Project number:	European Commission - 033685
Project acronym:	CHORIST
Project title:	Integrating Communications for enhanced environmental risk management and citizens safety
Instrument:	Integrated Project
Thematic priority:	Information Society Technology
Call identifier:	FP6-2005-IST-5
Start date of project:	01/06/06 **Duration:** 38 months

Deliverable reference number:	SP2.D4
Deliverable title:	SP2 training system definition and design
Version:	1.1
State within Consortium:	DRAFT: - FOR APPROVAL: - APPROVED: **X**
Due date of deliverable:	MONTH 18 (11/07)
Actual submission date:	15/10/09
Lead contractor of this deliverable:	JRC
Other contributing contractors:	-

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
DISSEMINATION LEVEL		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

CONTENTS

1 INTRODUCTION

1.1 PROJECT SCOPE

The CHORIST project will propose solutions to increase rapidity and effectiveness of interventions following natural hazards and industrial accidents, in order to enhance citizens' safety and communications between rescue actors.

1.2 PURPOSE OF THE DOCUMENT

This document presents the structure and data flows of the training system of the CHORIST SP2 (Risk assessment report systems).

1.3 DOCUMENT VERSIONS SHEET

Version	Date	Description, modifications, authors
1.0.1	16/01/07	Document template
1.0.2	03/09/08	First version
1.0	18/12/08	Release
1.1	15/10/09	Typos corrected & set to PUBLIC

Table 1 : Document versions sheet

2 REFERENCE DOCUMENTS

2.1 REFERENCE DOCUMENTS

[1] CAP reference: http://www.oasis-open.org/specs/index.php#capv1.1

[2] Efficient XML Interchange Working Group Public Page: http://www.w3.org/XML/EXI

[3] TSO: Definition of the OASIS Tactical Situation Object and Data dictionary of the OASIS Tactical Situation Object in: http://www.oasis-fp6.org/documents.htm

[4] WAP Binary XML Content Format: http://www.w3.org/TR/wbxml/

[5] XML Binary Characterization Working Group Public Page: http://www.w3.org/XML/Binary

[6] XOP Recommendation: http://www.w3.org/TR/xop10/

[7] SP2 input protocols study (CHORIST-SP2.D51-V1.0.doc)

[8] ERAW & ERAW System definition and design (CHORIST-SP2.D1-V1.0.doc)

[9] SP2 training system scope and methodology (CHORIST-SP2.D3-V1.0.doc)

2.2 DEFINITION

2.3 ABBREVIATION

CAP	Common Alerting Protocol
ERAS	Environmental Risk Assessment (system)
ERAW	Environmental Risk Awareness (system)
NA	Not Applicable
TSO	Tactical Situation Object
UDT	User Data Terminal
W3C	World Wide Web Consortium
WBXML	WAP Binary XML
XML	Extensible Markup Language
XOP	XML-binary Optimized Packaging

3 TRAINING PROCESS

When talking about training for SP2 users, it is easy to define two different cases. Training can be provided to users who do not know the system, or to users whose proficiency in using the system should be evaluated and kept tuned.

The tools that will be provided will satisfy the needs of both cases. They have been developed bearing in mind a simple model of training activities, where one or more trainers can access a fake replica of a real environment. The training tools will then act as the real environment does i.e. providing all the input needed to simulate a real case.

We suggest providing trainers with classrooms that have a work station for each trainee, with all the proper software installed and connected to the SP2 system. The desktop of the trainer's workstation should capable of being projected on a big screen, in order to show what operators should do and when. The training system will be installed on the trainer(s)'s workstation, thus allowing him/her to profile the simulation as needed.

