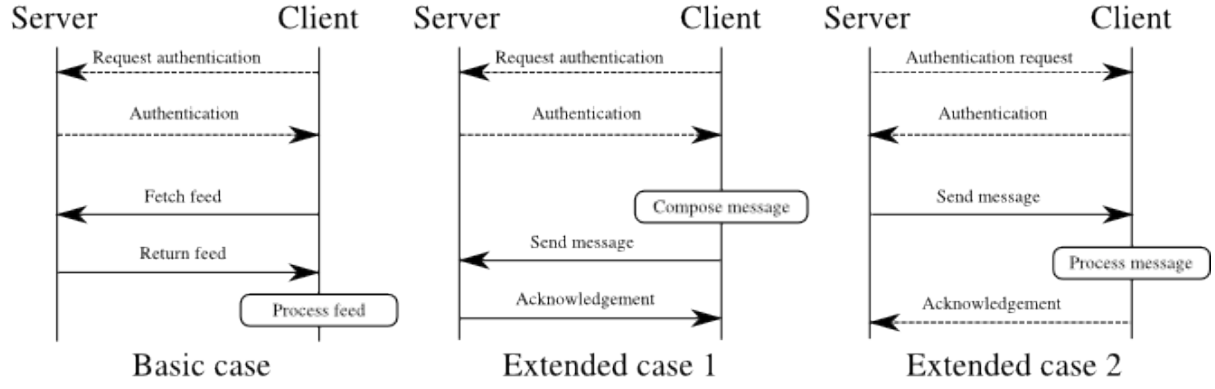

Figure 4. Sequence diagrams of most common use cases

In every case there is an optional authentication phase. During this phase both server and client may be authenticated and they may form a secure connection if necessary. In cases where the client makes the initiating connection to the server using an SSL or TLS connection, the server side can be verified by the certificate it uses. When the server initiates the connection, both sides may need to verify each other. For example, the challenge-response-authentication method may be used. Phases or actions with dashed lines can be skipped if they are not needed or if it

has been decided not to use them. Even though the authentication step can be skipped, it may not be advisable due to possible security threats.

Basic case

This is the most basic step (Figure 4 above, leftmost sequence diagram) that can be done in any reader that supports Atom feeds. Retrieval of the feed is done either by giving the direct access to a requested feed or by searching for a suitable feed from the available feeds. Firstly, the authentication step is made as described in the previous section. Then the feed will be retrieved to the client device. From that point the software decides how the feed is displayed. If the client software is a basic RSS reader, it will show the Atom content of the feed as it should. When a user is reading the feed with software designed to interpret the enhanced feed, not only will it show the content of the message(s), but also perform special features based on the extra information included in the feed.

Extended cases

These two cases (Figure 4 above, middle and rightmost sequence diagram) illustrate the use of the extended messaging channel. As a reminder, users of this messaging channel must be registered and authenticated. Therefore authenticated users are referred to as "users" below unless otherwise stated. In case number one (Figure 4, center sequence diagram), the client software initiates the connection and in the second case (Figure 4, rightmost sequence diagram) the server is the initiating side. In both cases a message, a reply to some earlier message, a command etc. can be delivered to the receiver. An example of using case 2 could be the following: the server receives a message that is a very important alert and certain user(s) must get it immediately. As a result the server opens the messaging channel to these clients and sends the alert to them. An example of case 1: the client software notices that the IP address bound to the device has changed for some reason. The client software then opens the messaging channel to the server and updates the user information to match the situation (this is just a simplified example and does not describe the steps that may need user intervention such as asking permission to open the data connection).

As both messaging channels are integrated in one system, they may seamlessly use information from/about the user, available feeds and other information to enable various emergency alerting services. Although the focus has been on emergency alerts and the like, the system is not limited to them alone. With proper extensions the system may be used for the delivery of traffic information, weather information and so on. Basically, the emergency alerting system could be just one small part of a larger information delivery system.

