The Seven Main Challenges of an Early Warning System Architecture

Jürgen Moßgraber, Fernando Chaves
Fraunhofer IOSB
{juergen.mossgraber, fernando.chaves-salamanca}@iosb.fraunhofer.de

Stuart E. Middleton, Zlatko Zlatev
University of Southampton IT Innovation Centre
{sem, zdz}@it-innovation.soton.ac.uk

Ran Tao
Queen Mary University of London
ran.tao@eecs.qmul.ac.uk

ABSTRACT

In this paper, we describe the work on the system architecture that is being developed in the EU FP7 project TRIDEC on "Collaborative, Complex and Critical Decision-Support in Evolving Crises". One of the two decision support use cases in the project deals with Tsunami Early Warning. A modern warning system that follows a system-of-systems approach has to integrate various components and subsystems such as different information sources, services and simulation systems. Furthermore, it has to take into account the distributed and collaborative nature of warning systems. Working on the architecture of such a system, you need to deal with a lot of current computer science and information technology problems as well as state-of-the-art solutions from the areas of Big Data and Human Sensors. In this paper, we present the seven main challenges we needed to solve and describe the necessary design decisions we made to tackle them.

Keywords

System architecture, System-of-systems, Middleware, Early Warning.

INTRODUCTION

Working on the architecture of a Tsunami Early Warning System (TEWS) we noticed that we deal with a System-of-systems (SoS). There is no finally accepted definition of the term SoS but five helpful characteristics to identify a SoS are given in (Maier, 1998) for distinguishing very large and complex but monolithic systems from true SoS:

1. Operational Independence of the Elements: If the SoS is disassembled into its component systems the component systems must be able to usefully operate independently.

2. Managerial Independence of the Elements: The component systems not only can operate independently, they do operate independently.

3. Evolutionary Development: The SoS does not appear fully formed. Its development and existence is evolutionary with functions and purposes added, removed, and modified with experience.

4. Emergent Behaviour: The system performs functions and carries out purposes that do not reside in any component system.

5. Geographic Distribution: The geographic extent of the component systems is large. Large is a nebulous and relative concept as communication capabilities increase, but at a minimum it means that the components can readily exchange only information and not substantial quantities of mass or energy.

All of these five characteristics fully match a TEWS. Several systems are implemented and operated by different governments and institutions, like Watch Centres (National and Regional), Warning Centres (National), Warning Focal Points (National), Task Forces and Authorities (National), Scientific Institutions (National and Regional) and Data Centres (National and Regional).

They are spread over a wide geographical area and are often extended and modified by integrating new sensor

networks, analysis algorithms, etc.

An important part of the architecture of a SoS is the specification of how the systems work together. For SoS dependent on information exchange, interface management focuses on how the systems share information (Baldwin, 2008). For these systems, there is a need to define shared communication mechanisms. Equally important is the definition of the common or shared data syntax and semantics.

Based on these understandings we identified the seven most important architectural challenges:

1. Build a scalable communication layer for a SoS
2. Build a resilient communication layer for a SoS
3. Efficiently publish large volumes of semantically rich sensor data
4. Scalable and high performance storage of large distributed datasets
5. Handling federated multi-domain heterogeneous data
6. Discovery of resources in a geo-distributed SoS
7. Coordination of work between geo-distributed systems

Each challenge is presented in the same way. First, the challenge is presented, then the design decision we made is described and finally alternatives and pros and cons are discussed.

RELATED WORK

Designing a large SoS covers a large area of computer science and therefore plenty of related work exists. The main areas are Service Oriented Architecture (SOA), Event Driven Architecture (EDA), sensor networks on a large scale and semantics. Projects who tried to combine these areas in a geo-related application are for example the European projects ORCHESTRA (Usländer, 2007) and SANY (Usländer, 2009).

Current early warning systems like the German Indonesian Tsunami Early Warning System (GITEWS) already make extensive use of standardisation and functional integration. Also further successful EU FP6 projects like DEWS, TRANSFER, NERIES and other international ventures like GITEWS (Rudloff, Flueh, Hanka, Lauterjung and Schöne, 2006) facilitated state-of-the-art mechanisms derived from the principles of standardisation and service oriented architectures: encapsulation of proprietary resources, loose coupling of components, transparency of service locations and separation of concerns.