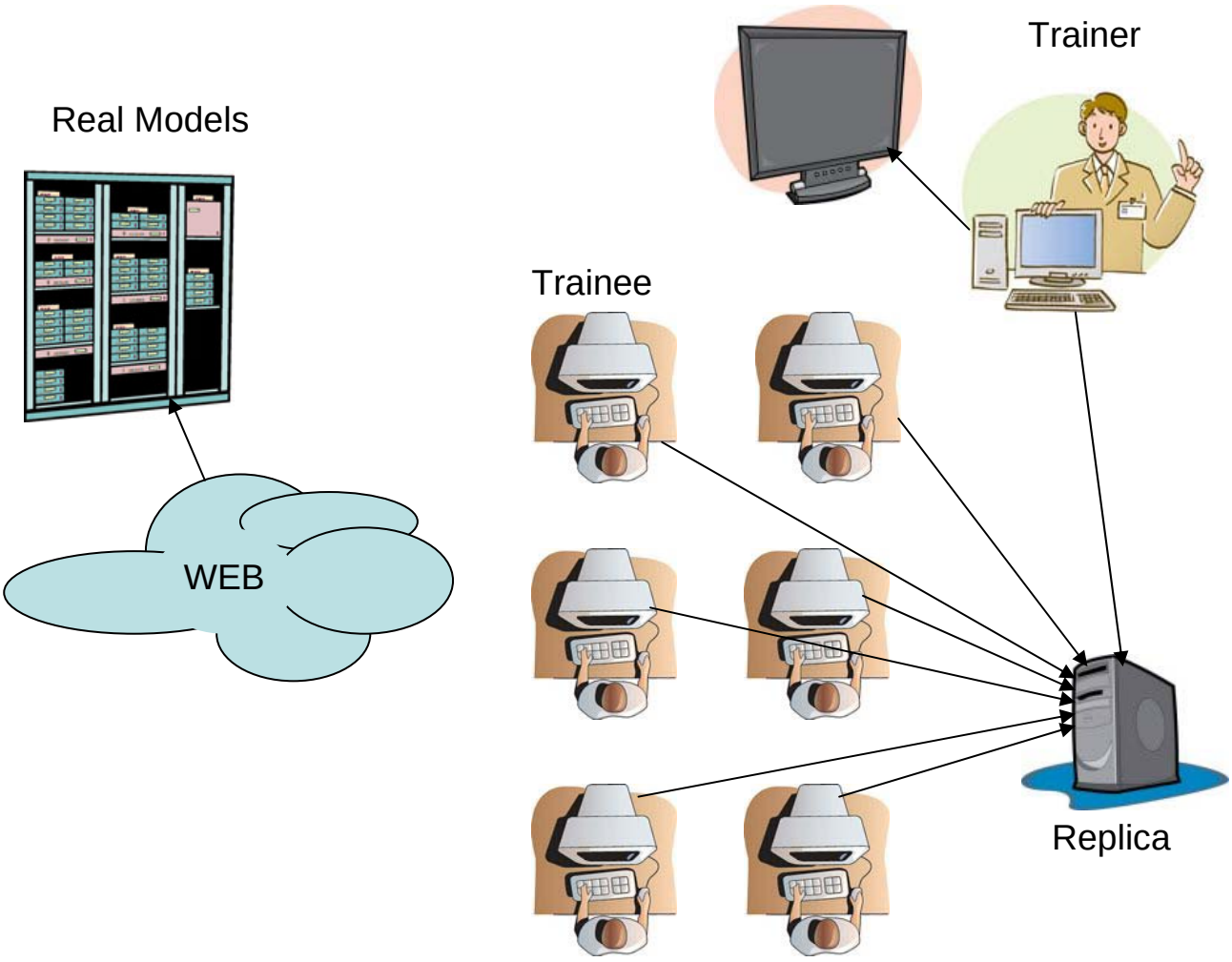


Figure 1Training class configuration

A small group of operators, possibly heterogeneous, will be in a classroom. The first part of training will show the different tools, and how to use them. 112 operators will see how the messages they compose are delivered to ERAW/ERAS components of the system, and how they are displayed to colleagues. This will help in creating policies to input valuable and effective information.

ERAS operators will then be shown how to handle event notifications, and how they can obtain more information about them and the areas they involve. This necessitates a lot of interaction with other systems that do not need to be replicated e.g. geo processing facilities that are provided externally. Simulators will also be used to generate information about possible natural or artificial events.

At last the operators will be asked to determine the need of propagating alerts to other systems and how.

To help trainees relate to reality, simple scenarios are provided to follow step by step the flow of information in SP2 system.

After the trainees are shown how to use the system, the trainer can launch a simulation and test the trainees' understanding.

Simulations can be based on one single scenario or can combine more than one scenario, providing more realistic though complex situations. The trainer should adopt the proper configuration based on trainees' proficiency and interests.

Simulations can also be used to perform drills and keep operators and procedures up-to-date.

In this case, complex real-time simulations can be used. Trainees should be separated, allowing them communicating only through the system. Each group will be led by a trainer, to evaluate trainees' response and to give advice.

4 TRAINING SYSTEM AND TOOLS

In order to run training sessions that simulate reality, SP2 systems will be as similar to those that users use in the real world as possible. It is necessary therefore to create a tool that can feed the systems as their environment will do on the field. The use of the CAP protocol for messaging facilitates the creation of tools capable of posting alert messages just like is the case with real world systems.

The Training system will be composed of the training tool itself and by simulators. Simulators will function as information providers, acting as sensor grids should, providing a series of data e.g. water levels in a flash flood. The training tool allows for information to be fed that typically originates from human actors like 112 calls etc..

4.1 SP2 INPUT INFORMATION PROVIDERS

The actors and entities providing information to the SP2 are classified in 4 categories:

The actors and entities providing information to the SP2 are classified in 4 categories:

