Model-based Development of Self-organising Earthquake Early Warning Systems

Joachim Fischer, Frank Kühnlenz, Klaus Ahrens

fischer|kuehnlenz|ahrens@informatik.hu-berlin.de

Abstract

A new approach for Earthquake Early Warning Systems (EEWS) is presented that uses wireless, self-organising mesh sensor networks. To develop the prototype of such IT-infrastructures, we follow a model-driven system development paradigm. Structure and behaviour models of network topologies in specific geographic regions are coupled with wave signal analysing algorithms, alarming protocols, convenient visualisations and earthquake data bases to form the basis for various simulation experiments ahead of system implementation and installation. The general objective of these studies is to test the functionality of an EEWS and to optimize it under the real-time, reliability and cost-depended requirements of potential end-users. For modelling a technology mix of SDL/ASN.1/UML/C++ is used to generate the code for different kind of simulators, and for the target platform (several node types). This approach is used for realizing a prototype-EEWS developed within the EU project SAFER (Seismic eArly warning For EuRope) in cooperation with the GeoForschungszentrum Potsdam. The first operational area of that EEWS is already planned for Istanbul in a region threatened by strong earthquakes. The presented paper focuses on our adopted and developed tool-based modelling and data base techniques used in that project, that are general and flexible enough for addressing similar prototyping use cases of self-organising sensor-based IT-infrastructures.

1 Introduction

The concept of Self-organising Seismic Early Warning Information Networks (SOSEWIN) is being developed within the EU-project SAFER[1] in cooperation with the GeoForschungsZentrum Potsdam (GFZ). The work benefits from the Graduate School METRIK[2] on disaster management, supported by the DFG (German Research Society). It focuses on the adoption of METRIK-technologies concerning self-organising, ad-hoc communication infrastructures and model-based software development for prototyping Earthquake Early Warning Systems (EEWS).

The SAFER project aims to fully exploit the possibilities offered by the real-time analysis of the signals coming from seismic networks for a wide range of actions, performed over time intervals of a few seconds to some tens of minutes. These actions include the shutting down of critical systems of lifelines and industrial processes, closing highways, railways, etc., the activation of control systems for the protection of crucial structures, as well as supporting the rapid response decisions that must be made by emergency management (continuously updated damage scenarios, aftershocks hazard etc.) [1] [2].

Present EEWS have a number of problems related to insufficient node density due to the high costs per node necessary for the purchasing, installation and maintenance of the usual more sophisticated seismological stations. However, such problems can be solved by using a low-cost, self-organising, ad-hoc mesh sensor network that avoids more costly planned infrastructure. Such self-organising communication networks were already successfully used within other application areas. One example is the Berlin RoofNet[3], which demonstrates the feasibility to build an autonomous wireless communication network in the city of Berlin at a moderate budget.

This paper demonstrates how the concept of such self-organising mesh networks can be extended and adopted for the development of low-cost EEWS prototypes. As in Berlin RoofNet inexpensive Commercial-Off-The-Shelf (COTS) hardware is used with Linux as operating system and existing communication technologies, such as IEEE 802.11g WLAN, which operates in the unlicensed 2.4 GHz ISM band. Communication is close to real-time (delay ~ 0.5 - 1.0s), robust (mesh-structure with redundant paths) and based on the Internet Protocol, allowing for easy integration with existing applications and with the external public Internet (where available) [3]. Low-cost ground acceleration seismometer and GPS receiver are the basic components to recognize wave signals in dependence of time and locality.

---

[1] Seismic eArly warning For EuRope, http://www.saferproject.net

[2] Model-based Development of Technologies for Self-organising Systems, http://www.gk-metrik.de

[3] http://www.berlinroofnet.de

A SOSEWIN network consists of nodes of different types with slightly different tasks. The elementary tasks are

- Routing Task: forwarding of received messages by wireless communication,
- Sensing Task: monitoring ground shaking using seismometer and GPS functionality,
- Alerting Task: issuing signals for alarms and reset alarms at different levels,
- Management Task: supporting installation, maintenance and control SOSEWIN for different manager types (seismological or network experts),
- Visualizing Task: supporting visualisation of SOSEWIN state information for different end users (public, decision maker in disaster's management).

In principle, each of the SOSEWIN nodes must undertake all of these tasks. However, there are different restrictions, depending upon the node's equipment and the task requirements. Fig. 1 shows a simplified SOSEWIN topology with typical nodes.

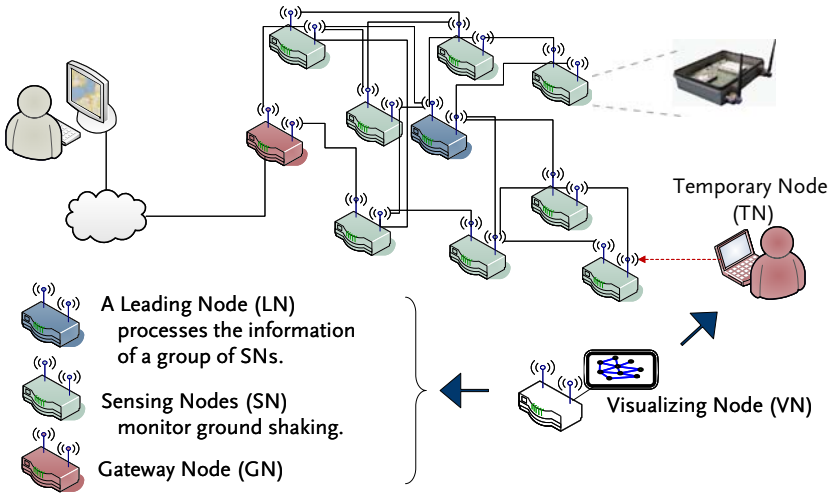


Fig. 1 A SOSEWIN example topology with typical nodes.

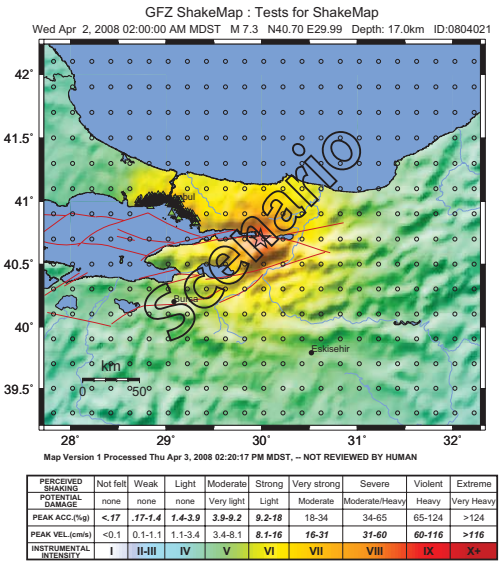
Developing the complex IT-infrastructure, we follow a model-driven system development paradigm. Structure and behaviour models of network topologies in specific geographic regions are coupled with wave signal analysing algorithms, alarming protocols, convenient visualisations and earthquake data bases to form the basis for various simulation experiments ahead of system implementation and installation.

The general objective of these studies is to test the functionality of an EEWS and to optimize it under the real-time, reliability and cost-depended requirements of potential end-users. For modelling a technology mix of SDL/ASN.1/ UML/C++ is used to generate the code for different kind of simulators, and for the target on several nodes. This approach is used for realizing a prototype-EEWS developed within the EU project SAFER (Seismic eArly warning For EuRope) in cooperation with the GeoForschungszentrum Potsdam. The first operational area of that EEWS is already planned for Istanbul in a region threatened by strong earthquakes. However, first SOSEWIN model tests were realized by using historical earthquake data, recognized by a centralised seismometer network in Taiwan.

Our paper is structured into several sections. The next *Section 2* gives some background information to the application area and motivates the impact of earthquake wave signal analysing approaches. *Section 3* summarizes the current situation in the development of EEWS. Especially the advantages of self-organized systems are discussed here. *Section 4* describes the general concepts of our SOSEWIN prototype, developed in the ongoing SAFER project. The principles of our Alarming Protocol (AP) are outlined in *Section 5*. One of the simplest protocol entity of AP is presented here by an UML state machine. However, all of the protocol entities were described in detail by us in SDL, where in the first design stage the Real Time Developer Studio (RTDS) in version 3.4 was used, which supports an SDL dialect (SDL-RT [4]) in combination with UML class diagrams and C++ for activity and data type descriptions. This toolkit was extended in the context of the development of our EEWS prototyping infrastructure. *Section 6* discusses this infrastructure, especially the SDL compiler and simulation components. The current status of the SOSEWIN development is given by *Section 7*. The last *Section 8* summaries the results and activities in future work.

2 Earthquake Waves, Early Warning and Rapid Response

Earthquakes produce different types of *seismic waves*. These waves travel through the earth and provide an effective way to create an image of both sources and structures deep within the Earth. In addition their analysis is the foundation for different activities in a disaster's management process, so for earthquake classification, early warning and first response.



There are four types of seismic waves: P-waves and S-waves (called body waves), Rayleigh waves and Love waves (called surface waves). Body waves travel through the interior of the Earth. *P-waves* (primary waves) are longitudinal or compressional waves, which brings the ground into alternately compressed and dilated movement in the direction of propagation. In solids, these waves generally travel almost twice as fast as *S-waves* (secondary waves) and can travel through any type of material. In air, these pressure waves take the form of sound waves, hence they travel at the speed of sound. Typical speeds are 330 m/s in air, 1450 m/s in water and about 5000 m/s in granite[4]. When generated by an earthquake they are less destructive than the S-waves and surface waves that follow them. *Surface waves* remain below the Earth's surface. They can be much larger in amplitude than body waves, and can form the largest signals seen in earthquake seismograms. Seismograms are more strongly excited by surface waves particularly when the seismic source (*hypo centre*) is in close to the surface of the Earth.