CHALLENGE 1: BUILD A SCALABLE COMMUNICATION LAYER FOR A SOS

In TRIDEC, different system components (subsystems), e.g., semantic registry, knowledge base and workflow service, are deployed in either the same "site" or different "sites" based upon their functionalities. The messaging service in such SoS architecture aims to build up a messaging channel between theses different subsystems (e.g., from workflow service to knowledge base) to enable the resilient (see Challenge 2) and scalable communication between them. These subsystems are designed and implemented with different programming languages (e.g., Java, C#) and deployed on different operating systems (e.g., Linux, Windows). Such heterogeneity of the subsystems requires a standard and open communication layer that allows loosely-coupled and asynchronous communication between the heterogeneous sources.

Design Decision

A Message-oriented middleware (MOM) provides a messaging service layer between the transport and application layer of the networking protocol stack (Wang et al., 2010, 2011). It enables distributed applications and distributed systems in heterogeneous environments to communicate by message exchange. The MOM infrastructure does not only support synchronous communication model, but also allows asynchronous information exchange, which is preferable in many cases for temporally and spatially distributed application integration and information dissemination (Pietzuch and Bhola, 2003). In addition, it enables applications or systems to exchange messages with other applications or systems, without having to know details about the others' platforms and networking, thus increasing the interoperability, portability and flexibility.

Many potential messaging protocols and implementations exist to be used by MOMs. Among these products and protocols, we selected Apache Qpid that supports the Advanced Message Queuing Protocol (AMQP). Qpid implements the latest stable AMQP specification (version 0-10), providing transaction management, queuing,

distribution, security, management, clustering, federation and heterogeneous multi-platform support. All of these functionalities fit the requirements of the SoS design. In addition, to build a scalable communication layer in TRIDEC, we enhanced the scalability function of Qpid by introducing a novel distributed broker topology and overlay management. This design ensures that clients and brokers can join or leave the system without interrupting other components.

Discussion

There are other alternative design choices that allow the communication between different applications or systems, e.g., point to point (P2P) systems. However, P2P systems do not provide many-to-many publish/subscribe communication which is a fundamental requirement in a SoS design for TRIDEC. Therefore, P2P communication system was discarded.

MOMs can be divided into proprietary MOMs and open-source MOMs based upon the ownership. Normally, the proprietary ones provide more advanced functionality such as ultra-fast transmission and enhanced security, but they have two major limitations in scientific research. First, they are more difficult to make extensions to enhance the system for a better fit of the design requirements. Second, they introduce extra cost to get the license. Therefore, in TRIDEC, we focus on open-source MOMs.

With focuses on open and standard MOM protocols, the following MOM protocols were initially selected:

- Advanced Message Queuing Protocol (AMQP),
- Extensible Messaging and Presence Protocol (XMPP),
- Streaming Text Orientated Messaging Protocol (Stomp),
- RESTful enterprise-level Messaging System (RestMS), and
- OpenWire.

To select a protocol that is most suitable for us, we adopt the following three rules according to the requirements of the system design:

1. The protocol must be a wire protocol, not just an API standard, since we need the interoperation of one or more applications in a network.
2. The protocol must be a binary protocol, not a text-based protocol, as the former one provides a faster translation and interpretation speed.
3. The protocol must be platform-agnostic and language-agnostic, so that different implementations running on different platforms are able to interoperate seamlessly.

AMQP, an open, royalty-free and unpatented networking protocol for messaging middleware, which in addition supports the publish/subscribe paradigm, makes different implementations interoperable and allows the transmission of any kind of information as the content of a message. It met all of the requirements and was finally selected.

CHALLENGE 2: BUILD A RESILIENT COMMUNICATION LAYER FOR A SOS

During the message exchange process, system failures, including broker failure, link failure, and client failure, may occur due to either running out of underlying system resources or a poor Quality of Service resulting in message loss. E.g., in Tsunami early warning systems the resilience between regional and national warning centres is important to ensure critical messages are not lost or delayed. Therefore, to maintain the message dissemination process in the face of failure, a resilient communication layer for a SoS is required.

Design Decision

Apache Qpid provides guaranteed delivery and reliable communication by adopting High Availability Clustering techniques. However, this design provides no guarantee to the link failure and requires more space to back up brokers. To overcome these limitations, following techniques are introduced to provide resilient messaging service in a SoS:

Topic Mirroring

As a topic based publish/subscribe MOM (PSMOM) is developed, the messages exchanged by the MOM are labelled with topics. Topic mirroring is introduced to replicate messages under specific topics to both primary broker and mirror broker, i.e., if any of the brokers fail, the messages can still be continually disseminated.