- The **citizens** (or more generally, the population) are a source of information for SP2. The types of information they provide are either information concerning present or soon to occur disasters **through emergency call centres** like 112 call centres, or information concerning past disasters. The latter source can be collected through studies and is valuable for identifying the nature and probability of disasters in order to improve knowledge on the associated the risks and vulnerabilities; however these are not useful for the CHORIST project, as this information relates to recovery phase of a disaster event, and not to the early warning - the objective of the CHORIST project. Thus, the CHORIST project will only focus on alerts coming from emergency call centres and resulting from citizens' calls.
- **National or Regional Environmental and Industrial Risk Survey Agencies** monitoring a specific type of disaster (such as Meteorological and/or Hydrological Agencies, Geological Survey Agencies…) are of paramount importance as they already exist and they provide substantial and reliable information. They provide two kinds of information: (1) situation information and (2) alerts. The situation information, such as meteorological current or forecast maps, is useful to evaluate the impact of a disaster. The sharing of these data between the legacy agency and CHORIST is not studied in this document (but probably in the ORCHESTRA project). This document will focus on **alerts** issued from these Agencies.
- **Public Safety Members** (e.g. Firemen, Policemen, Forest Rangers) may also raise alerts or provide new elements (updates) to the SP2 concerning disasters that are already being monitored or that will presumably soon occur. .
- And a wide variety of **sensors**:
    - Some **continuous sensors** provide "real-time" data which are injected in software running models to forecast disasters. For example, weather radars can provide raw information on precipitations which is used for flood forecast; seismographs can help to forecast tsunamis by detecting earthquakes. Other continuous sensors are used to follow the development and displacement of phenomena to vulnerable sites, satellite data for dusts/sands storms monitoring.
    - Other sensors raise alerts only when the values they measure exceed some predefined thresholds. For example, there are some sensor devices collocated with sirens, which automatically raise alerts for tsunami threats.

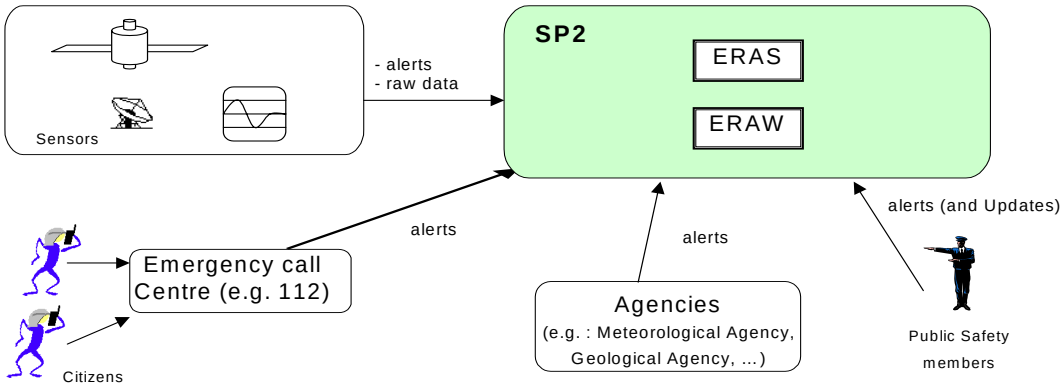


Figure 2: SP2 input information providers

The training tool will act as the first three categories, while simulators will feed data for the last category.

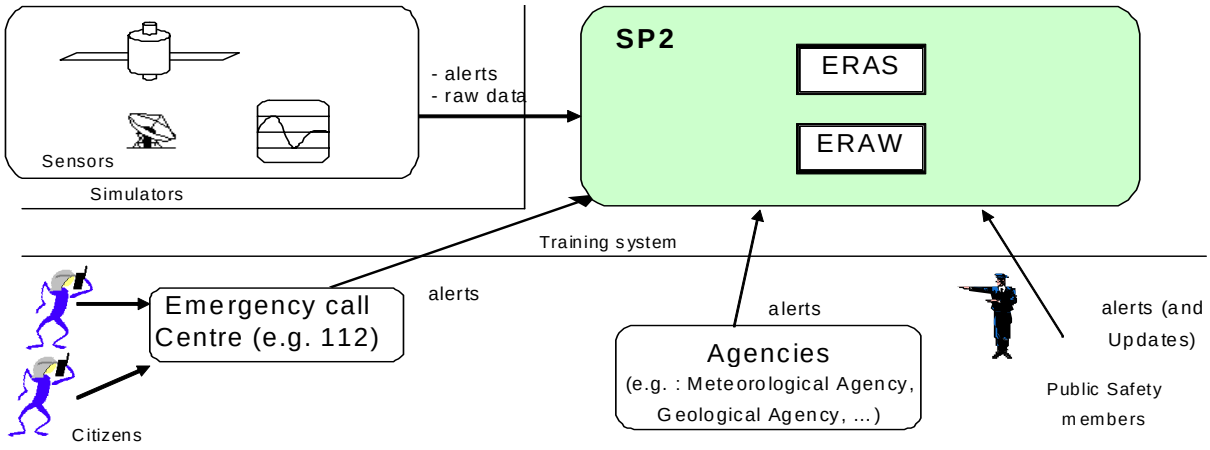


Figure 3: SP2 training system interaction model

4.2 SP2 TRAINING TOOL

The training tool will deliver messages to ERAW following a scenario description, which is composed of steps. The steps are organized in a linked structure and the sequence is established by means of starting conditions. When the starting condition of a step evaluates to true, the step is started only once. Starting conditions can be different in type, and be based on the status of other steps or they can be time dependent. Conditions can also be combined in more complex conditions.

The trainer will load the simulation configuration and browse it in a graphical display. Parameters of each step can be changed at any moment, even while the simulation is in progress. This will not affect completed steps, but can alter the behaviour steps that are in progress or those that are soon to start.