Fig. 2 Shake Map Example of a synthesised Earthquake.[5]

It is not possible to predict an earthquake event. The only chance for preparation on the coming disaster is to use the most of the time delay between the arrival times of the P- and S-wave. In dependence of the distance between the *epicentre* of the earthquake (transferred hypocentre on the Earth's surface) and the critical area locations only few seconds to some tens of minutes remain for an *early warning*. But there is another important task in analysing earthquake waves which supports to save human life. This is a fast generation of so-called *shake maps*, which show the wave peaks in the area (influenced by the earthquake event) in form of isobar lines or different colours. The combination of such shake maps with existing building and inhabitation structures can offer start estimations of the disasters when these information would be available very fast after an event. A special kind of shake map is an alert map. It is based on incomplete earthquake event descriptions (only on entrance signal data series) during the earthquake itself. The generation of such maps is an actual engineering challenge.

3 Earthquake Early Warning Systems

EEWS are based on the detection of P-waves that do not cause damage but precede the slower and destructive S-waves and surface waves. Dependent upon the distance between the hypocentre and the target area, a maximum early warning time before the S-wave arrives can be computed, based on wave travel characteristics and ground parameters. Therefore, the primary goal of an EEWS is simple: maximizing the early warning time under a minimal number of false alarms (which includes false positives and false negatives). An important secondary goal is to generate alert maps.

3.1 Present: Centralised Approach

Present EEWS always use a centralised approach (for example in the Marmara region, Turkey [5], Southern Apennines, Italy [6] and Taiwan [7]). Each station delivers its measured data or the alarm message for the case of P-wave detection over a (more or less) direct connection to a central data centre (which usually has a secondary data centre for backup). Within the data centre, it can then be decided whether an early warning message should be issued to the end users (e.g. nuclear power plants) who can then decide what actions will be instigated.

[4] Dependent upon the geology of the specific region and the hypocenter depth, P-waves travel at 5-8 km/s, and S-waves at 3-7 km/s.

[5] Created with the ShakeMap generator software provided by the USGS (http://earthquake.usgs.gov/shakemap)

In the case of the already existing Istanbul Earthquake Rapid Response[6] and Early Warning System (IERREWS), ten strong-motion stations were placed as close as possible to the Great Marmara Fault zone, forming the online-sensor-part of the early warning system [5]. These stations are connected to the data centre of the Kandilli Observatory and Earthquake Research Institute via a digital spread spectrum radio link and continuously deliver ground-motion data for archiving and early warning purposes. Depending upon the location of the earthquake's epicentre and the recipient facility, the early warning time can be as high as about 8s [5].

3.2 Dilemma of current EEWS

Current early warning systems, like the above described IERREWS, often consist of only a few, but expensive (several thousands euros) stations. This fact results in a number of problems:

Malfunction: If one station breaks down, then the area it would normally observe can now only be monitored from afar, resulting in time delays that could seriously compromise the network's early warning capacity.

Instrumental Density: This problem is related to the generation of precise information about an earthquake's intensity for city square cells, the size necessary being generally of the order of 500 m. Civil protection experts need such detailed information for reliable loss estimation maps (destroyed buildings, injured people and fatalities) that are the basis for effective planning by rescue teams. By comparison, EEWS usually have a station spacing of several kilometres.

Cost: However, increasing the density of seismic stations is limited by their expense.

Communication: The reliable transmission of all station information to central data stations or civil protection headquarters is very important, especially following an earthquake, where usually centralised communication infrastructures may have collapsed.

3.3 Vision: Decentralised Approach based on Low-cost Wireless Ad-hoc Mesh Sensor Networks

The basic idea presented in this work aims to avoid the problems identified above by deploying a much higher number of much cheaper stations (costing only a few hundreds of euros per station, which is of the order of 10% compared to a classical station).

Another cost factors are the communication modules necessary for the link to the central data centres (in some cases within IERREWS, involving several hundred kilometres). Wireless, ad-hoc mesh sensor networks will allow much cheaper radio modules because a single station needs only to reach the nearest neighbour station, which would be only a few hundred meters away.

In addition, the reliability of such a mesh sensor net is a crucial point, since while single sensors may be destroyed, the whole system nonetheless can still detect the earthquake. This can be achieved because the sensor nodes act cooperatively in a self-organising way. However, a number of challenging problems must be solved first (e.g. development of strategies for self-organization regarding the special requirements of EEWS; routing in huge multi-hop networks; deployment of software components).

The main advantages of such an approach, besides providing a more robust and cheaper architecture than centralised systems, may be summarized as follows:

- The simple deployment and installation of a temporary sensor net. This would be of particular value to, for example, groups such as the German Earthquake Taskforce7, who deploy temporary arrays for the detection of aftershocks. Time consuming planning and (costly) installation of a traditional infrastructure-based system can thus be avoided.
- As mentioned above, in the event of an individual sensor node being destroyed, the self-organising nature of the network will allow alternate communication routes to be established, while the information regarding the loss of the sensor or sensors may be utilized in damage assessments.
- At a latter stage it is planned to provide the capability of using the network as an information system. In the event of a damaging earthquake, individuals will be able to send short messages such as "I am alive." or "I need help!" through a reliable mesh network that is still functional when other systems such as GSM may have collapsed.

[6] The Rapid Response part of the IERREWS comprises about 100 stations that report only peak ground accelerations every 20s in the event of an earthquake. This system is used mainly for rapid damage assessment after an earthquake.

[7] http://www.gfz-potsdam.de/pb2/pb21/Task_Force/index_e.html

4 SOSEWIN Overview

In contrast to existing EEWS, which are planned and centralised, we propose the use of a self-organising ad-hoc wireless mesh network to overcome the problems of planning such a large network and administrating potentially thousands of Sensing Nodes (SNs). The advantages of such a network include robustness, independence of infrastructure, spontaneous extensibility as required, and a self-healing character in the event of failing SNs. However, these networks still pose a great research challenge, particularly regarding a routing-strategy to accomplish scalability requirements and time constraints.

To realise a hierarchical alarming system, the nodes of SOSEWIN are organised into clusters using criteria (the so-called wireless metric parameter) that determines the optimum communications efficiency. Each cluster is headed by a SN that is designated, again based on communications efficiency, as a Leading Node (LN), with whom the other SNs within its cluster communicate general "housekeeping/status" information and initial alarms. The LN in turn communicated with other LNs, including the issuing of system alarms, based on each LN knowing the status of the nodes that make up their clusters.

4.1 Hardware

The Sensing Node prototype consists mainly of two units:

- The *WRAP board*
  serves as a wireless mesh network node and computing unit for signal analysing and alarming. It is an embedded PC produced by PC Engines that can be purchased off-the-shelf. It is practically a 486er embedded PC with a 266 MHz CPU and offers one slot for a CompactFlash card, which acts as the hard disk, and two Mini PCI slots for two WLAN Mini PCI cards. In addition, it has a power supply plug, a serial port and 100 MBit/s Ethernet. Supplying the WRAP Board over the Ethernet interface by using PoE (Power over Ethernet) is also possible.
- The *sensor board*
  which samples the data by analogue-digital-converters and provides them together, with GPS readings, to the WRAP Board. It is powered and connected to the WRAP Board via USB.

4.2 Node Types

For SOSEWIN, the following node types have been defined:

- *Sensing Nodes* (**SN**) monitor ground shaking. Most nodes in the network are of this type.
- *Leading Nodes* (**LN**) are basically Sensing Nodes as they consist of the same hardware. The “leading” property is a role that any SN can fulfil. A LN processes the information of a group of SN in its neighbourhood (usually not more than five SN).
- *Gateway Nodes* (**GN**) represent information sinks in the SOSEWIN that have connections to the end users (via the internet/satellite/cable) outside of the network, and are used for sending early warning messages. It includes the functionality of a SN.
- *External Nodes* (**EN)** are outside of the SOSEWIN and are connected via Gateway Nodes. They are to be informed first in the event of an alarm (e.g. GFZ, HU, Kandilli Observatory KOERI, police stations …).
- *Temporary Nodes* (**TN**) are present in the network only for a short time to access data. An example of a temporary node is the laptop of an earthquake task force member, who wants to access ground shaking maps or waveform data.
- *Routing Nodes* (**RN**) ensure that communications between far-away nodes, which could not communicate otherwise. A Routing Node only delivers messages that it receives and undertakes no analysis. It is useful in being a low-cost way of extending the monitoring to a larger area.
- *Visualizing Node* (**VN**) A Laptop acting as a TN is a typical VN, which is able to come with a GUI to visualise subsequent SOSEWIN states on different abstraction levels by request. It’s also easy to imagine that some of the SNs also have restricted visualization capabilities.

4.3 Software Architecture

The software architecture of SOSEWIN identifies all components that have to be installed according to the acquired use cases. The architecture is characterized by two main levels,

- a lower communication layer combined with the operating system and
- an upper application layer.

Fig. 3 shows the application layer on top of the communication layer. The lower communication layer is responsible for all specific features of SOSEWIN that need communication and/or operating system resources. It offers peer-to-peer communications between any nodes of the SOSEWIN. For that it is divided into four levels of functionality abstraction; a peer-to-peer message transportation layer (realized in a standardised way as TCP/UDP), a routing protocol layer (realized by the Optimized Link State Routing (OLSR) protocol), a Medium Access Layer (MAC) and a Wireless Local Area Network protocol layer (WLAN).

The application layer using operating system functionality and the peer-to-peer message transportation service of the OLSR-Layer to regulate the activities between SNs, LNs, and external nodes (ENs) is based on rules specified for event detection and for distributing alarm messages among LNs and issuing them to end users identified by the EN.