Self-healing

Self-healing is the idea that the system can automatically recover to the resilience status after a component fails. For example, if a broker fails, it will be automatically restarted; and a new broker will be allocated to the clients that are served by the failed broker, i.e., the replicated topics are still served by two brokers.

Redundant WAN Messaging with Source Routing

For the message exchange over WAN, a source routing strategy is introduced. That is, the overlay paths (i.e., from one broker to another broker) of the messages are defined at the source domain from which the messages are published. The paths are selected based upon the current link status. In addition, a redundant link is introduced. The messages are routed through different paths to reach the destination in a way that even if a link is broken, the messages can still be disseminated through the redundant path to avoid message loss.

Durable Message Queue

Durable Message Queue, which will exist in the broker unless the related subscriber clients are closed properly, is adopted in the design. It is used to provide resilience against subscriber client failure as the messages are stored in the queue before the failed subscriber client recovers. However, this may introduce trouble when a subscriber client is terminated unexpectedly and no longer recovers since that the queue will be built up and eat more resources. So a queue management component is introduced, in which a maximum waiting time is set. That is, if the waiting time exceeds the maximum value, the durable queue will be deleted.

Discussion

There are some alternative available design choices, e.g., SOA, brokerless messaging, and Cloud Computing, but none of them provide a better resilience support than a MOM infrastructure. The drawbacks are listed below:

- SOA provides a loose coupling communication but there is no inherent resilience.
- Compared to PSMOM system, brokerless messaging focus less on the functionalities of discovery and management while building a messaging network, i.e., it is harder to develop a resilient messaging for a brokerless messaging system because it is quite difficult to setup resilience management for clients and dissemination paths.

Cloud Computing is a highly scalable model for processing and data storage. It can be used to deploy a messaging system (e.g., Qpid) but itself focuses more on immediate scalability, virtualization, resource management, and utility computing (Vaquero et al., 2009).

CHALLENGE 3: EFFICIENTLY PUBLISH LARGE VOLUMES OF SEMANTICALLY RICH SENSOR DATA

In common with many open sensor and environmental SoS we have a need to publish large volumes of heterogeneous data. In TRIDEC we see temporal datasets (e.g. time series of tide gauge sensor data), spatial datasets (e.g. shape files representing spatially clustered data fusion results), spatial-temporal datasets (e.g. time sliced Tsunami simulations of wave propagation), web 2.0 datasets (e.g. twitter) and thematic datasets (e.g. classifications of earthquakes with Tsunami potential). Data throughput varies by data source, with throughputs ranging from hourly measurements (e.g. tide gauges) to measurements every 10th of a second. Data volumes also expand continually over time, with for example web 2.0 crawlers generating gigabytes of data per day. The challenge is to make this data accessible efficiently, but still keep the semantic meaning of the data for subsequent intelligent processing.

Design Decision

In TRIDEC we have separated our data and metadata and published each using a different communication channel. We store data source metadata in a semantic registry which our client applications can look up before they subscribe to the message-oriented middleware. This metadata follows the Sensor Web Enablement (SWE) Observations and Measurements (O&M) information model[1] and includes descriptions of the data encodings and machine readable descriptions of how to decode them (e.g. for textual data the delimiters to parse the textual messages). Data publication uses a minimalist encoding for efficient transmission in message oriented middleware message payloads. In this way we have 'self-described' data sources but have avoided compromise on the data throughput we can achieve.

Discussion

There are of course other options available we have considered and rejected. The simplest is to provide all data in a set of a distributed database and allow remote database queries (e.g. via a SSH tunnel or a HTTP tunnel over a message oriented middleware). In practice we found reluctance from system administers to allow this due to security concerns with executing SQL statements derived from foreign computers on our server clusters.

Having rejected direct database connections we looked at options for the publication of data over a message oriented middleware. A lot of the domain standards (e.g. SWE O&M for the geospatial domain combines data and metadata in a verbose self-descriptive message. Most XML encodings like SWE O&M are very expressive but become inefficient for high-throughput data transfer. JSON encodings are more efficient, but lack metadata beyond simple key/value pairs. (Compressed) binary data encodings (e.g. HDF5[2] or netCDF[3]) are perhaps the most efficient, but you need to handle the metadata yourself.