We believe that there are two promising features in the system introduced here. The first is the combination of two communication channels that allows a "simple" messaging service using one of the channels and the other channel enables the feedback from user to server. This also allows for additional possibilities such as executing certain features on the client device. The second feature is that the new system may be run virtually on any platform or system that is able to connect to the Internet. The next step we are taking in the research will be the implementation of the system as a proof-of-concept styled demo system. The demo is anticipated to be ready in summer 2009.

CONCLUSION

Mobile terminals have become an essential part of everyday life. These have provided new opportunities for the delivery of emergency messages using mobile phones. In this paper we have shown how to utilize mobile technology to supply disaster information to both mobile terminals and desktop computers. We have also presented a few emergency messaging system proposals made by other researchers and also introduced a new proposition of our own. We have given a brief survey of the delivery of emergency alerts using the Internet. We have outlined a structure for communication between server and client. This is the most important part of a messaging system, when further research and demo application development are included in the scope. The proposed system uses Atom feeds to deliver alert messages, which makes it possible for clients to use any kind of device, from mobile terminals to desktop computers. The next step in our research will be to develop a demo of both the server and the client application.

REFERENCES

1. Alexander, D. (2002) Principles of Emergency Planning and Management, Terra Publishing.
2. Botterell, A. (2003) An Advanced EAS Relay Network Using the Common Alerting Protocol, White Paper, http://www.incident.com/cap/docs/aps/Advanced_EAS_Concept.pdf (retrieved 10.12.2008)

3. Botterell, A. (2006) The Common Alerting Protocol: An open Standard for Alerting, Warning and Notification, in proceedings of the 3rd International ISCRAM Conference (B. Van de Walle and M.Turoff, eds.), Newark, NJ.
4. CAP Cookbook (2008) http://www.incident.com/cap (retrieved 10.12.2008)
5. Common Alerting Protocol v1.1 (2005) OASIS Emergency Management TC, http://www.oasisopen.org/committees/download.php/14759/emergency-CAPv1.1.pdf (retrieved 10.12.2008)
6. Emergency Data Exchange Language, Distribution Element, v1.0 (2006) OASIS Emergency Management TC, http://docs.oasis-open.org/emergency/edxl-de/v1.0 (retrieved 10.12.2008)
7. Finnish Funding Agency for Technology and Innovation (Tekes), http://www.tekes.fi/eng (retrieved 15.4.2008)
8. IBM developerWorks (2007) X-Capatom, https://www.ibm.com/developerworks/library/x-capatom (retrieved 10.12.2008)
9. NICT (2007a), in proceedings of the First International Symposium on Universal Communication (ISUC), Kyoto, Japan.
10. NICT (2007b), in proceedings of the First International Workshop on Knowledge Cluster Systems. Keijanna, Japan.
11. Organization for the Advancement of Structured Information Standards (2008) http://www.oasis-open.org (retrieved 10.12.2008)
12. Organization for the Advancement of Structured Information Standards (2005) User Requirements Document OASIS_TA22_REQ_003_DSF
13. RFC 4287 (2005) The Atom Syndication Format, IETF Network Working Group
14. RFC 5023 (2007) The Atom Publishing Protocol, IETF Network Working Group
15. Soini, J., Leppäniemi, J. and Jaakkola, H. (2008) Towards Seamless Collaboration in Distributed Disaster Knowledge Management, in proceedings of the Information Society 2008 Conference, Ljubljana, Slovenia.
16. Tampere University of Technology (TUT), http://www.tut.fi (retrieved 15.4.2008)
17. Verma, P. and Verma, D. (2005) Internet Emergency Alert System, in proceedings of Military Communications Conference (MILCOM 2005), Hawthorne, NY, USA.
18. Westfall, J. (2007) CAP Index Format Proposal, http://www.incident.com/cap/feed-format-proposal.pdf (retrieved 10.12.2008)