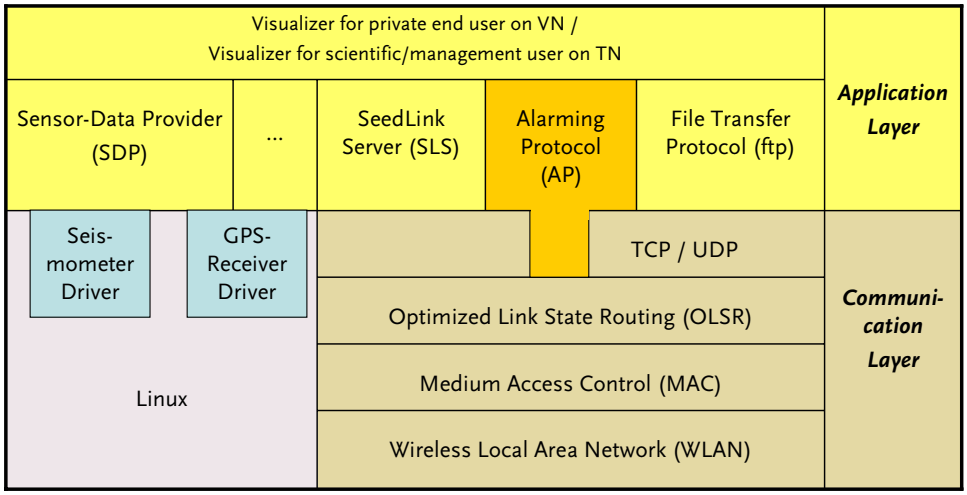


Fig. 3 The SOSEWIN software architecture.

5 Alarming Protocol

5.1 Overview of the system hierarchy

SOSEWIN supports a hierarchical alarming system. That's why the network is composed of node clusters, each cluster headed by a LN, where the cluster members are ordinary SNs or GNs. The definition of the clusters and the designation of which nodes are the LNs (which are themselves simply normal SNs within the network) are given by the initial installation. However, this can be changed dynamically if the network topology is changed. In principle, every sensor node can play two roles, a sensing and a leading role. There is one main rule for the clustering procedure:

> *Each cluster (as a set of SNs) has a cluster head, the LN, where communications between the cluster head node to any other node of the cluster needs no more than two hops*[8].

The Alarming Protocol (AP) uses peer-to-peer communication services realized by the underlying communication layer (TCP/UDP, OLSR [8], WLAN). There is one important requirement in the restriction of TCP message length. To avoid expensive de-fraction and fraction policies, the length should have a maximum of 1024 bytes.

As Fig. 4 shows, the AP is realized by different asynchronous communicating protocol entities:

[8] A distance in terms of topology and of a length that may be not specified topographically, i.e. one hop is the step from one router to the next, on the path of a packet on any communications network (on the Internet often discovered with pings or traceroutes) (http://en.wikipedia.org/w/index.php?title=Hop_%28telecommunications%29&oldid=200567153).

- **SAE** (Signal Analysing Entity)

  This one is responsible for analysing the incoming streams of accelerometric raw data and informing the SE in the event of certain state changes. As presented in Fig. 4 we distinguish two different data sources for SAE. Alternately to a seismometer driver a stored data file (containing synthesized data) can be used here. This is controlled by data provider functions. An additional procedure is to save the raw data in a special data format (SEEDLink [9]) in a ring buffer.

- **SE** (Sensing Entity)

  It reacts on the results from the SAE trigger by informing its associated LE itself if the hosting node is a leading node (otherwise the LE of another node). To improve the SAE-SE-communication they operate over a semaphore-controlled common data base.

- **TE** (Transport Entity)

  It implements the message transport from the communications layer to the corresponding AP entities and vice versa. There are different message types (signal description, alarming, management, …) with individual signatures. To do so it has the knowledge of the member node IP numbers of that cluster where it belongs to. A special task is the coding and decoding of all AP messages to and from other nodes. The coding/decoding procedure is realized by an ASN.1 compilation [10], [11]. For that, a C++ library modelling ASN.1 data types was used, developed by an earlier SDL/ASN.1-C++ compiler project [10].

- **ME** (Managing Entity)

  This entity (not described in detail here) is responsible for interpreting and processing network management messages which come from TNs or the SOSEWIN itself. For instance, one task is to switch and organize the functionality of the owner node as a function of the life cycle stage *pre-operating*, *operating*, and *post-operating*. Further tasks are connected with the dynamic establishment of cluster structures and cluster heads.

- **LE** (Leading Entity)

  The LE monitors all associated SEs (that is the SE of the same node and the SEs of the Sensing Nodes within its cluster). An LE is able to cause or invoke group alerts and is also able to cause system alerts to be issued after communicating with other LEs; each system alert will sent to the TEs of all GNs. When a node is not a LN, then this entity is idle.

These entity ensembles (consisting of SAE, SE, LE, ME, and TE) have to be installed on the SOSEWIN SNs. In addition also the LEs have to be installed on SNs, but remain in an idle state. GNs have at least TEs.

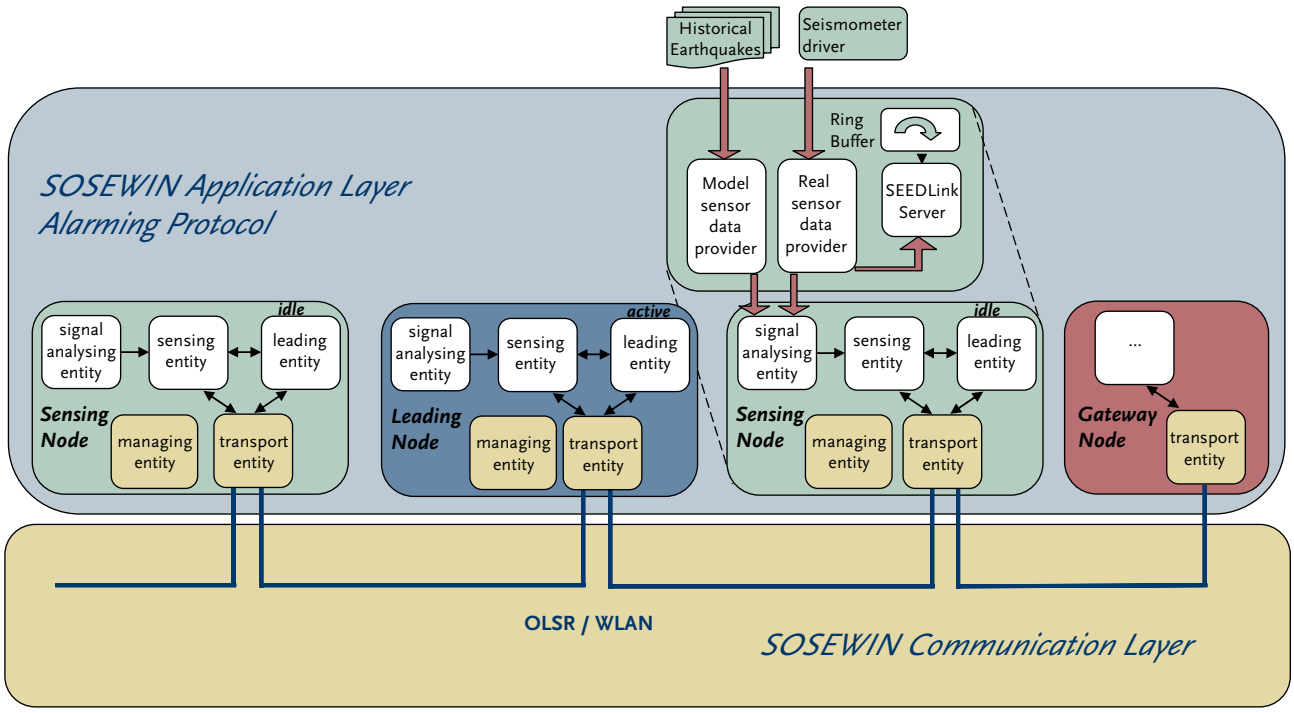


Fig. 4 Nodes and the associated protocol entities of the SOSEWIN Application Layer.

5.2 Informal Protocol Description

Besides event detection, a general goal of the AP is to offer a network service that actualizes the knowledge about the states of all distributed SOSEWIN seismometers by their associated LEs as fast as possible after an individual change. The AP functionality is defined by two sub layers, an internal cluster protocol and a protocol for between clusters. The internal cluster protocol defines the communications between a SAE and SE, and the communication between all SEs of a cluster and their representing LN. The inter-cluster protocol defines the communications between all LEs. If a critical number of P-wave triggers have reached the LE of a cluster's LN, this node informs its neighbouring LNs. In the case that a LE of a LN has received enough cluster alarms, a so-called system alarm will be sent as fast as possible to the GNs of SOSEWIN that are responsible for forwarding those alarms to defined ENs by peer-to-peer communication. According to this hierarchical principle, three alarm levels are recognized by the SOSEWIN:

- Pre-alarm (recognized by the LE of a LN, the requirement being a registration of a P-wave trigger by at least one SN of its cluster);
- Group alarm (recognized by the LE of a LN, the requirement being a certain number of node alarms of this cluster have been registered);
- System alarm (recognized by the LE of a LN, requires a certain number of group alarms registered by this LE).

Because of the independent reaction of the distributed nodes and their corresponding protocol entities, these alarm levels are reached by the individual nodes in a time-displaced manner. In addition to the three alarm states of the nodes represented by their protocol entities, two other states can be distinguished:

- Idle (recognized by the SE of each node, in that no event is occurring and preliminary analysis of the data input is going on);
- Final reporting (recognized by the LE of a LN, and the SE of all nodes, that the event is considered to be finished and the final data/result files (e.g. for ShakeMap) are being produced.