We decided in the end to avoid compromising on throughput performance and metadata expressivity by splitting the data and metadata. We see in the TRIDEC project message sensor throughputs of up to 700 msg/sec (messages about 6 Kbytes) and tweet throughput of up to 12,000 msg/hour (JSON messages about 2 Kbytes).

CHALLENGE 4: SCALABLE AND HIGH PERFORMANCE STORAGE OF LARGE DISTRIBUTED DATASETS

In TRIDEC we expect to see distributed data sources publishing up to gigabytes of data each day, accumulating over the period of several months to the terabyte scale. This raises the challenge of how to efficiently store these distributed datasets, both in working caches for fast real-time access and archived forms which can be re-instantiated for offline data analysis. In TRIDEC the processing services (e.g. Tsunami signal detection from different tide gauge sensor networks) need to access several datasets at once (e.g. cached sensor data, pre-calculated tidal harmonic datasets etc.) to produce intelligent data fusion results, which are subsequently made available to decision makers in real-time.

Design Decision

For working caches of data we use a hybrid relational database and triple store database solution; MySQL[4] for data (text, numeric and blob) and OWLIM[5] for metadata (RDF[6]). The RDF metadata stored describes both the SQL table structure and the phenomena being stored. This allows clients to connect to new, unseen datasets and in a machine understandable way determine which SQL queries can be executed to retrieve specific phenomena. Domain ontology URIs are used to link SQL column descriptions to concepts representing measured phenomena. If required binary files can also be stored on file storage solutions (e.g. FTP, network accessible disks) and URLs added into the database to reference them.

For archived datasets we adopt a design strategy of creating HDF5 binary archive objects for datasets. HDF5 was chosen as it allows metadata to be stored alongside the data. The RDF metadata can also be archived and

[1] http://www.opengeospatial.org/projects/groups/sensorweb

[2] Hierarchical Data Format (HDF, HDF5), http://www.hdfgroup.org/

[3] Network Common Data Form (NetCDF), http://www.unidata.ucar.edu/software/netcdf/

[4] MySQL, http://www.mysql.com/

[5] OWLIM, http://www.ontotext.com/owlim

[6] Resource Description Framework (RDF), http://www.w3.org/RDF/

this allows an offline process to download a HDF5 file, unpack it and re-create the databases ready for work.

Note: MySQL is adequate for the real-time throughput we experience in TRIDEC. Commercial highly-optimized relational database solutions can replace MySQL (e.g. Oracle, DB2, etc.) if additional performance is required.

Discussion

Within TRIDEC we have performed a number of data storage benchmark tests, using representative geospatial type queries, and reviewed comparative database studies (Mironov, Seethappan, Blondé, Antezana1, Lindi and Kuiper, 2010). We found triple stores offer good metadata query support (i.e. SPARQL standard) but query times do not scale well, with 100,000 measurements exceeding our 1 second query response target. Relational databases offer excellent performance (i.e. SQL standard), with 10,000,000 measurements exceeding our 1 second query time target, but are limited in metadata support and inflexible to dynamic table structure changes (e.g. as the result data structure evolves). Combining the two solutions was therefore an attractive option.

We reviewed some NoSQL class solutions (e.g. column stores (Tudorica and Bucur, 2011)) which were suitable for certain data structures, along with Map Reduce solutions (e.g. Apache Hadoop[7]) which were most suitable for distributed processing where the query is moved to a data centre. These types of technology could replace MySQL in our hybrid solution effectively, but in TRIDEC these were 'overkill' for our real-time processing needs since we mostly provide data fusion results for short processing time horizons.

CHALLENGE 5: HANDLING FEDERATED MULTI-DOMAIN HETEROGENEOUS DATA

From the authors experience across many open geospatial information systems, for which TRIDEC is one example, there is a clear need to support federated queries on distributed datasets rather than naively move all the data to one site (e.g. a data centre). Reasons to distribute data include performance considerations (e.g. move individual datasets close to processing servers), ownership/control issues (e.g. datasets owners want local control of data they own) and political issues (e.g. national Tsunami warning centres each keeping sensor data controlled in-house). Coupled with this is the problem of handling heterogeneous domain datasets, each with potentially different, or even conflicting, domain vocabularies for the same core phenomena.

Design Decision

In TRIDEC we have adopted a broker-based mediation design pattern for access and transformation of data between domains, focussing on scalability over raw performance. The mediation pattern is a scalable approach, allowing domain brokers to be added incrementally but suffers from a performance penalty by adding an extra ‘hop’ to the information workflow.

