FP7 Grant Agreement N° 312450

CIPRNet

Critical Infrastructure Preparedness and Resilience Research Network

Project type:	Network of Excellence (NoE)	
Thematic Priority:	FP7 Cooperation, Theme 10: Security	
Start date of project:	March 1, 2013	Duration: 48 months

D6.5 Documentation and user manual of the CIP MS&A based 'what if' analysis demonstrator

Due date of deliverable: 31/03/2016
Actual submission date: 31/05/2016

Revision: Version 1

Fraunhofer Institute for Intelligent Analysis and Information Systems (Fraunhofer)

Project co-funded by the European Commission within the Seventh Framework Programme (2007–2013)		
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)	

Author(s)	Jingquan Xie, Betim Sojeva, Stefan Rilling, Thomas Doll, Norman Voß, Erich Rome (Fraunhofer)
Contributor(s)	Marianthi Theocharidou (JRC), Nikolas Flourentzou (UCY)

Security Assessment	Michal Choras (UTP)
Approval Date	24/05/2016
Remarks	No issues found

The project CIPRNet has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 312450.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF METHODS

LIST OF TABLES

ABBREVIATIONS & DEFINITIONS

Abbreviations & Definitions	Explanation
Base layer	Reference layer (background map)
BBK	Bundesamt für Bevölkerungsschutz und Katastrophenhilfe (engl. Federal Office for Civil Protection and Disaster Assistance)
CA(M)	Consequence analysis (module)
CEP	Complex Event Processor
CGI	Common Gateway Interface
CI	Critical infrastructure
CMA	Crisis management action
COM	Component object model
CRA	CI-related action
Docker	Open-source tool to isolate applications and transfer them into containers making these independent from the distribution systems
Fast-CGI	Fast Common Gateway Interface
FRA	First responder action
GIS	Geographical Information System
GNU	Gnu's Not Unix
GUI	Graphical User interface
KNN	K-nearest neighbours
Landing Page	Single web page that shows upon launching a hyperlink from a different domain (e.g. Google search results)
MS&A	Modelling, simulation & analysis
NAT	Network address translation
Scenario	A composition of CI models that feed the federated simulators, a storyline, which is spatially and temporally limited, and a set of options that contain operational measures or actions.
SDL	Scenario Description Language
SyMo	System Modeller
Trainer Dashboard	Tool for administration of training sessions
UI	User Interface
WFS	Web Feature Service
WIA	What-if analysis
WMS	Web Map Service

1 Introduction

1.1 Purpose of this document

This deliverable D6.5 documents the CIP MS&A based 'what if' analysis demonstrator called 'CIPRTrainer' by describing its main features, system requirements, maintenance information, and installation instructions. The user manual part of D6.5 contains detailed instructions of how to use the training engine (see Figure 1) of the demonstrator, including usage examples and screen shots.

1.2 Relation to other CIPRNet deliverables

D6.5 is the last deliverable in WP6 that describes the single-trainee and stand-alone version of CIPRTrainer, the technical demonstrator for the 'what if' analysis capability. It builds on the previous four deliverables in WP6 in the following way. The development of CIPRTrainer followed the four-layered MCRI approach (mission, concepts, realisation, and implementation), introduced in deliverable D6.1 "Conceptual design of a federated and distributed cross-sector and threat simulator" [CIPRNetD61]. D6.1 covered the top two layers, mission and concepts, and parts of the realisation layer. The deliverable D6.4 "Implementation of the integrated CIP MS&A based 'what if' analysis" [CIPRNetD64] covers the remainder of the realisation layer and the implementation layer, including a description of the design engine of CIPRTrainer. D6.5 focuses on using the training engine (see Figure 1) of CIPRTrainer.

In order to test and to use CIPRTrainer, it was also a prerequisite to develop application scenarios, that is, complex cross-border crisis scenarios that constitute the setting for the training of crisis managers. Those scenarios have been described in D6.2 "Application scenario" [CIPRNetD62]. The nature of the scenarios determines to some extent also what technical subsystems are needed, like Critical Infrastructure simulators and threat simulations. In order to make the scenarios usable inside CIPRTrainer, one more step was necessary, namely the creation of models of these scenarios. The deliverable D6.3 "Scenario models" [CIPRNetD63] describes how these models have been created, on what data they were based, and how we coped with missing data. D6.5 contains references to all those deliverables.

In WP6, we are currently creating a multi-trainee version of CIPRTrainer, such that a small crisis management team could use the system for a joint training. The usage of this version of CIPRTrainer will be described in the forthcoming CIPRNet training textbook, deliverable D8.3 [CIPRNetD83]. Also, an interface to the decision support system CIPcast of WP7 is in the testing phase and has been documented as part of deliverable D7.5+D7.6 [CIPRNetD756].

1.3 Structure of the document

The remainder of this document is structured as follows: In the main part of the document, In part A (User manual), the user learns how to set up, start and perform a training session with CIPRTrainer, how to use 'what if' analysis, and how to modify the system settings. In part B (documentation), we provide a "Quick Start" guide for users of CIPRTrainer and an overview of the structure of CIPRTrainer's modular user interface. We conclude by summarising the main properties of the single-trainee version of CIPRTrainer in the following figure.