The user will then be able to compose new messages as a 112 operator would, in order to better tailor the simulation by adding new data.

The system will be able to submit messages to the ERAW by using its standard interfaces, i.e. by posting CAP messages to it. Actions are matched by steps, since it is feasible that other actions can be useful to improve the simulation e.g. a file being moved to be published on a web site to simulate a news broadcasting service giving additional or wrong information on the crisis.

4.3 SP2 SIMULATORS

The ERAW should also automatically receive retrieved information like that coming from sensors array or scraped sites. To improve the realism of simulation, this also must be simulated. Keeping this task separated by the main tool allows a simpler user interface, and a more flexible structure, since new and totally different simulations can be added without changes to other software.

Simulators will then be separated from software components and configured separately from the main tool and their simulations launched at about the same time as main simulation.

They can vary in form and implementation, since they must only adhere to exchange interface as stated in [7].

First implementation of such a simulator is the Flash Flood simulator. It simulates the data collected by an array of sensors along the basin of a river that suddenly floods its basin.

At scheduled (and configurable) time intervals, a set of data is generated and collated in CAP messages that are sent to a configured ERAW system.

5 SP2 TRAINING TOOL

This component is aimed at providing a trainer the means to simulate all data exchange that should happen in case a crisis is monitored by SP2 systems.

It will allow for simulating the flow of information coming from people based on a scenario. A scenario definition will be a set of document and information to be provided at appropriate times to trainees.

The tool will give constant feedback on simulation progress in a graphical way that allows user interaction, thus changing simulation parameters even at runtime.

5.1 COMPONENTS

The Training tool is composed mainly of two components: a GUI and the simulation engine. While the GUI is a normal Windows™ application, the simulation engine is a self contained component, which runs separately from the GUI.

When the application loads, an engine is instantiated and configured, then the GUI will draw a graphical representation of engine configuration from its data.

The GUI can interact with the engine requesting modification of the parameters of single steps, and requesting all information about them.

When ready, the engine is started, and the GUI provides graphical feedback about its activity. In order to improve user experience, the GUI and the engine run in separate threads. This gives a prompt and responsive user interface, and guarantees that user interaction will not alter simulation progress. In case of simultaneous steps, more threads are started.

Please, refer to Figure 4 GUI engine interaction for clarification.

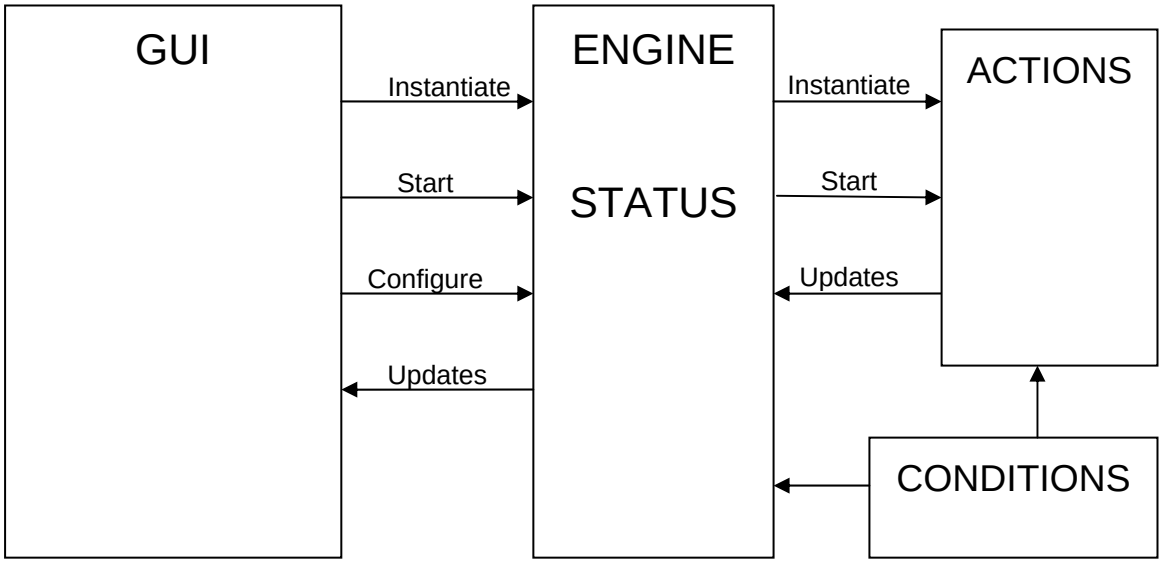


Figure 4 GUI engine interaction

5.1.1 Simulation engine

The scenarios are defined by steps the engine will process at appropriate times. Every step is defined by the moment it should start. This can be decided because a given amount of time is passed from simulation beginning or other steps have been completed or started or a combination of these.

Conditions are used to define the moment the step should be performed. Performing a step means performing the action related to it: this is implemented by starting a new thread, which performs the action.

At this moment the step gets *Started* status.