The AP is characterized by the principle that all SNs inform their LNs with as short time delays as possible about their current state without any explicit demand. Doing so, the LNs will be informed about the whole life cycle of an earthquake event according to their SNs. An explicit demand is necessary if ENs (via GNs) or TNs want to collect detailed information on the last event observed by the SOSEWIN. Once the first SN of a cluster has been triggered, the LN assigns an ID to the event, which will be based on the GPS time of the trigger at the first SN that detected it. The ID of the event is therefore the minimum event time, and maybe also with a code identifying the SN. Hence, both the real and false events will be recognized by the network by that code.

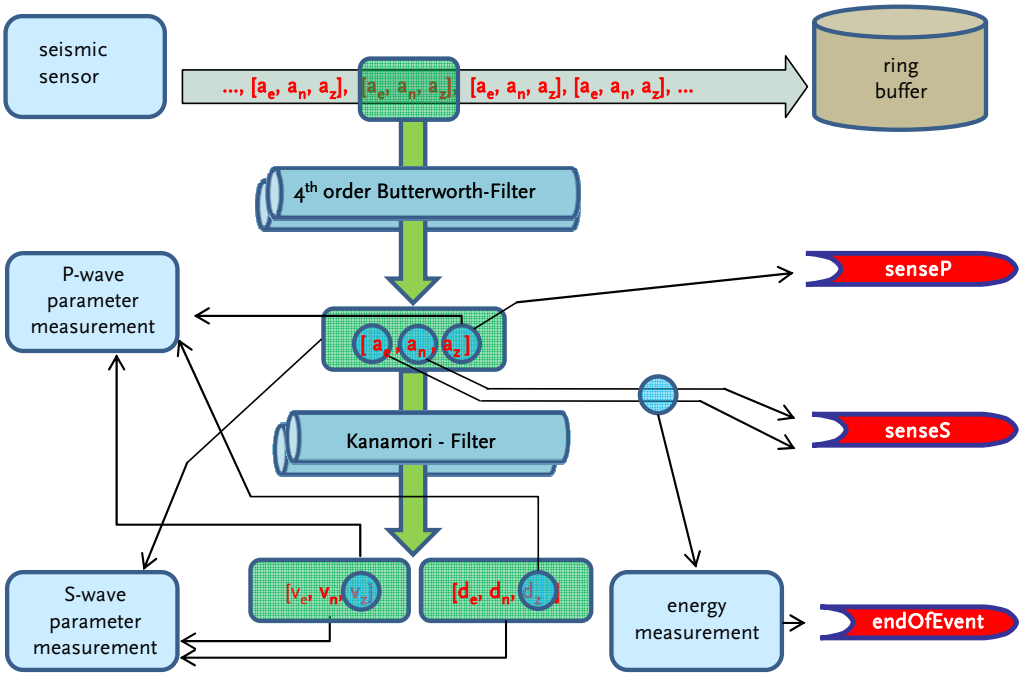


Fig. 5 Scheme of Signal Analysing by SAE.

5.3 Signal Analysing Entity

The SAE is a good example for the impact of a behaviour description, which requires an integration of state machine concepts with C/C++ as action description language, how it is offered by SDL-RT.

The scheme of signal analysing by an SAE is described in Fig. 5. This entity continuously records its local acceleration [a] in three dimensions (e - latitudinal, n - longitudinal, z - vertical) with a given *sampling rate* (e.g. 100Hz). These raw acceleration data are processed first by an IIR pass band filter of Butterworth type. The filtered vertical acceleration is used to detect a P-wave when its signal-noise ratio exceeds a *P-threshold value*. The filtered accelerations in all directions are moreover used to compute the local velocities [v] and displacements [d]. This is done by Kanamori-filters which numerical integrate and filter simultaneously. After P-event detection all vertical magnitudes are monitored to derive crucial parameters of the earthquake event. The function *senseP* allows asking the current state in continuously processing the sensor signals, whether there is an P-wave detection or not. Starting from the P-event a combination of the filtered horizontal accelerations is used both to watch for the S-wave (another *S-threshold value* for signal-noise ratio) and for a progressive measurement of the energy released by the event. After an S-event all derived data are monitored to derive further crucial parameters of the event until the energy gain falls below a *quantile*[9]. The function *senseS* allows asking the current state in continuously processing the sensor signals, whether there is a S-wave detection or not.

The signal analysing functions, derived from that scheme, are called by the SAE state machine within the *processRecord* method (see Fig. 6).

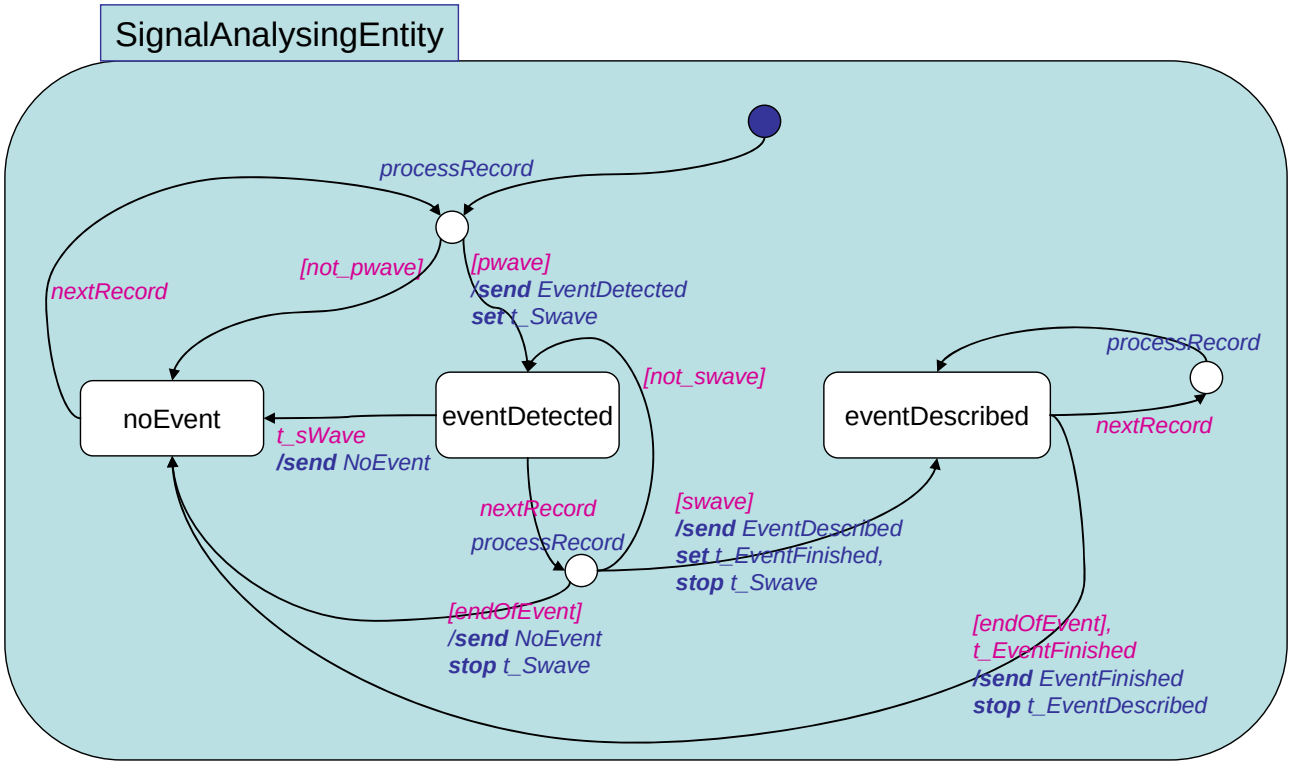


Fig. 6 Signal Analysing Entity as an UML State Machine.

Fig. 6 describes the SAE behaviour in form of a simple UML state machine with three real states. Most of the time the entity is in the state *noEvent*. In this case, the received accelerometric data records were identified only as noise by the *senseP*. However, if a P-wave arrival event is recognized the entity will go into the state *eventDetected*. In addition an *eventDetected* message will be sent to the associated SE and the timer *t_Swave* will be started.

In the state *eventDetected* the incoming data record will be analysed by a function call *senseS* in relation to the current aggregated state, with the aim of detecting the epected S-waves. The SAE will remain in the same state or it will switch to the *eventDescribed* state at which point two more timers (*t_EventFinished, t_EventDescribed*) will be started, and a *EventDescribed* status message will be sent. In the case of the expiring *t_Swave* message, *NoEvent* will be sent to the SE and the entity will return to the state *noEvent*.

[9] *Sampling rate, thresholds* and *quantiles* are only some of the analysis parameters which are to be configured and tuned by seismological experts.

Also in the state *eventDescribed* each incoming data record will be analysed by the function *processRecord* (which is *senseP*, *senseS* and *endOfEvent*) where the signal status will be actualized by each call. This status will be delivered to the SE using *EventDescribed* messages. The SE can access the computed values via a shared memory segment actualized by the SAE. Finally, if *t_EventFinished* expires, the SAE will return to the state *noEvent.*

5.4 Evaluation of SDL-RT Model Descriptions

All AP entities are described in detail in SDL-RT as process types where data and actions semantics come from C/C++. This allows not only a simulated execution and testing of the protocol entities, it also simplifies the code generation by an available cross compiler for the target hardware/operating system architecture. In addition to that the built-in real time features of SDL-RT also support the design and implementation process. For the simulated execution of our formal described protocol entities we have to distinguish different main analysing goals, where each of them is of certain complexity:

- functional evaluation of a single SAE SDL process by varying historical or synthesised earthquake data to check and improve the used signal analysing numerical methods.
- functional evaluation of a single SN as an ensemble of different communicating state machines.
- functional evaluation of a configuration of SNs and LNs by varying historical or synthesized earth quake data,
- performance evaluation of a configuration of SNs and LNs by varying historical or synthesized earthquake data to estimate the capability of early warning.