We store domain ontologies, each holding domain specific vocabulary for measurement phenomena, in locally hosted semantic registries and allow federated data queries between datasets. Mediation brokers are used to map results from source domains to a single known target domain. This is a very scalable approach, with domain mappings incrementally added over time as need arises, but does require extra query result aggregation to be performed by clients.

Discussion

We considered in TRIDEC a number of alternative architectural solutions for this challenge. We could map vocabulary from data sources and/or applications to a global ontology but, whilst an efficient solution to handle the vocabulary mapping problem, it was not practical to get agreement between different domains as to what this global ontology would be. We adopt instead a 'use what's there' approach to domain standards. The opposite is also possible, where data sources and/or applications each locally map vocabulary between data models. This approach is problematic when scaling as inconsistencies between local mappings are likely to arise. Finally there are options to use automatic metadata and ontology alignment services (Haslhofer, 2010), which is a scalable solution relying on automation to support new domains, but was considered difficult and error prone in practice.

[7] Apache Hadoop, http://hadoop.apache.org/

CHALLENGE 6: DISCOVERY OF RESOURCES IN A FEDERATED GEO-DISTRIBUTED SOS

The management of resources is generally the responsibility of local, in parts very heterogeneous stakeholders. Therefore, in evolving crises it is essential to be able to discover and permanently keep track of all potential and factually available geo-distributed resources at all times based on metadata descriptions of their purpose and capabilities. This is essential for planning and disposition purposes. However, centralized command and control systems are confronted with a number of problems on the organisational and even political level which tend to reduce the actuality and reliability of their data and which represent critical bottlenecks in a crisis. In a federated environment, modularity, interoperability and resilience of the supporting systems therefore play a crucial role.

Design Decision

In TRIDEC we follow an approach based on multiple semantic registries hosted by the stake-holders, who manage and are responsible for the respective resources. Each semantic registry provides a number of human and programmatic interfaces (frontends) and a local ontology store (e.g. OWLIM) based on standards. The metadata description of a resource is an instance stored in a local ontology store, i.e. a set of RDF triples which comply with a shared core-ontology. The core-ontology itself consists basically of classes/sub-classes and properties defined as extensions of the Semantic Sensor Network Ontology (SSNO) and other aligned ontologies (e.g. DUL, SWEET). A metadata description of a resource can be retrieved as an OGC/SWE metadata document, e.g. SensorML, accessed via remote SPARQL or via a RESTful interface, e.g. in JSON. Similarly, metadata descriptions of new or changed resources are published over specific MOM topics by the responsible stakeholders. They can thus be adopted, as local copies of the resource metadata description in their respective ontology stores, by other semantic registries which have subscribed to the dedicated MOM topic. Metadata descriptions use ontology URIs to refer to concepts in namespaces provided by the semantic registries, e.g. for consistent annotation of OGC/SWE documents. By strictly following agreed-on ontology evolution patterns (e.g. by using subClassOf, equivalentClass), each stakeholder can specify own local extensions of the core-ontology and, nevertheless, maintain semantic interoperability with other semantic registries in the network. In this manner, data and services can be described in a semantically rich way, which allows for searching/browsing of instances by both humans and machines and which offers high flexibility for adaptation to different domains.

Discussion

When designing the semantic registries for TRIDEC we considered different alternative approaches:

Classical search engines and catalogues, such as the Catalogue Service Web (CSW) of OGC, follow a “pull” paradigm (“harvest” and “index” – as against the “push” paradigm described above) and generally rely on a central system for maintenance and access to (unified) metadata descriptions of resources coming from very heterogeneous data sources. Although many out-of-the-box commercial and open-source solutions exist, they generally have very limited semantic-indexing and search capabilities. Even slight changes of formats in individual data sources may require arduous adaptation of the harvesting or indexing components.

Monolithic and proprietary systems (as against federated systems) are generally optimized and show good performance for specific tasks. Usually they encompass additional tools for development and operation of applications, but they rely strongly on implicit dependencies between the different components and tools.

We also considered centralization of services in one semantic registry (as against many federated registries) in order to avoid possible inconsistencies and the need for “synchronisation” between registries. However, such an approach greatly reduces the desired flexibility which is necessary for local adaptation of individual registries and administration of local resources because access-control mechanisms tend to become complex and – at least to some extent – have to be centralized also. This approach might also lead to bottlenecks if (e.g.) the central registry is not available or is not capable of handling the loads which might arise due to simultaneous but otherwise independent crises.