Actions can be one-shot or ongoing. One-shot actions are performed just once, then the step status is marked as *Completed*. Ongoing actions perform at least once, then they check for completion conditions: if reached, the action stops, and the step gets *Completed* status, otherwise the action waits for one second, before performing again and repeating the conditions check.

In case of error, the action is stopped and the step gets an *Error* status. All steps dependent on this step completion are not started.

Every step has an action to perform. It can be an interaction with SP2 like posting a CAP message, or an interaction with trainees like publishing news on a web site.

After the action is completed, the engine will be notified in order to once again evaluate the list of actions to be performed and to update the GUI.

Refer to **Erreur ! Source du renvoi introuvable.** for the format of following information in the database.

5.1.1.1 Step

Each step is defined by its id, a number that identifies it uniquely, but has no order meaning. A step is therefore defined by:

- Id: unique numeric identifier
- Name: short description
- Description: long lore about the step
- Action: the type of action to be performed like post, move or even no operation
- Source: Url of the information to be transferred to SP2 system and users
- Destination: Url of the place that will receive such information, like a folder for a file or the ERAS service for a CAP message

At the moment implemented actions are only for moving content from a folder to another (e.g. publishing on a web site), MOVE action, or to post a content to a Url (e.g. posting to the ERAS system), POST action. A *no operation* action is provided to implement pauses.

5.1.1.2 Condition

Each condition is linked to a step, but it can be used in different ways. Its definition is:

- Id: the id of the step the condition is related to
- Type: start or end, to mean the way the condition will be used - start conditions are used to know if the step is ready to be performed, end conditions are used to know if the action should be performed again
- Operand: the information the condition is based one - time to mean the number of seconds since the simulation started, while a positive number means the id of a step whose status will be used in comparisons
- Operator: a comparison symbol like =, > or <

- Value: a numerical value to be compared with the *operand* by means of the *operator*

When conditions are loaded from the database, they are divided in different sets: the engine uses the starting conditions to decide if a step must be started, and actions use ending condition to stop their activities.

5.1.2 Graphical User Interface

While the engine processes its configuration, performing actions as required, the user can interact with the GUI. This allows inspecting the configuration, the status of each single step and the results of corresponding actions. In this way, the trainer can change the behaviour of the steps that are not yet completed. The published file can be changed or an ongoing action can be asked to last for alonger or much less time.

Another possible interaction is a form which allows the trainer to insert a message akin to what a 112 operator would do. This will be posted to the ERAW immediately or after a short delay.

5.1.3 Main Window

The main window of GUI presents the chain of steps to be performed and a panel to display the details of each step; please see Figure 5 Main GUI window.

The start button is used to start the simulation engine, which can be stopped with a pause button, allowing the simulation to continue later, or with a Reset button, that reverts all steps to their initial status.

The progress of the simulation is displayed in the progress bar on the top, while a clock is updated to display simulation time. As each step changes its status, it is displayed in a different colour.

The steps are arranged in the picture as close as possible to their correct time reference. The time reference of a step is the minimum time after the simulation started, that should pass before the step is performed. It means that it is computed as the maximum of its starting time (if defined by a time dependent starting condition) and the time references of all the steps it depends upon, if any.

Let Step A be bound to start after 10 minutes of simulation time and Step B be bound to start after Step A completion and 5 minutes of simulation time. If Step C is bound to start after Step B start and 7 minutes of simulation time, its time reference computes to 10 minutes of simulation time, because of time related A starting condition, which prevails upon B and C time related starting conditions.

When a step is performing its action, or after it completes, its data cannot be changed. Before being performed, or when the simulation is reset, the steps' data can be altered, thus allowing changing the simulation definition.

This is the same mechanism used when defining the simulation. Please see **Erreur ! Source du renvoi introuvable.**.

While running the simulation, the trainer could wish to add information by sending them to ERAS system as a 112 operator would.

This can be done via the 112 form, raised by Post message button. See 5.1.4.