A lot of parameters have to be tuned by the above scenarios in dependence on the target local area, the network size and topology, on the influence of environmental noise, on the behaviour of used underlying transport protocols. The successful realisation of this complex task to fulfil a compromise of different requirements is the base of continuation of further development steps, so for code generation, software deployment, network installation, network test, and finally for network operating.

To ease the management of the complex prototyping task of EEWS an infrastructure was developed in parallel to the model development itself, implemented by a student project.

6 Prototyping Infrastructure

6.1 Infrastructure Components

The foundation of the tool integration in our EEWS prototyping infrastructure is a centralised management of models, software artefacts, and simulation results by several repositories that are implemented by data base technologies. Fig. 7 shows an overview on the core components realizing the identified requirements and concepts, which are shortly described in the following (from top to down in Fig. 7).

- The *Experiment Management System*

  supports planning, configuration, automated execution of simulations and storage of simulation results. It provides additionally GIS-based visualization capabilities for simulation results (e.g. Detection Maps) that can also be used for planning software deployment and monitoring of an installed SOSEWIN network.

- The *Model Repository*

  stores used SDL(-RT), UML and C++ models defining the entities of the AP. It also holds models of the environment (e.g. for network clustering, message transport properties or node breakdowns).

- The *Model Configurator*

  knows the target platform and uses platform dependent artefacts to configure the compiler (e.g. cross-compilation). It also specifies certain input parameters (e.g. threshold values or network clustering) and stores the whole configuration into the Experiment Repository.

- The *Compiler* (with libraries)

  is indeed a tool chain of several transcompilers, which accept SDL-RT models and compile C++ code at the end into different executable binaries (simulators, target code).

- The *Simulator*

  represents in fact a collection of several simulators of different functionality (simulation framework).

- The *Earthquake Repository*

  comprises time series of historical recorded or synthetic generated earthquake data stored in a relational database system. Various input formats, such as (Mini-)SEED, SAC and several well-known-text (WKT) formats where mapped to the same database scheme, that is used as an uniform interface for simulations.

Using this infrastructure, different testbeds can be offered, namely for the detection of P-waves, for the functional correctness of different protocol concepts, and for the simulation of complete EEWS models. In addition, our infrastructure will be used for prototyping software components of the target EEWS. Additional a TN equipped by components of this infrastructure can play a temporary manager of the EEWS to visualise different dynamic installation, maintenance and operating activities.

Some of the concepts will described now more in detail.

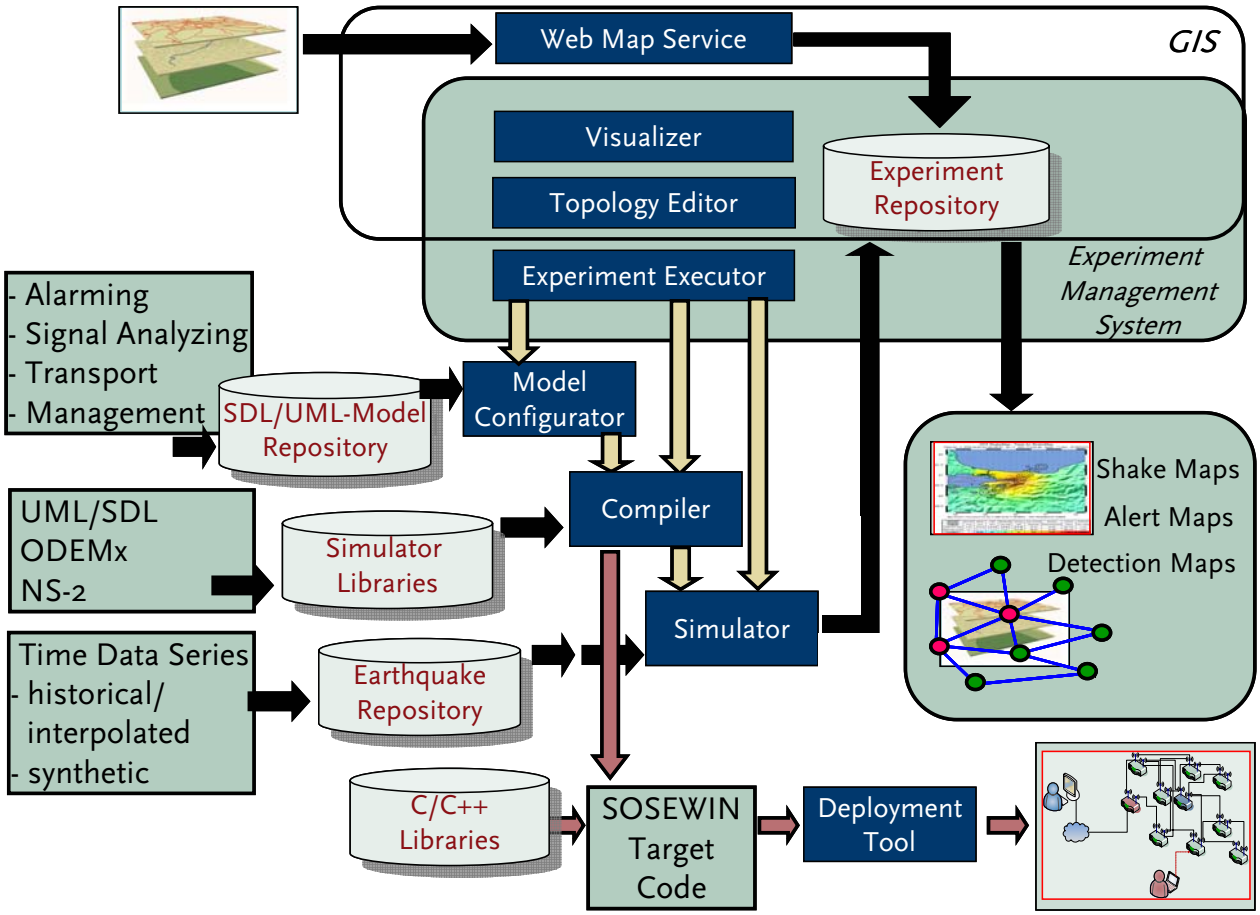


Fig. 7 Model-based Prototyping Infrastructure for EEWS.

6.2 Experiment Management System (EMS)

The results of the simulator runs will also be stored within the relational database *Experiment Repository*, which is part of the *Experiment Management System* (EMS). Experimental results can then be evaluated manually by the *Visualiser*. This tool allows the presentation of a P-wave travelling through the network, with its detection (or non-detection) being marked by the sensor nodes changing colour (green to red, *Detection Map*). Other experimental output would include the so-called *Alert Maps* and *Shake Maps* [12]. Both maps describe the spatial variation in the maximum ground shaking resulting from an earthquake for a given ground motion quality. A shake map is generated from the complete time series of an event for each sensing node. In contrast, alert maps follow an evolutionary approach. Based only on the first few seconds of an earthquake's time series, a predicted shake map is computed. Hence, while alert maps have a

lower quality/accuracy than shake maps, they are generated during an earthquake and are an early warning tool while shake maps are used for post-event response planning.

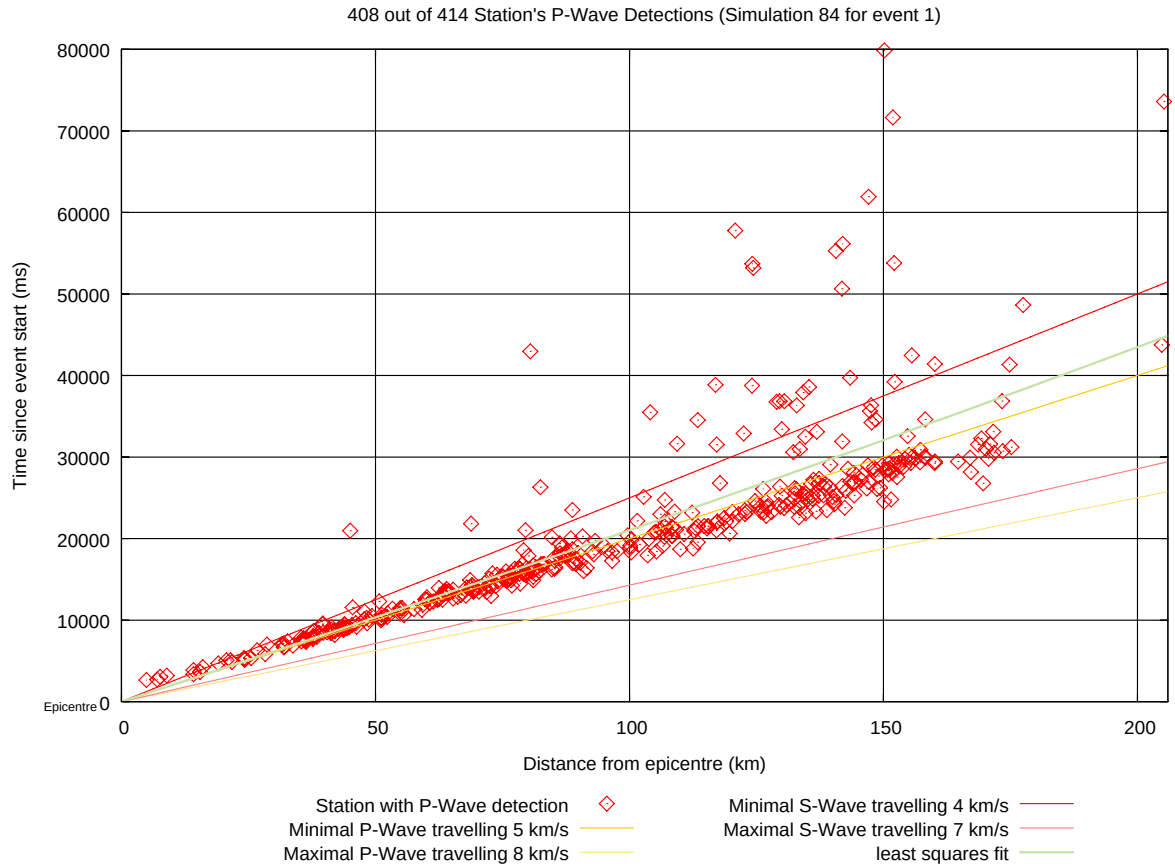


Fig. 8 Time-distance diagram of event detections for ChiChi earthquake (showing simulation 84).