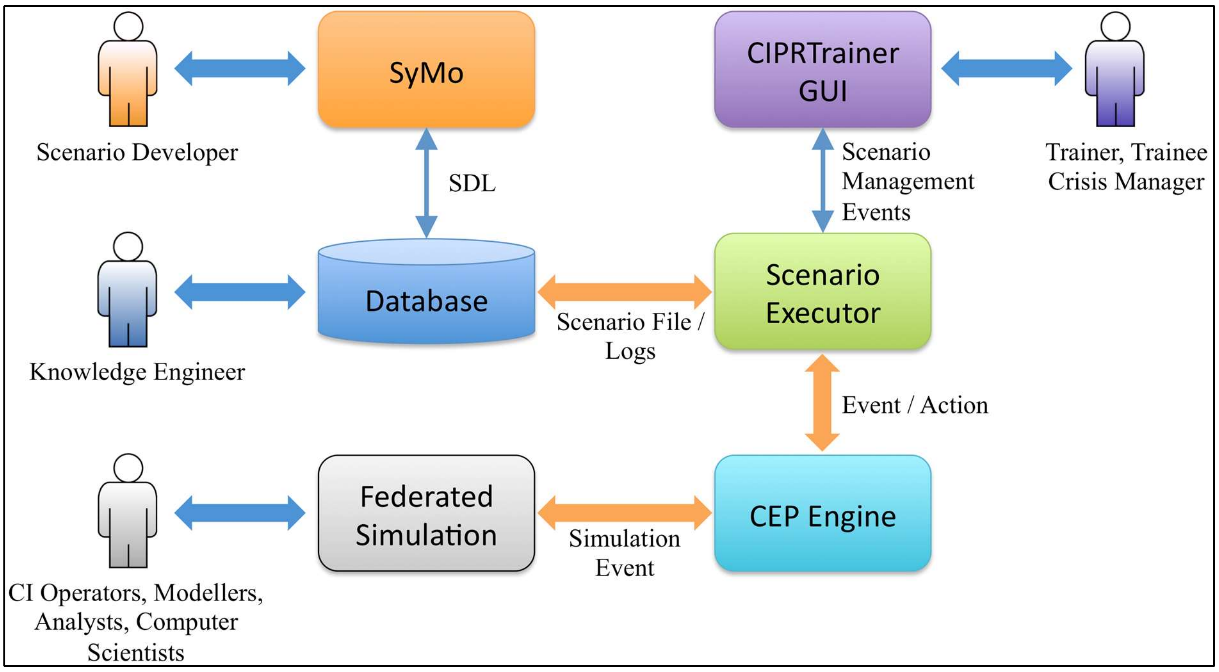


Figure 1: Simplified scheme of CIPRTrainer's main components. The design engine is represented by the rounded rectangle labelled "SyMo" and the scenario developer's icon. All other shown components belong to CIPRTrainer's training engine. The most important element for users of CIPRTrainer is the CIPR-Trainer Graphical User Interface (GUI), through which both trainer and trainee operate CIPRTrainer. As written in deliverables D6.3 [CIPRNetD63] and D6.4 [CIPRNetD64], two of the modelling activities for setting up the scenarios take place in the Federated Simulation part (Critical Infrastructure models) and in the Database part (definition of events, rules, and dependencies).

PART A: USER MANUAL

2 CIPRTrainer's user interface components

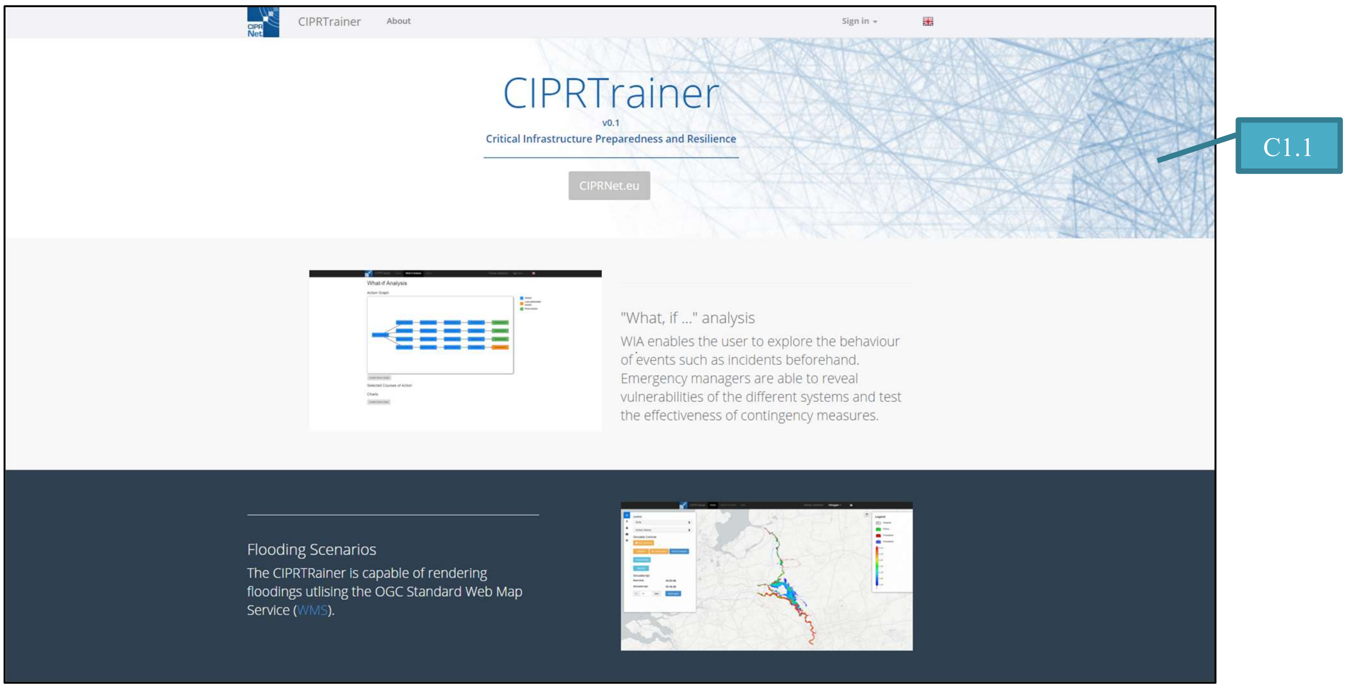
This section describes the user interface (UI) components of CIPRTrainer including main module, trainee module, and trainer module. All components are referenced by indexes (**CX.Y**) and can be found in the UI index list (section 6).

2.1 Main module

Main module is accessible for both user roles trainee and trainer. It consists of a landing page describing implemented features and a navigation bar for authentication and language customisation.

2.1.1 Landing page

The landing page of CIPRTrainer includes main description of the features (**C1.1),** CIPR-Trainer version number (currently **v1.0**), 'about' view and a navigation bar.