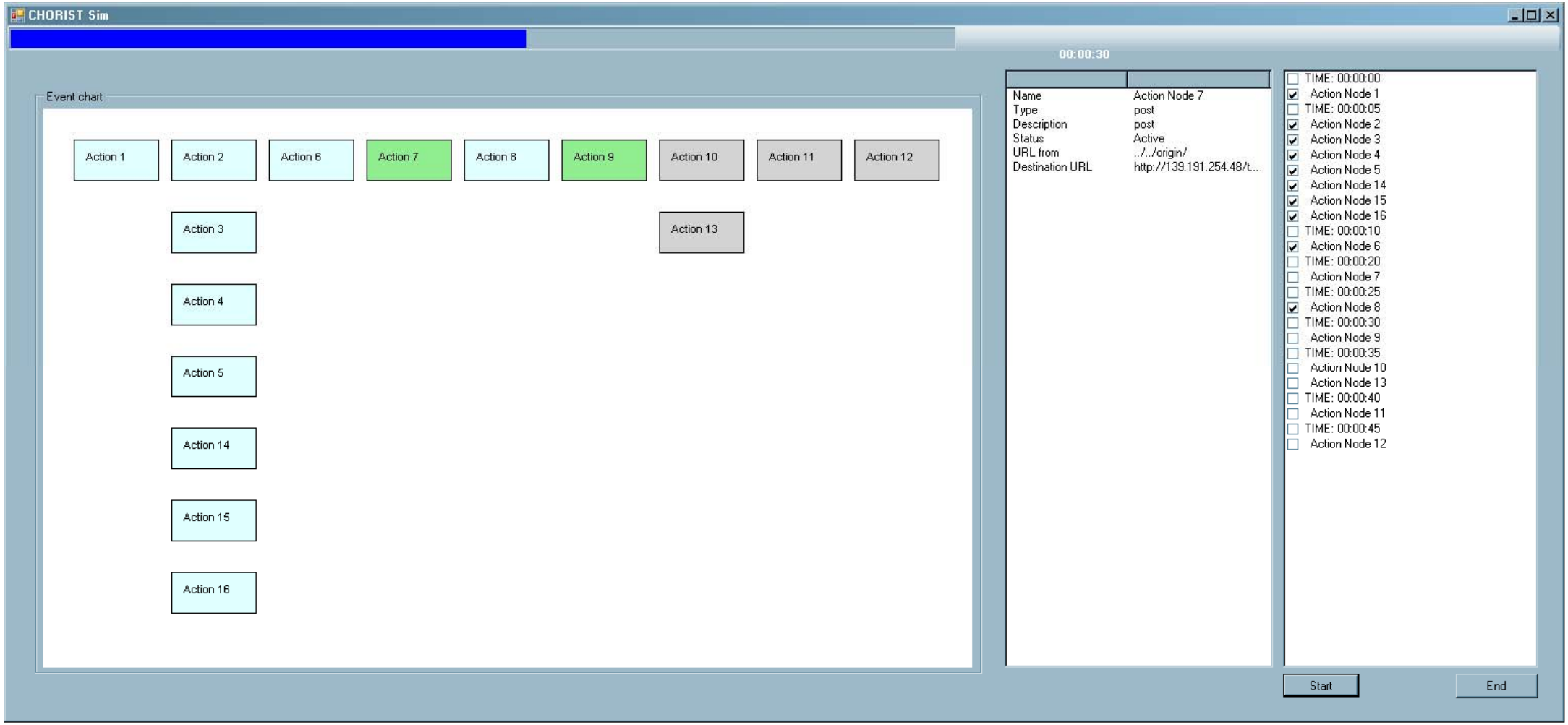


Figure 5 Main GUI window

5.1.4 112 form

Figure 6 112 Form

By means of this form, the trainer can add messages to the flow of information delivered to the ERAS service.

The trainer will enter all relevant information and finally he will request the message to be delivered immediately or after a given delay.

The form mimics the 112 form, thus allowing inserting the data about the nature of the alert and its assessment. Information about the location of the event can then be added.

This form can be used also in scenario definition, since it allows saving the CAP messages for subsequent use.

The scenario definition is stored in a set of tables in a database. It is based on steps definition. Each step is defined by its starting condition and the action it must perform.

The structure is therefore fairly simple, as described in Figure 7 Database schema.

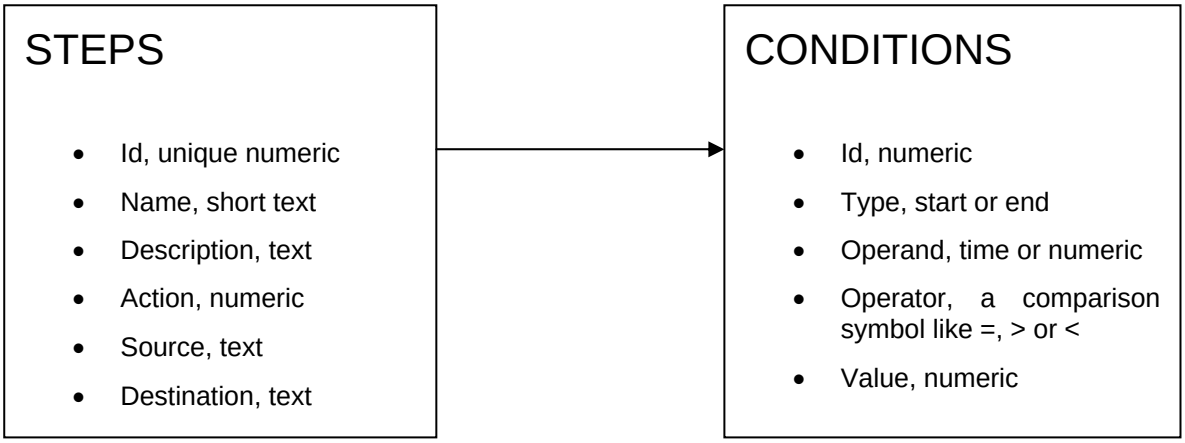


Figure 7 Database schema

The Training Tool allows for building and configuring a simulation from scratch.

It simply requires to click on Add Step, to create a new step, that can be configured properly in terms of action and conditions. The configuration can then be saved to the database.