The experimental results can also used to evaluate several event detection algorithms and vary their parameters. Fig. 8 shows a time-distance diagram for a simulation based on the ChiChi earthquake data (simulation number 84). The time-distance diagram is a useful method for quick manual reviews of a simulation's result. The event detection by each station is correlated to a point in time and the distance from the epicentre of the detecting station. The minimal and maximal wave travelling velocities are visualised as linear functions. They form two overlapping planes in the diagram that are actually time windows, one for the P-wave and one for the S-wave.

Based on the time window idea, an automated evaluation and comparison of simulations is possible by counting the detections that are within the P-wave time window.

For the configuration of EEWS models (network topology, software architecture of nodes, geographic area) under load (earthquake events, transmission disturbances) a graphical *Topology Editor* based on a *Geographic Information System* (GIS) is necessary. Adding and removing nodes is implemented using the OGC standard *Web Feature Service* (WFS). With WFS, a layer of spatial objects (e.g. points and lines with additional attributes) defined by the OGC standard *Simple Features for SQL* can be placed in a topographic map (overlay).

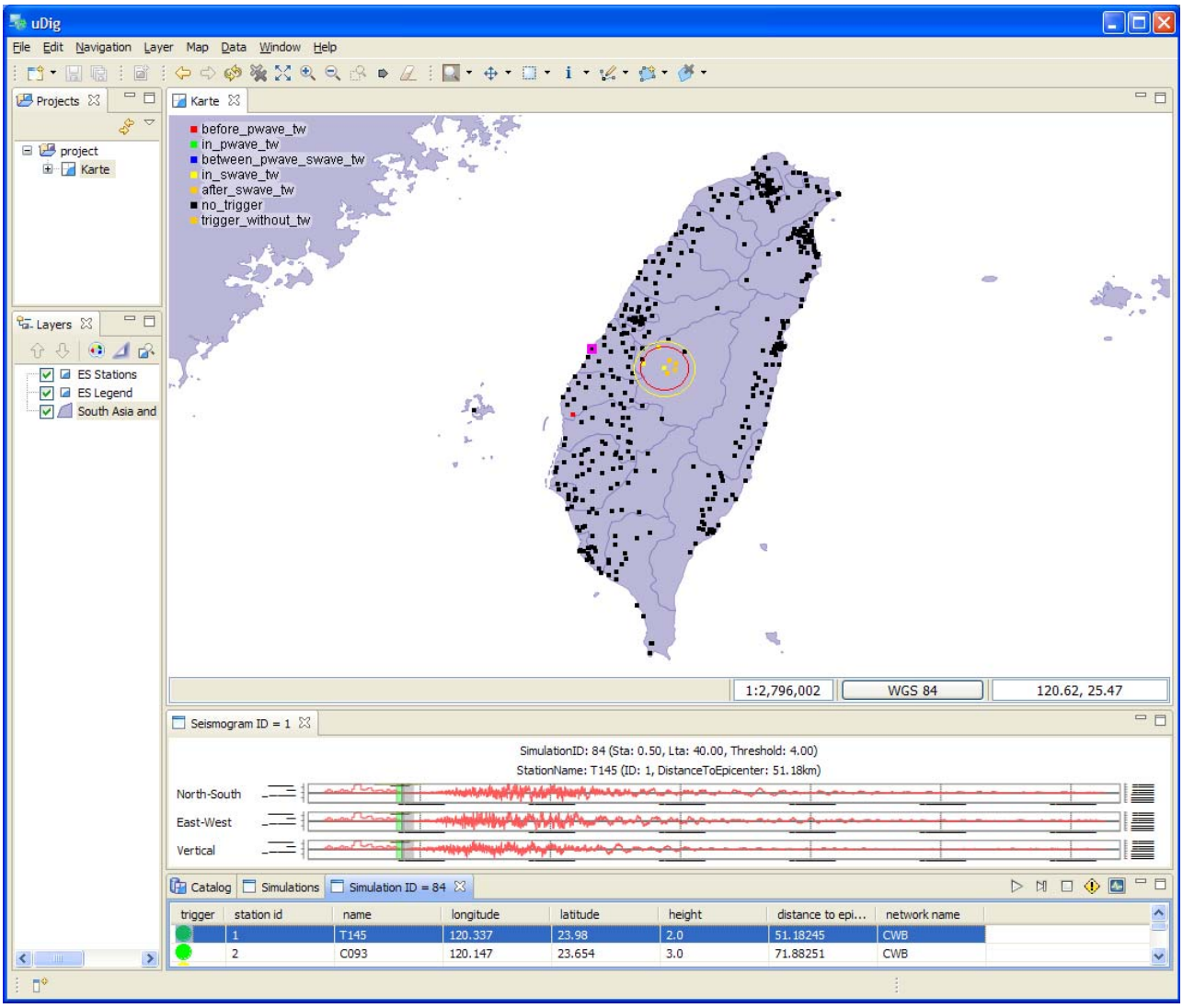


Fig. 9 Screenshot of GIS-based Visualiser and Topology Editor (showing simulation 84).

Fig. 9 shows a screenshot of the GIS-based Visualiser and network Topology Editor displaying the ChiChi earthquake in Taiwan, 1999. The black squares on the map represent the stations of the KNet[10] for monitoring seismic activities. The circles visualise the wave propagation through the network (the outer yellow circle represents the P-wave, the inner red is for the S-wave). After the waves have passed a station, its colour will change according to their detection performance. Seismograms recorded by selected stations can also be displayed, as shown in Fig. 9.

It is planned to extend the Visualiser to show alert and shake maps that are calculated during a simulation.

With our EMS, various automatic evaluations of the experiments can be computed. It considers the seismic wave velocities for a certain area and computes the estimated arrival of the P- and S-wave for each sensing node based on the hypocentre information in the repository. Then it checks the P-wave arrival time as determined by the sensing node, and determines whether this time is within a certain tolerance. In Fig. 9 this is visualised by several colours for the station's squares (see legend in the upper left corner or the first column in the table). Based on that mechanism, our EMS offers a comparison feature to evaluate different experimental results, for example the efficiency of different detection methodologies. Furthermore, it ensures reproducibility and consistency between the various development cycles of the simulator.

Fig. 10 shows a screenshot of the EMS tool. The upper table (green coloured) lists the executed and planned simulations. Simulation number 84 is highlighted, which is the same as depicted in Fig. 9 visualizing the ChiChi earthquake event. The attributes of the simulated event are displayed in the second table (from top).

[10] KNet, Kyoshin Network, National Research Institute of Earth Science and Disaster Prevention, Japan, http://www.k-net.bosai.go.jp.

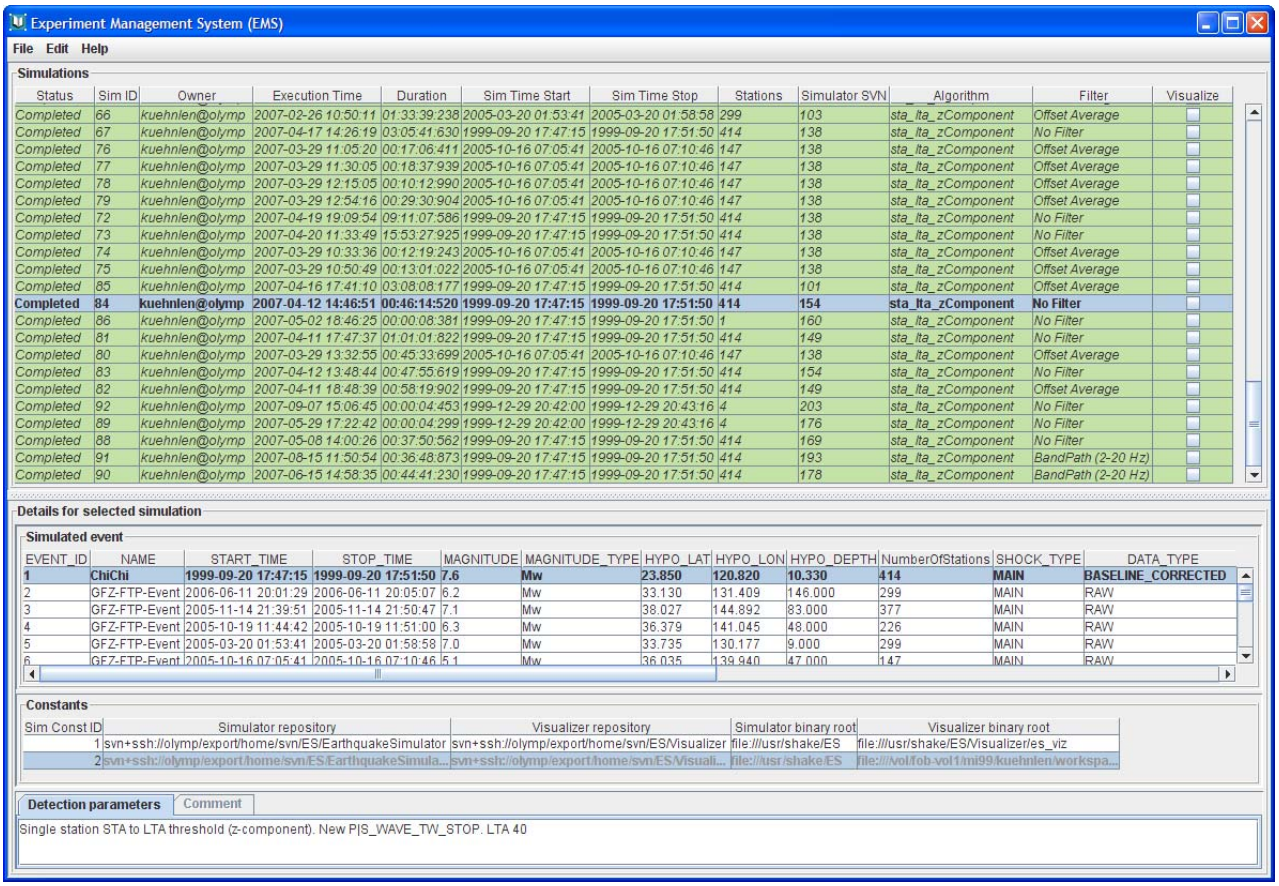


Fig. 10 Screenshot of the EMS tool (showing simulation 84).