CHALLENGE 7: COORDINATION OF WORK BETWEEN GEO-DISTRIBUTED SYSTEMS

Coordination of work between geo-distributed systems for crises management implies providing support to a potentially high number of stakeholders and actors who play different and changing roles within a crisis and who require individual access to a multitude of resources – including other actors – in order to be able to make timely and well-founded decisions. Setting-up a shared vocabulary and equivalent language-specific vocabularies for gaining and describing a common situation picture poses a major challenge in itself which –

amongst others – has been addressed by FP6 EU project OASIS (Cullen, 2009). Furthermore, organisational and legal rules which govern decision-making at different localities affected by a crisis may vary from country to country, between regions and even between involved organisations. The formalisation and codification in machine-readable form of this kind of "business rules" is carried out by local domain experts generally lacking knowledge of and expertise with appropriate IT tools (e.g. workflow or rule languages such as BPEL[8] or SWRL[9]).

Design Decision

The TRIDEC approach focuses on the exchange of (standardized) messages via specific MOM topics between TRIDEC architecture conformant but otherwise largely independent system instances or nodes within the SoS. The messages trigger standard or locally adapted workflows and rule sets on each of the nodes of the network which have subscribed to the specific MOM topics. This kind of message-based event-processing allows for complex and rich choreographies where each node can react specifically in accordance with local rules as well as on global constraints. Encoding of rule sets and adaptation of workflows is carried out by domain experts themselves via easy-to-use decision table and BPMN[10] editors (Riedel and Chaves, 2011). Decision tables are simply an intuitive and easy-to-use notation technique for encoding complex and self-documenting rule-sets even by IT non-experts (Huysmans, 2010) and for which a great variety of tools and interfaces exist. Both decision tables and workflows make use of vocabularies and metadata descriptions modelled by means of ontologies and stored in the semantic registries as described for challenge 6.

Discussion

Specification of complex business rules based on ontology reasoning is an alternative approach for which powerful and expressive reasoning and rule systems exist. However, these systems require a high level of expertise in the field of semantic technologies. Performance and the quality of results strongly depend on the size and the complexity of the underlying ontologies. Many reasoners still have to load the ontologies involved in a reasoning request into main memory for performance reasons. In this context, persistent and at the same time efficient storage of large ontologies, e.g. with many individuals, as considered in the TRIDEC scenarios, may constitute a serious bottleneck. Other, non-standard reasoning paradigms are often very domain specific and out-of-the-box tools are not easily available or even lacking.

Similarly, workflow engines facilitating languages such as BPEL tend to become bottlenecks because, when orchestrating basic distributed services, they must take account of many different combinations of use cases and roles. This makes orchestrating workflows complex, inflexible and difficult to design and edit.

An interesting approach is the enhancement of decision tables to (so-called) semantic decision tables, which make use of ontology elements in their rule sets. However, mapping of "fact models", i.e. sets of variables and rules, to ontology elements is not standardized and still subject of research, e.g. by STAR.lab of Vrije Universiteit Brussel (Tang and Tram, 2012), as well as of on-going work within TRIDEC.

CONCLUSION AND FUTURE WORK

In this paper we presented the main problems when dealing with the architecture of a SoS on the example of a Tsunami Early Warning System. Building a SoS requires a scalable and resilient communication layer as its basis. Large amounts of data need to be published and processed which requires a scalable storage concept that respects the geo-distributed nature of the data.

A well designed system may still fail in the real world for example if it ignores the effects of the human factor. Therefore, the TRIDEC software framework has already been deployed as a tsunami early warning system at the premises of the Instituto Português do Mar e da Atmosfera (IPMA) in Lissabon and the Kandilli Observatory and Earthquake Research Institute (KOERI) in Istanbul for testing purposes. Both institutes are partners in the TRIDEC project that is coordinated by the German Research Centre for Geosciences (GFZ). Additionally, on November 27–28, 2012, two scenarios in the European-wide Tsunami exercise NEAMWave2012 were successfully validated. The software demonstrated the seamless integration of multiple sensor systems,

[8] http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[9] A Semantic Web Rule Language Combining OWL and RuleML, http://www.w3.org/Submission/SWRL/

[10] Business Process Model and Notation (BPMN), http://www.omg.org/spec/BPMN/2.0/

simulation data, and dissemination hardware. New functionalities like direct Centre-to-Centre communication via software systems between Turkey and Portugal and the ingestion of eyewitness reports sent from mobile devices via apps were available for the first time.