"What, if ..." analysis
WIA enables the user to explore the behaviour of events such as incidents beforehand. Emergency managers are able to reveal vulnerabilities of the different systems and test the effectiveness of contingency measures.

Flooding Scenarios
The CIPRTRainer is capable of rendering floodings utlising the OGC Standard Web Map Service (WMS).

Figure 2: CIPRTrainer landing page consists of a front page including major features and navigation bar.

2.1.2 Navigation bar

The navigation bar incorporates

- a hyperlink to the 'about' view (**C1.2**)
- a button for user authentication (**C1.3**)
- button to change the language (**C1.4**)

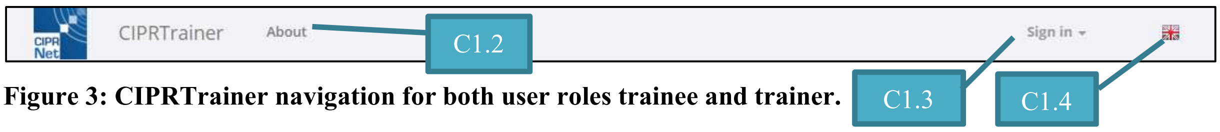


Figure 3: CIPRTrainer navigation for both user roles trainee and trainer.

2.1.3 Authentication panel

The authentication panel includes

- field for username (**C1.5**)
- field for password (**C1.6**)
- a button to sign in (**C1.7**)
- a toggle button to save the created session cookie and keep the user logged-in (**C1.8**)

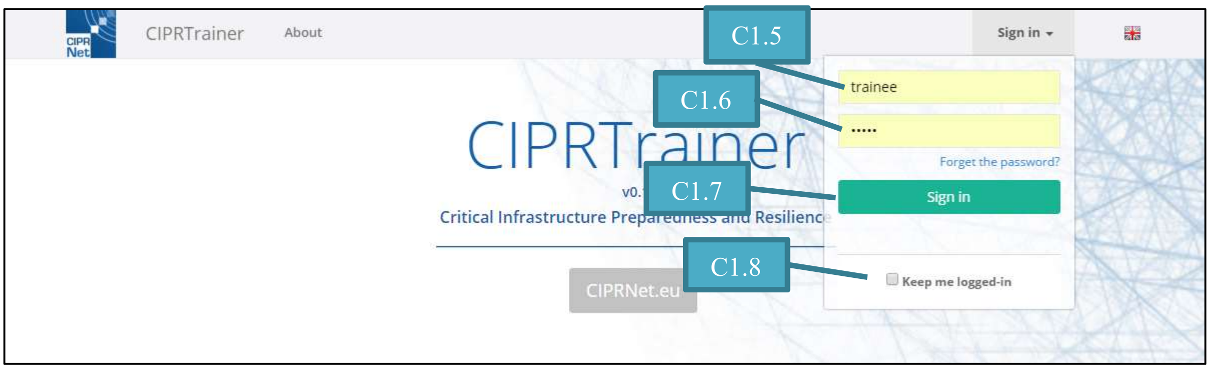
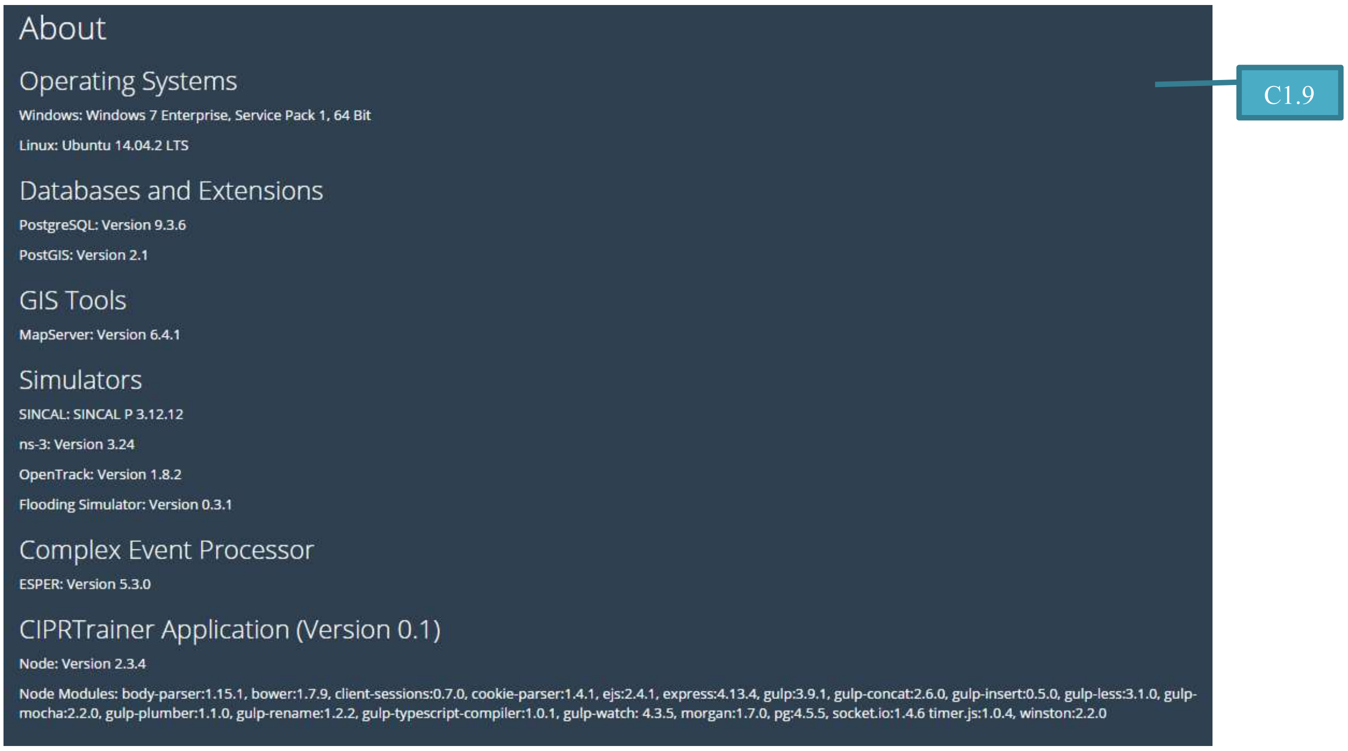


Figure 4: Sign in panel view provides input fields for user authentication.