6.3 SDL-UML-C++ Transcompiler

By adopting PragmaDev-tools[11] our transcompiler follows the UML MDD approach to produce code for different platforms starting from SDL-RT model descriptions. This allows to process compositions of UML[12] (class, use case and sequence diagrams), SDL (communicating processes) and C++ (data structures and sequential actions). We are able to use the RTDS[13] simulator to debug the model execution by SDL interpretation. In addition to this technique, other simulation frameworks can be coupled according to specific modelling and investigation requirements. Whereas by an SDL-based simulation, so far "only" functional characteristics have been examined, the *ODEMx library* ([17], [18]) will allow non-functional performance characteristics of self-organising systems to be determined by simulation, while varying the topology and environmental influences. The RTDS compiler is currently adopted by following extensions

- annotations (prefixed SDL identifiers) in the SDL-RT source code allow a post-processing of the generated C-code, produced by RTDS,
- additional pattern-controlled transcompiler which transforms the generated C-code of RTDS to C++ supporting different targets. Using these patterns special parts of the structured RTDS C-Code will be substituted in each case following the related substitution patterns. Currently two alternatives are supported by our transcompiler:
  - transcompilation to C++ using the network simulator library ODEMx[14] which also handles time dependencies of state machine actions and message transportations by the network (as a main preposition for a model-based performance evaluation of SOSEWIN networks),

[11] www.pragmadev.com

[12] Standardized by the Object Management Group (OMG).

[13] Real Time Developer Studio (V3.4) supports SDL-RT (a combination of UML, SDL, and C/C++) as a suited representation in the embedded / real time world today because it is basically a set of graphical representations of classical concepts such as tasks, messages, states, timers, and semaphores.

[14] Object-oriented Discrete Event Modelling is a C++ library for modelling and simulation of ensembles of discrete event driven processes combined with time-continuous processes.

– transcompilation by using Boost library thread and network functionality[13] to C++ as target code for the SOSEWIN nodes running a POSIX-compliant Linux.

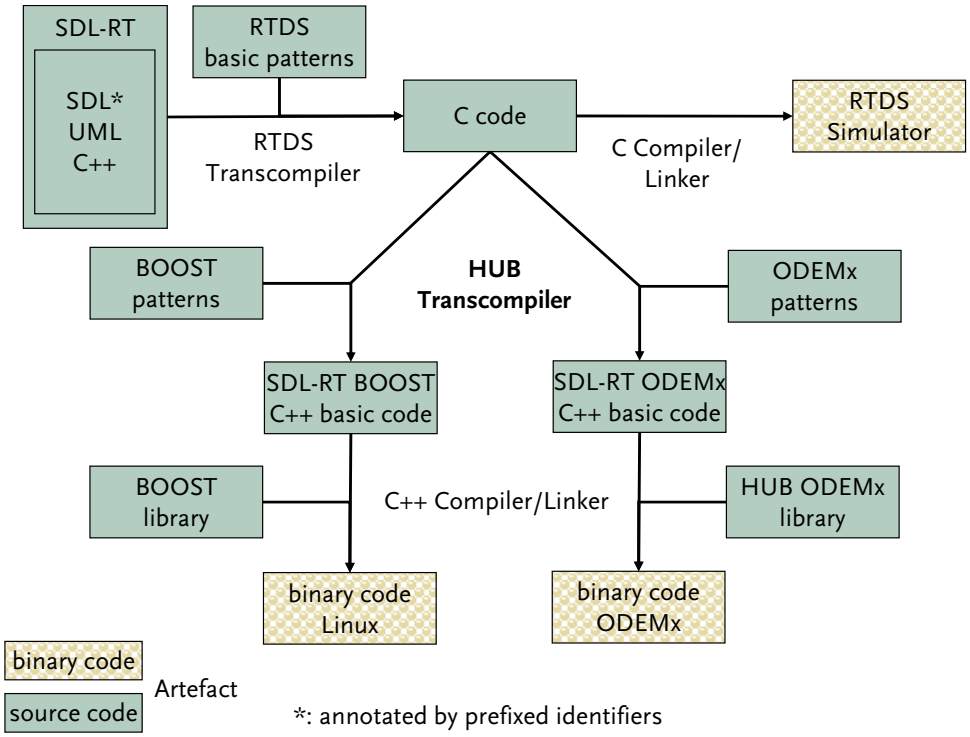


Fig. 11 SDL-UML-C++ Transcompiler Architecture.

6.4 Simulation Framework

Our prototyping infrastructure integrates various simulators for different evaluation goals under common experiment management strategies. The last section has demonstrated different kinds of C/C++ code generations, where two of them were related with the simulation framework.

6.4.1 Simulator-I: Evaluation of Signal Analysing Algorithms

This simulator executes for a given number of SNs and provided time series of sensor's raw data (for each of them) the behaviour of their SAEs without any communication between themselves. With the help of this simulator an isolated test of the signal analysing functionality can be realized. With the EMS topology editor the nodes can be positioned in a map. Using their GPS coordinates a synthesiser of an Earthquake can produce event data individually for each node by fixing a hypocentre and the earthquake parameters (e.g. rupture length, depth, energy). The simulator visualises on one hand side the distribution of the earthquake waves in dependence of time and on the other side the P-wave detection by switching a virtual light controller from green to red by each of the node (Detection Map).

So, the simulator allows to evaluate several event detection algorithms and to vary their parameters. Fig. 8 shows a time-distance diagram for a simulation based on the ChiChi earthquake data (simulation number 84). The time-distance diagram is a useful method for quick manual reviews of a simulation's result. The P-wave detection by each station is correlated to a point in time and the distance from the epicentre of the detecting station. The minimal and maximal wave travelling velocities are visualised as linear functions. They form two overlapping planes in the diagram that are actually time windows, one for the P-wave and one for the S-wave.

Based on the time window idea, an automated evaluation and comparison of simulations is possible by counting the detections that are within the P-wave time window.

6.4.2 Simulator-II: Functional Evaluation of AP Entities (SDL-Systems)

Here we use the RTDS SDL-RT-simulator to test the functional behaviour of smaller ensembles of the SOSEWIN nodes. We abstract from concrete earthquakes, and underlying protocol layers. One further important preposition is a perfect transmission behaviour of used communication channels over the air. The results of functional tests allow us to

evaluate and improve the logic of our alarming protocol. Typical outputs here are MSCs, which can be represented as XML and also stored in the experiment repository for further filtering by using data base functionalities.

Fig. 12 visualises the issuing of a system alarm by a SOSEWIN-SDL-model with four node clusters (groups) and three nodes per group. The figure shows a shortened Message Sequence Chart (MSC) containing six lifelines per group: one for each SE, one for the active LE and two for the idle Leading Nodes (SN with idle LE). The most left lifeline pSimulation, takes the role of an SAE and sends signals for detecting the P-Wave directly to the corresponding SE of a SN. The next two lifelines are for organising the simulation and can be filtered out.

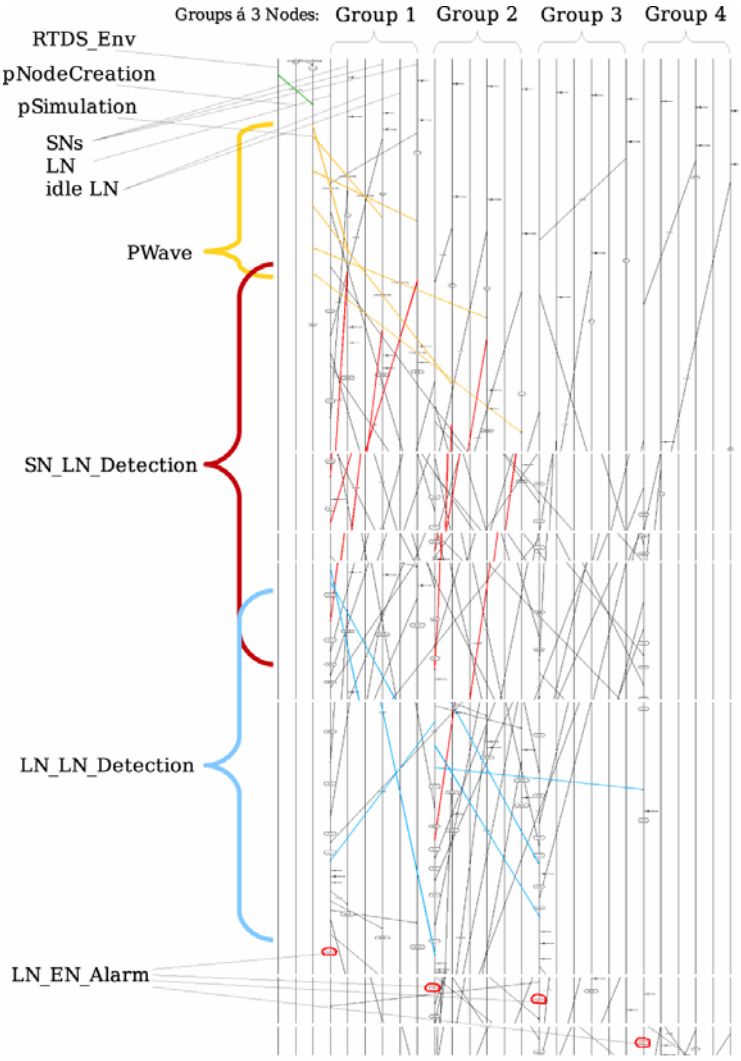


Fig. 12 Alarm issuing by an example SOSEWIN.