6 SP2 SIMULATORS

Simulators should be standalone systems capable of simulating the continuous feed from automatic sources like sensor grids or scraped web sites. This means that this kind of application should input in SP2 system the same kind of data that should come from an automatic source during a crisis. The data format is CAP messages, as usual.

At present, the only simulator available is the Flash flood simulator. We will see briefly how it works.

As the training tool, it is composed of a GUI and a simulation engine. The GUI serves to instantiate an engine and to monitor its work, therefore the engine is a standalone component easily usable in other contexts.

The Flash flood simulator GUI is composed of a main panel and several sensor panels, which are used to monitor the signal coming from each sensor.

The main panel contains a graph of the wave flowing downstream, and all parameters configuration – providing controls for the user to start and manipulate the simulator as appropriate.

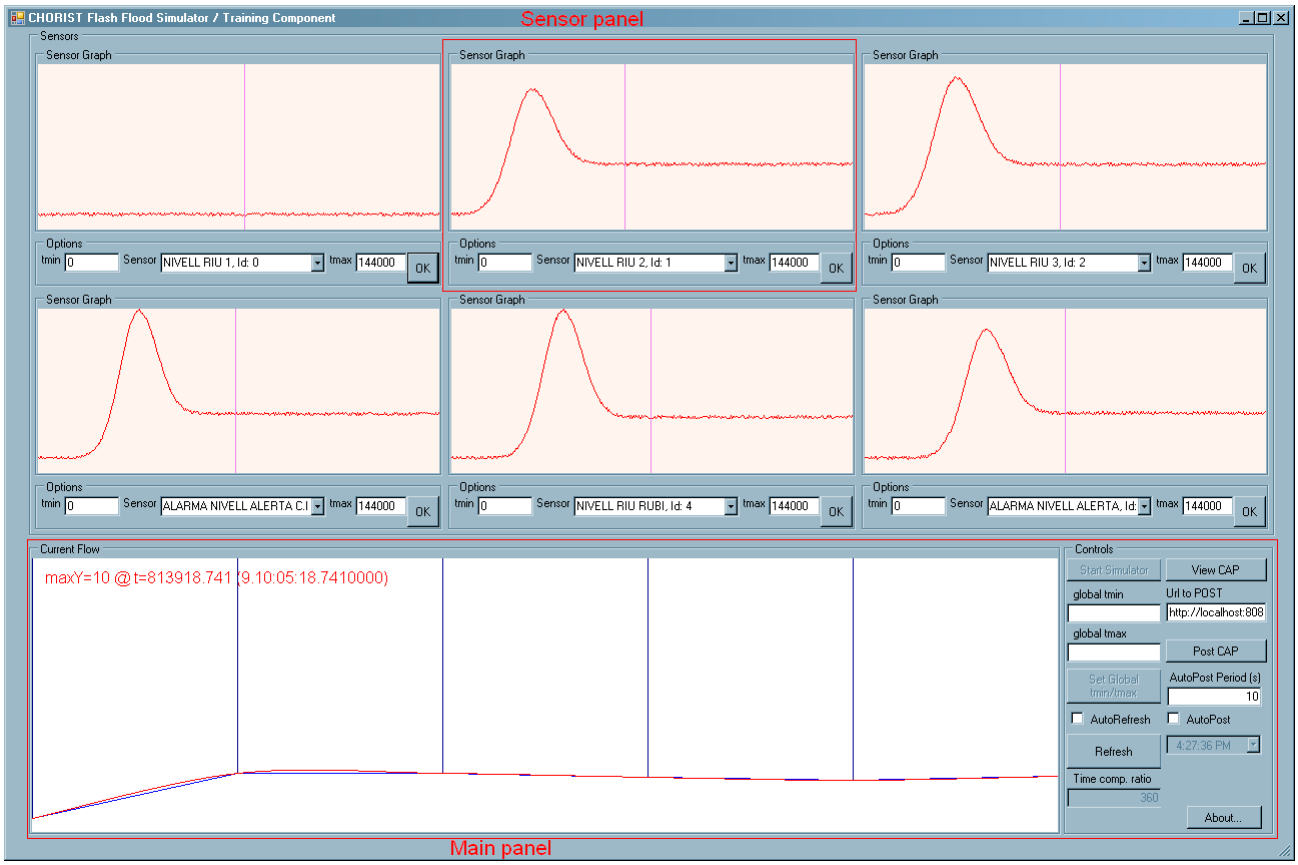


Figure 8 Flash Flood Simulator GUI

6.1 PROGRAM STRUCTURE

The program is divided in two subsystems: the GUI and the simulator.

The subsystems work independently and exchange information: the GUI starts the simulator, and provides it with the configuration as read from the file, while the simulator notifies changes in data to the GUI to visualize the situation updates. This is implemented by using two different threads for the GUI and the simulator.

Since the program can be started providing different configuration files, several instances of the program can be started to simulate floods in different basins and at different times.