2.1.4 About view

The 'about view' contains information about tools being used in the development process including the exact version numbers (**C1.9**). Moreover, it contains detailed information about the CIPRTrainer.



About

Operating Systems

Windows: Windows 7 Enterprise, Service Pack 1, 64 Bit

Linux: Ubuntu 14.04.2 LTS

Databases and Extensions

PostgreSQL: Version 9.3.6

PostGIS: Version 2.1

GIS Tools

MapServer: Version 6.4.1

Simulators

SINCAL: SINCAL P 3.12.12

ns-3: Version 3.24

OpenTrack: Version 1.8.2

Flooding Simulator: Version 0.3.1

Complex Event Processor

ESPER: Version 5.3.0

CIPRTrainer Application (Version 0.1)

Node: Version 2.3.4

Node Modules: body-parser:1.15.1, bower:1.7.9, client-sessions:0.7.0, cookie-parser:1.4.1, ejs:2.4.1, express:4.13.4, gulp:3.9.1, gulp-concat:2.6.0, gulp-insert:0.5.0, gulp-less:3.1.0, gulp-mocha:2.2.0, gulp-plumber:1.1.0, gulp-rename:1.2.2, gulp-typescript-compiler:1.0.1, gulp-watch: 4.3.5, morgan:1.7.0, pg:4.5.5, socket.io:1.4.6 timer.js:1.0.4, winston:2.2.0

Figure 5. CIPRTrainer about view includes tools being used in the development phase.

2.2 Trainee module

This section gives a detailed description of the UI components for the trainee.

2.2.1 CIPRTrainer UI trainee home

The overview for trainee includes a control panel, user-specified navigation bar, layer panel, a timeline panel and a GIS map.

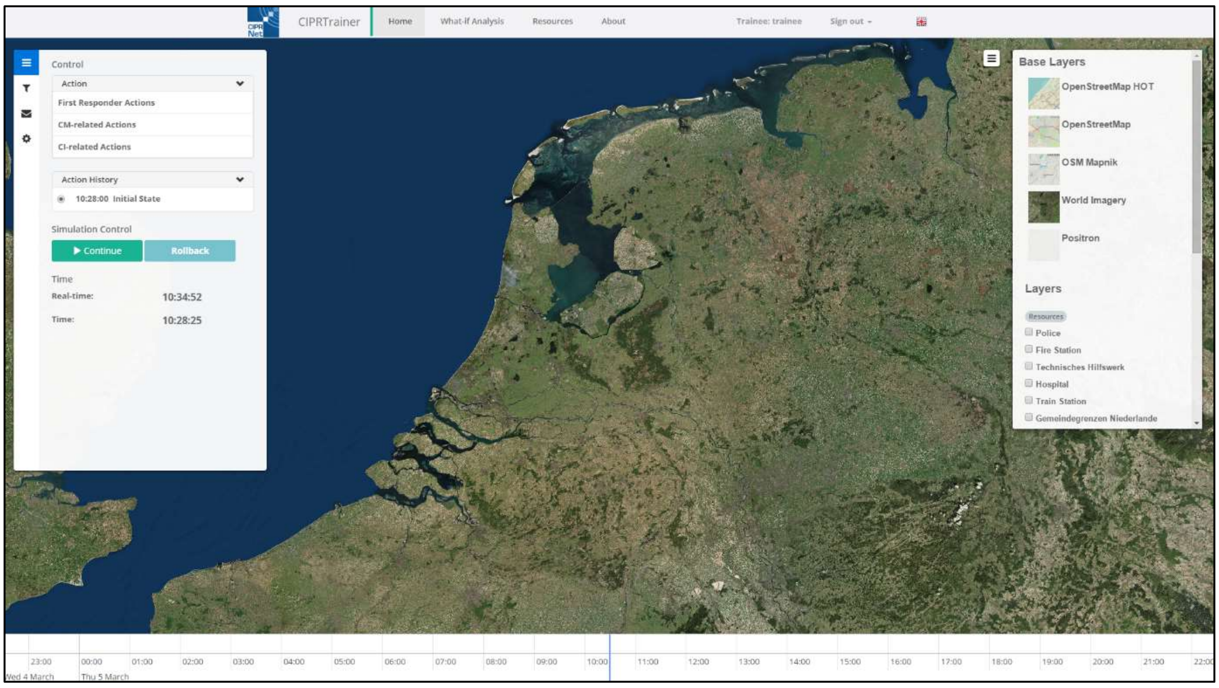


Figure 6: Main view of CIPRTrainer UI for trainee consists of a navigation bar, two sidebars (left and right), a timeline (bottom) and a standard GIS map (centre).

2.2.2 Trainee navigation bar

Trainee navigation bar (Figure 7) encompasses additionally hyperlinks to the main view (**C2.1**) and 'what-if analysis view' (**C2.2**), a label of the user role and username (**C2.3**), and a button for signing out (**C2.4**).

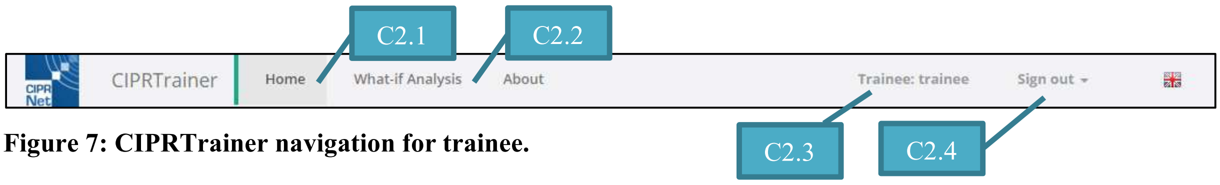


Figure 7: CIPRTrainer navigation for trainee.