6.4.3 Simulator-III / IV: Performance Evaluation of SDL-Systems

Here we use the capability of our general-purpose ODEMx library [21], which supports the modelling and simulation of parallel process, where their state changes are described by discrete events in combinations with differential equations. This library contains especially concepts for simulation computer networks, where the protocol entities are extensions of the built-in ODEMx process concepts. Using this library two different simulators are produced by our transcompiler technology:

- Simulator III allows the estimation of required transmission times and transmission quality of alternative SOSEWIN configuration which guarantees the early warning functionality in dependence on different earthquake scenarios.
- Simulator IV should support in extension of SIMULATOR III the simulation of node breakdowns and of the behaviour of underlying protocol layers. Especially this simulator could be used for the training of disaster's management experts.

Alternative to Simulator IV a NS2-Simulator [14] could eventually be used similar to the known SDL-NS2 approach from [26]. For that purpose our transcompiler technology has to be extended before.

All simulators produce MSCs and other event traces for a further information aggregation or visualisation. To simplify this kind of operations the trace raw data are managed by our experiment repository, realized as a data base system.

6.5 Earthquake Repository

The *Earthquake Repository* is implemented by a relational database system holding historical or synthesized time series of ground motion in the same format as a real sensor would provide. GFZ has suggested selecting those earthquakes with a magnitude greater than 5 and a close distance (< 200 km) between the network and the hypocentre. Currently the repository comprises 56 earthquakes, including the well-known ChiChi event in Taiwan and ten events in Japan. Both countries provide a dense and high-quality monitoring network. The other 45 events in Europe are recorded by only a few stations and the data is of a lower quality.

The synthetic seismograms are generated by the method of Wang (1999) [15]. This involves the use of a computationally efficient and stable propagation method applying orthonormalization for the calculation of the Green's functions of a half-space consisting of an arbitrary number of laterally homogeneous layers with a stress-free surface on top and the homogeneous deepest layer extending to infinite depth. From these functions, the ground motion resulting from a defined seismic event, or combination of events, is determined.

Synthetic seismograms offer the opportunity to test different methods of event detection and classification, with the freedom to introduce as much (or as little) “noise” to the data as required. Likewise, different configurations of seismic stations can be assessed to determine which is the optimal array geometry, or on the other hand, to determine the limitations of a given network that is bounded by certain practical considerations (terrains, buildings, etc.).

7 Current Status – A Prototype for Istanbul

Besides an existing small laboratory testbed of ten SOSEWIN nodes at Humboldt-Universität, the main field test was planned to be in Istanbul. In order to establish a small (about 40 nodes) network in the city of Istanbul, in April 2008 a scientists group performed an inspection of the Ataköy area in the Bakirköy district.

During the inspection, two communication tests with SOSEWIN nodes were performed, in order to verify the quality and distance capability of the communication between nodes when they operate into the urban context.

During the first test the quality and strength of communications between the base station and three buildings at 80m, 140m, and 180m were measured, while at the second base station a building at the considerable distance from the base station of 260m was tested. The positive results obtained from these preliminary tests make possible the planning of the SOSEWIN test network in June 2008 (Fig. 13).

8 Conclusion

We have presented a prototyping infrastructure for the model-driven development of EEWS based on self-organising sensor networks. This architecture is based on OGC, OMG and ITU-T standards and combines different technologies GIS, databases, behaviour modelling, code generation and simulation technologies for special application domain by one integrated framework. So, it allows the evaluation of the real-time behaviour of projected earthquake monitoring and alarming systems and supports automatic code generation from evaluated structure and behavioural models. Modelling techniques which we used here are based on SDL and UML under special real-time requirements. Our prototyping infrastructure, implemented in C++, is used in the projects SAFER and EDIM for optimizing self-organising seismic earthquake early-warning and rapid response systems, a real testbed is in preparation for Istanbul.

An evaluation of the real-time behaviour of such complex systems is almost impossible or too expensive without prior modelling experiments, involving computer simulations. For that we identified several investigation goals supported by different simulators. This involves functional and performance evaluation of EEWS models by tuning topologies and parameters. Additionally to the model-based development our prototyping infrastructure supports also the installation, test, and operating of the network.

Fig. 13 Planned SOSEWIN network in Istanbul

Currently the concepts of a cooperative signal analysing are tested and experiments to evaluate and improve performance characteristics will start very soon after stabilisation of our compiler technology.

Although this contribution is naturally focussed on earthquake driven applications, the presented architecture of prototyping system may be adopted to those use cases where meshed sensor-based self-organising infrastructures in combination with GIS are applied, such as in Heat Health Warning Systems [19].

Bibliographic References

[1] M. Wieland "Earthquake Alarm, Rapid Response, and Early Warning Systems: Low Cost Systems for Seismic Risk Reduction", *Electrowatt Engineering Ltd. Zurich*, Switzerland 2001.

[2] R. M. Allen, H. Kanamori "The Potential for Earthquake Early Warning in Southern California", *Science*, 300, 786-789, 2003.

[3] M. Kurth, A. Zubow, J.P. Redlich "Multi-channel link-level measurements in 802.11 mesh networks", *IWCMC '06*, Canada 2006.

[4] http://www.sdl-rt.org, 2008/05/28.

[5] M. Erdik, Y. Fahjan, O. Ozel, H. Alcik, A. Mert, and M. Gul. "Istanbul Earthquake Rapid Response and the Early Warning System", *Bulletin of Earthquake Engineering*, 1(1):157–163, 2003.

[6] E. Weber, et al "An Advanced Seismic Network in the Southern Apennines (Italy) for Seismicity Investigations and Experimentation with Earthquake Early Warning", *Seismological Research Letters*, 78, 622-634, 2007.

[7] W. H. K. Lee, T. C. Shin, and T. L. Teng. "Design and implementation of earthquake early warning systems in Taiwan", *Proc. 11th World Conference on Earthquake Engineering, Acapulco, Mexico*, Oxford, England: Pergamon, Disc 4:2133, 1996.

[8] "Optimized Link State Routing Protocol (OLSR)", RFC – http://ietf.org/rfc/rfc3626.txt.

[9] http://www.iris.washington.edu/data/dmc-seedlink.htm, 2008/05/28.

[10] J. Fischer et. al. "A Run Time Library for the Simulation of SDL'92 Specifications", Proc. of the 6th SDL Forum.

[11] ITU-T X.690 "ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER)" *International Telecommunication Union*, 1992.

[12] D. J. Wald, B. C. Worden, V. Quitoriano, and K. L. Pankow "ShakeMap Manual; Technical Manual, Users Guide and Software Guide", *U.S. Geological Survey*, version 1.0 6/19/06 edition, 06 2006.

[13] http://www.boost.org, 2008/05/28.

[14] http://www.isi.edu/nsnam/ns, 2008/05/28.

[15] R. Wang "A simple orthonormalization method for stable and efficient computation of Green's functions", *Bulletin of the Seismological Society of America*, 1999, 89, 733-741.

[16] J. Fischer, T. Neumann, A. Olsen ”SDL Code Generation for Open Systems”, In: *12th SDL Forum “SDL 2005: Model Driven”*, A. Prinz, R. Reed, J. Reed (editors), *Springer LNCS 3530*, 2005, pp. 313-323.

[17] J. Fischer, K. Ahrens. “Objektorientierte Prozeßsimulation in C++“, *Addison-Wesley Publishing Company*, 1996.

[18] R. Gerstenberger “ODEMx: Neue Lösungen für die Realisierung von C++-Bibliotheken zur Prozesssimulation“, *Diplomarbeit Humboldt-Universität zu Berlin*, 2003.

[19] W. Endlicher, G. Jendritzky, J. Fischer, and J.-P. Redlich “Heat waves, urban climate and human health”, In: F. Kraas, W. Wuyi, and T. Krafft, editors, *Global Change, Urbanization and Health*, pages 103–114. China Meteorological Press, 2006.

[20] E. Weber, V. Convertito, G. Iannaccone, A. Zollo, A. Bobbio, L. Cantore, M. Corciulo, M. Di Crosta, L. Elia, C.Martino, A. Romeo, and C. Satriano “An Advanced Seismic Network in the Southern Apennines (Italy) for Seismicity Investigations and Experimentation with Earthquake Early Warning”, *Seismological Research Letters 78(6)*, 622-634, 2007.

[21] http://sourceforge.net/projects/odemx, 2008/05/28.

[22] http://www.saferproject.net, 2008/05/28.

[23] ITU-T Z.100 “Specification and Description Language (SDL)” *International Telecommunication Union*, 2002.

[24] http://www.pragmadev.com (RTDS V3.4), 2008/05/28

[25] R. Kluth, J. Fischer, K. Ahrens: “Ereignisorientierte Computersimulation mit ODEMx”. *Informatik-Bericht N. 218*, Humboldt-Universität zu Berlin, 2007.

[26] T. Kuhn, A. Geraldy, R. Gotzhein, and F. Rothländer “ ns+SDL – The Network Simulator for SDL Systems”, In: *12th SDL Forum “SDL 2005: Model Driven”*, A. Prinz, R. Reed, J. Reed (editors), *Springer LNCS 3530*, 2005, pp. 103-116.