6.1.1 Simulator components

The components of the simulation library are:

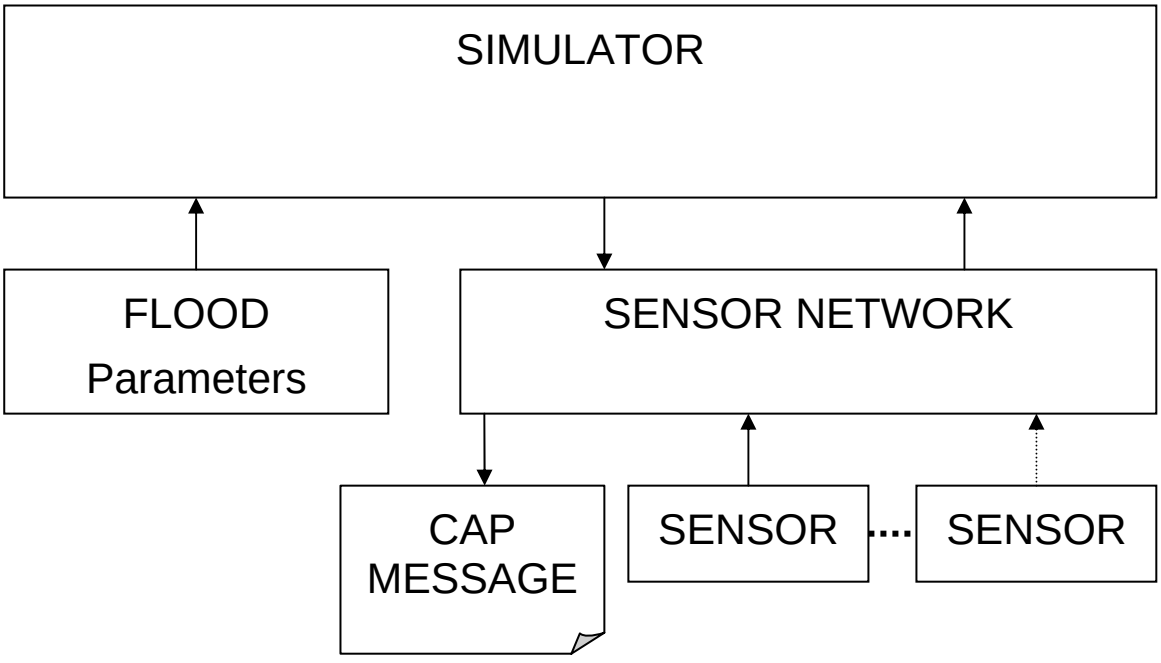
- Sensor
- SensorNetwork
- Flood
- Simulator

The `Sensor` class holds the data about the sensor as read from configuration file, and those computed by the simulation

The `SensorNetwork` is a collection of `Sensors`, which is used by the `Simulator` to organize the `Sensors` and is able to produce the CAP message, which contains the computed information the sensors provide.

The `Flood` class contains all parameter used by the flood theoretic model implemented in the `Simulator`.

The `Simulator` class is configured using `Flood` and `SensorNetwork`; then, when started it will continuously compute a value for all sensors every half second.



6.1.2 GUI components

The GUI allows having a clear image of the ongoing simulation. It contains one sensor panel for each sensor configured, and a graphical presentation of the water level along the river basin.

When the application starts, it reads the configuration of the simulators and creates a sensor panel for each sensor, arranging them in the proper way. The sensor will be shown in reading order, with the first (in the configuration file, which should reflect the position along the basin) in upper left corner and the last in lower right corner.

Each sensor panel can be configured to show each of the sensor signals and different parts of the whole timeline.

When the simulation is started, the GUI starts the simulator. After a given amount of seconds, the GUI reads the simulator status and posts the CAP message generated by the `SensorNetwork`. This action can be requested by the user by means of a button.

If asked by the user, the sensor panels are automatically refreshed, otherwise the OK button needs to be pressed.

6.2 CONFIGURATION FILE

By means of an XML file, it is possible to configure every sensor series. A brief overview of an example can explain how.

```
<?xml version="1.0" encoding="utf-8" ?>

<SensorConfiguration speed="360" start="3600" first="NIVELL RIU 2">
      <Sensor name="NIVELL RIU 1" lat="46.11399" long="-4.05007" />
      <Sensor name="NIVELL RIU 2" lat="45.962449" long="-4.09795" />
      <Sensor name="NIVELL RIU 3" lat="45.915142" long="-4.12226" />
      <Sensor name="ALARMA NIVELL ALERTA C.INFANTA" lat="45.875538" long="-
4.16441" />
      <Sensor name="NIVELL RIU RUBI" lat="45.892762" long="-4.161671" />
      <Sensor name="ALARMA NIVELL ALERTA" lat="45.820882" long="-4.181339" />
</SensorConfiguration>
```

In `SensorConfiguration` node attributes are set about the behaviour of the simulation in terms of:

- Speed: the rate the simulation time is divided by in order to speed up the simulation progress
- Start time: the moment in simulation time the flooding will start
- First sensor: the label of the first sensor that will record a raise in level

Then the configuration will specify in `Sensor` nodes the parameters for each sensor in terms of:

- Name: a label that clearly identifies the sensor
- Lat: the latitude in decimal degrees
- Long: the longitude in decimal degrees

There is no limitation on the number of sensors - the GUI arranges as many panels as needed.