2.2.3 GIS map

The GIS map (**C2.5**) of CIPRTrainer depicts various kinds of events (**C2.6**) and CI elements as well as resources containing specific crisis management tactical symbols. In addition, GIS elements are clickable for providing access to detailed information about events, CIs and resources (**C2.7**).

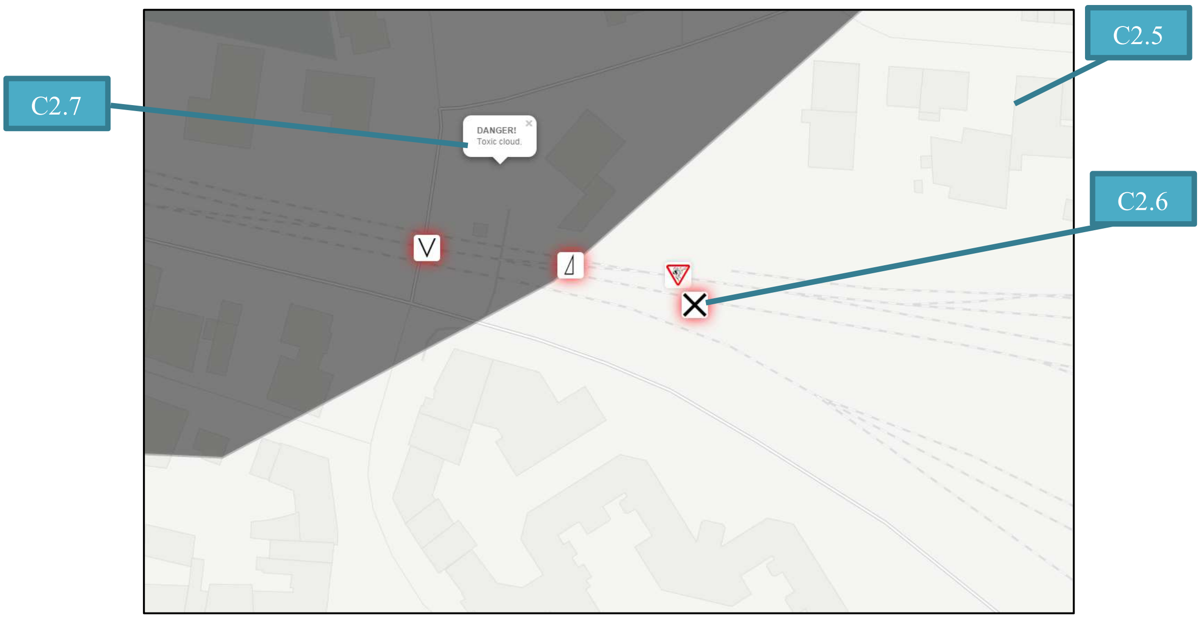


Figure 8: CIPRTrainer GIS map is capable of accumulating various sets of overlays containing information about entities (CIs, resources, socio economic data) as well as critical events attached with a tactical symbols for crisis managers.

2.2.4 Control panel

The control panel (Figure 9) incorporates hyperlinks (left side) to

- control panel itself (**C2.8**)
- filter panel (**C2.9**)
- notification panel (**C2.10**)
- settings panel (**C2.11**)

The main page (right side) provides

- a list of actions (**C2.12**) also referred to as the Action List
- a list of preformed actions (**C2.13**) also referred to as the Action History
- a control-button to continue or pause the simulation (**C2.14**)
- a button to perform a rollback (**C2.15**)
- real time information (**C2.16**)
- simulation time information (**C2.17**)