The findings of our design challenges cannot only be applied to other Early Warning Systems (as we already did in TRIDEC for a Drilling scenario) but also for other systems who need to make decision based on large scale sensor systems like the Internet of Things (IoT) domain.

In the last year of the TRIDEC project we will focus on improving the resilience and workload allocation of the MOM, improve the resilience of the Semantic Registry by providing a replication mechanism and research the federated access to Big Data stores.

ACKNOWLEDGMENTS

This research work was funded by the European Commission FP6 IST Programme under contract TRIDEC IP FP7-258723.

REFERENCES

1. Maier, M. W. (1998) Architecting Principles for Systems-of-Systems. Systems Engineering 1(4), 267-284.
2. Baldwin, K. J. (2008) Systems Engineering Guide for Systems of Systems, Version 1.0. Washington, DC: ODUSD(A&T)SSE.
3. Usländer, T. (ed.) (2007) Reference Model for the ORCHESTRA Architecture (RM-OA) V2 (Rev 2.1), Open Geospatial Consortium Inc.
4. Usländer, T. (2009) Specification of the sensor service architecture Version 3.0 (Rev. 3.1), OGC Discussion Paper 09-132r1, Deliverable D2.3.4 of the European Integrated Project SANY, FP6-IST-033564.
5. Rudloff, A., Flueh, E. R., Hanka, W., Lauterjung, J., Schöne, T. (2006) GITEWS Project Team, The German-Indonesian Tsunami Early Warning System, 3rd General Assembly European Geosciences Union (Vienna, Austria 2006).
6. Pietzuch, P. R., Bhola, S. (2003) "Congestion Control in a Reliable Scalable Message-Oriented Middleware", Proceeding of the ACM/IFIP/USENIX 2003 International Conference on Middleware, pp. 202-221.
7. Wang, J., Bigham, J., Chew, J., Novkovic, M., Dattani, I. (2010) Adding Resilience to Message Oriented Middleware, Serene 2010 ACM, 13-16 April, 2010 London, UK.
8. Wang, J., Murciano, B. V., Bigham, J. (2010) Towards a Resilient Message Oriented Middleware for Mission Critical Applications, The Second International Conference on Adaptive and Self-adaptive Systems and Applications (ADAPTIVE 2010), November 21-26, 2010 - Lisbon, Portugal.
9. Wang, J., Bigham, J., Wu, J. (2011) Enhance Resilience and QoS Awareness in Message Oriented Middleware for Mission Critical Applications, ITNG '11 Proceedings of the 2011 Sevenh International Conference on Information Technology: New Generations, IEEE Computer Society Washington, DC, USA.
10. Mironov, V., Seethappan, N., Blondé W., Antezana1, E., Lindi, B., Kuiper, M. (2010) Benchmarking triple stores with biological data, Proceedings of the 3rd International Workshop on Semantic Web Applications and Tools for the Life Sciences, Berlin, Germany.
11. Tudorica, B. G., Bucur, C. (2011) A comparison between several NoSQL databases with comments and notes, Roedunet International Conference (RoEduNet).
12. Tang, Y., Tram, T. (2012) Using SOIQ(D) to Formalize Semantics within a Semantic Decision Table, RuleML'2012, Springer (2012), cf. http://www.starlab.vub.ac.be/website/SDT_SOIQ.
13. Huysmans, J. et al. (2010) An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. Decision Support Systems, cf. http://dx.doi.org/10.1016/j.dss.2010.12.003.
14. Cullen, A. et al. (2009) The Oasis Approach to Civil/Military Information Sharing for Disaster and Emergency Management, C3I for Crisis, Emergency and Consequence Management Symposium, IST-086/RSY-019, May 2009, Bucharest.
15. Riedel, F., Chaves, F. (2012) Workflows and Decision Tables for Flexible Early Warning Systems,

Proceedings of the 9th International ISCRAM Conference, Vancouver, Canada.

16. Vaquero, L. M. et al. (2009) A Break in the Clouds: Towards a Cloud Definition, ACM SIGCOMM Computer Communication Review, Volume 39, Issue 1, DOI=10.1145/1496091.1496100 http://doi.acm.org/10.1145/1496091.1496100