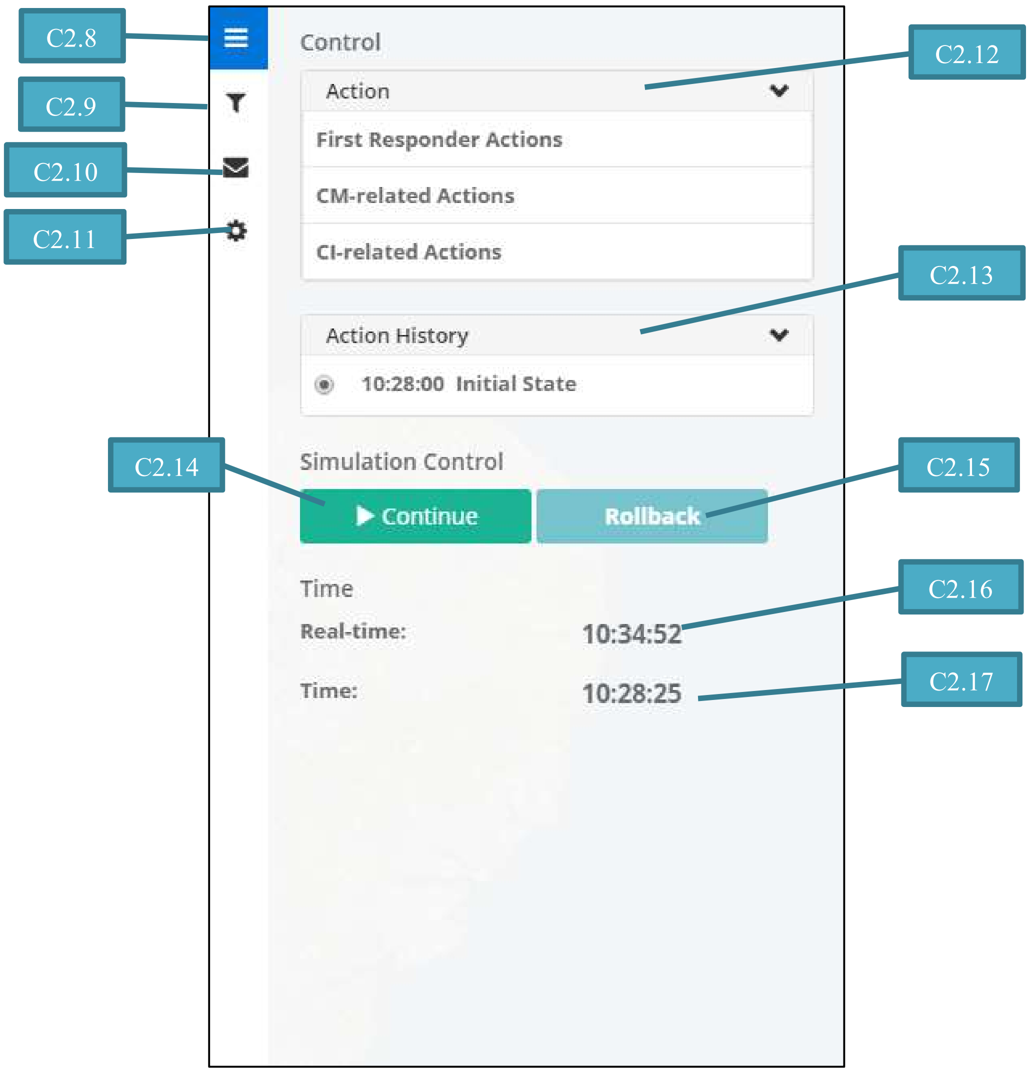
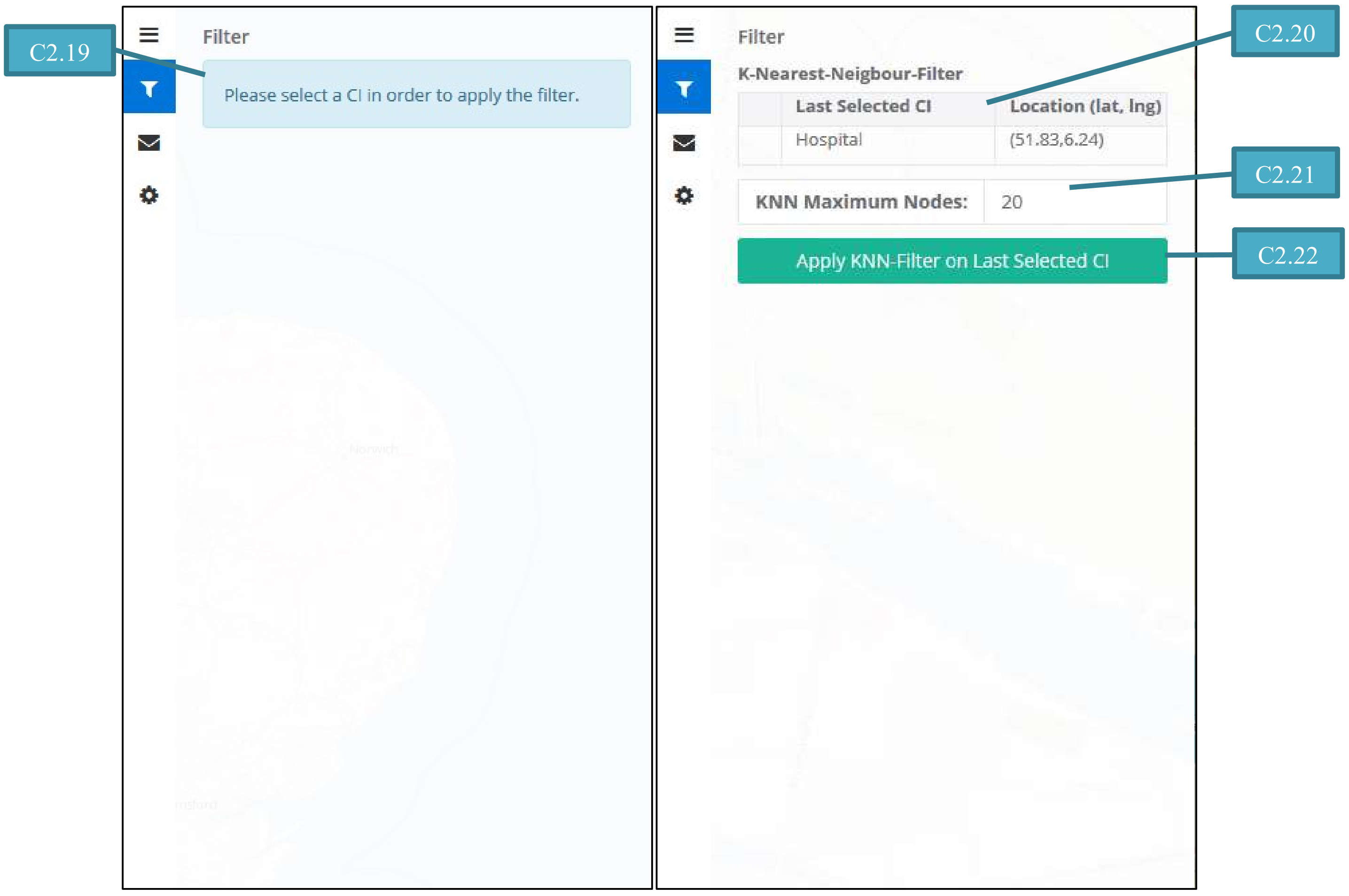


Figure 9: CIPRTrainer control panel.

2.2.5 Filter panel

Filter panel (Figure 10) encompasses implemented filters (currently K-nearest neighbours in CIPRTrainer v1.0) to apply on CI elements on the GIS map. Whenever there is no CI selected on the map, a notification field shows up giving instructions on how to use the filter (**C2.19**). The last selected element is depicted on a table (**C2.20**) containing information about CI and its location. KNN filters requires a parameter (*K*, **C2.21**). A button **(C2.22**) is provided to apply the filter.


C2.19
Filter
C2.20
Filter
K-Nearest-Neigbour-Filter
C2.21
C2.22
Apply KNN-Filter on Last Selected CI

Please select a CI in order to apply the filter.

	Last Selected CI	Location (lat, lng)
	Hospital	(51.83,6.24)

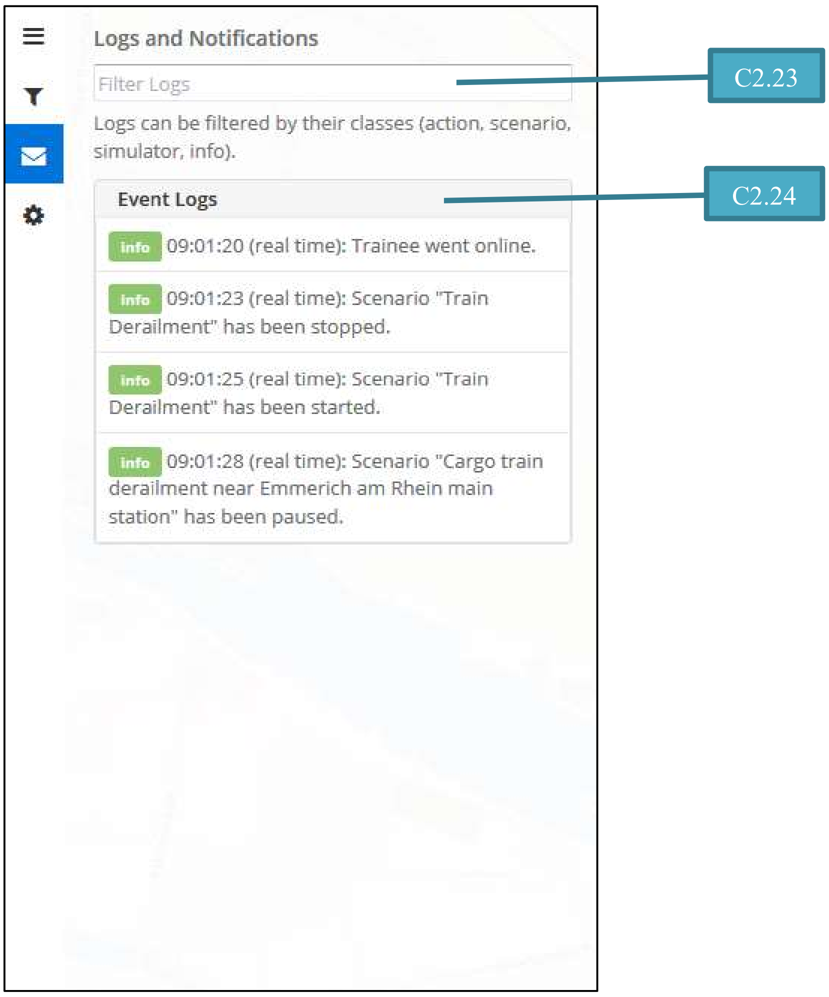
KNN Maximum Nodes: 20

Figure 10: CIPRTrainer filter panel.

2.2.6 Notification panel

Notification panels provides

- filter for logs (**C2.23**)
- list of event logs (**C2.24**)



Logs and Notifications

Filter Logs

Logs can be filtered by their classes (action, scenario, simulator, info).

Event Logs

info 09:01:20 (real time): Trainee went online.

info 09:01:23 (real time): Scenario "Train Derailment" has been stopped.

info 09:01:25 (real time): Scenario "Train Derailment" has been started.

info 09:01:28 (real time): Scenario "Cargo train derailment near Emmerich am Rhein main station" has been paused.

Figure 11: CIPRTrainer notification panel.

2.2.7 Settings panel

Settings panel provides

- toggle buttons to hide or show layers, resources, legend and timeline (**C2.25**)
- unchangeable input field of the username (**C2.26**)
- input fields for first name, last name and email address (**C2.27**)
- input fields to change password (**C2.28**)
- "Save" and "Cancel" button to confirm and discard changes, respectively (**C2.29**)

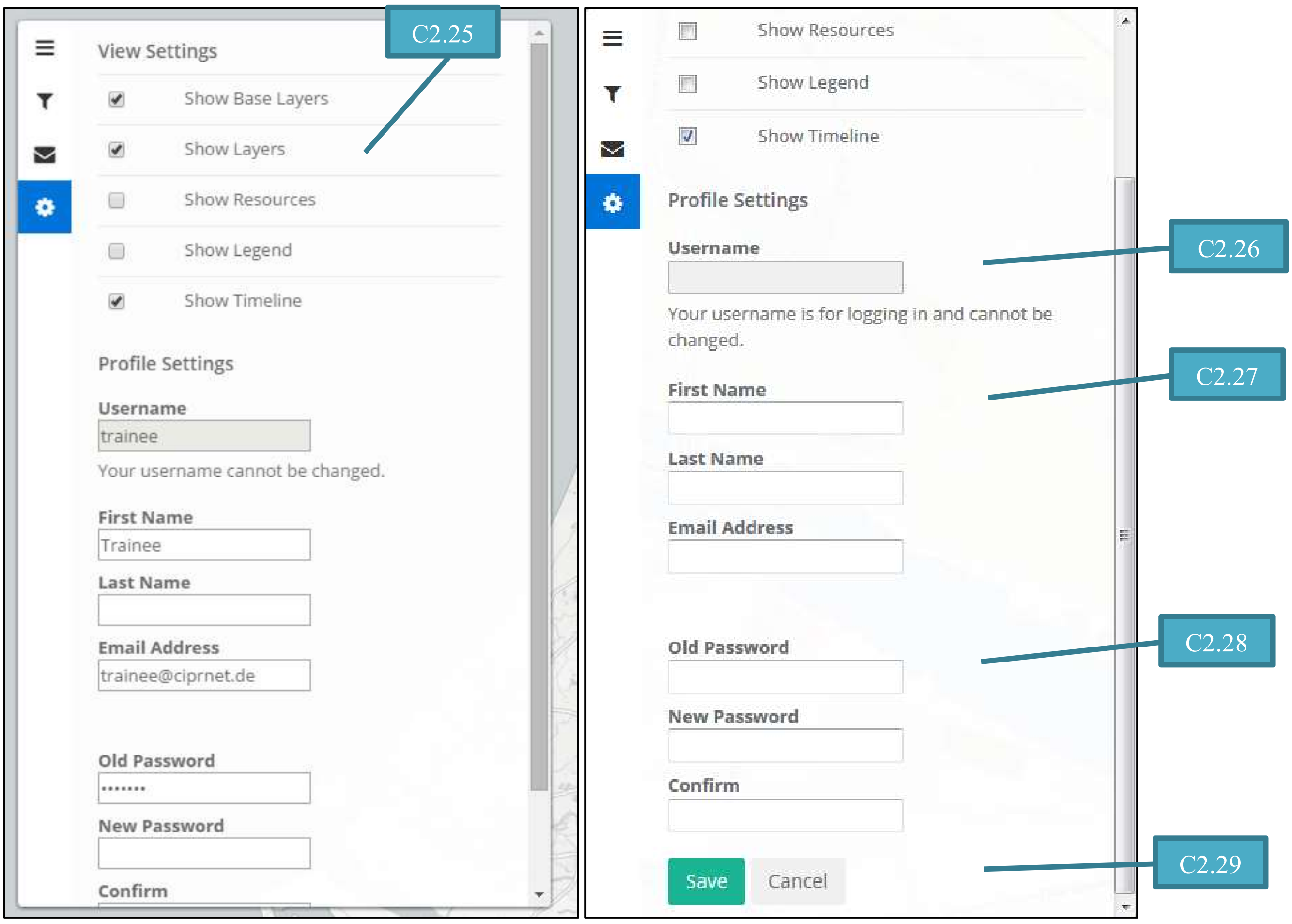


Figure 12: CIPRTrainer settings panel.

2.2.8 Layer panel

Layer panel shows a list of

- various GIS reference maps also referred to as base layers (**C2.30**)
- thematic GIS layers depicting resources (**C2.31**) and CI elements of power networks (**C2.32**), telecommunication networks (**C2.33**) and railway networks

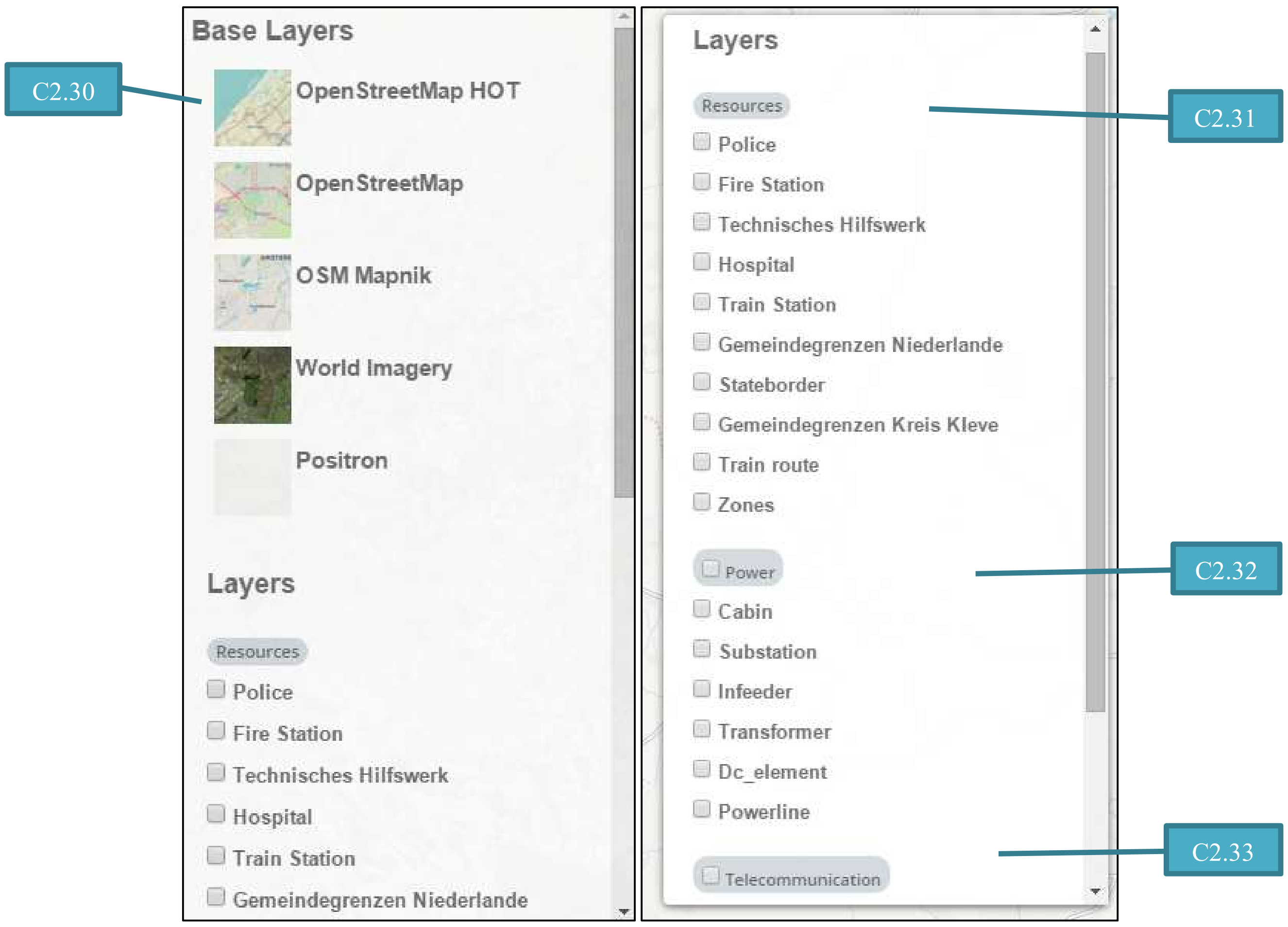


Figure 13: CIPRTrainer layer panel.

2.2.9 Timeline panel

Timeline shows different types of events in a chronical order, of which there are:

- events that are described in the storyline, also visible on the GIS map (marked as red; **C2.34**)
- external events produced by the simulators and not visible on the map (marked as yellow; **C2.35**)

Moreover, the current simulation time is indicated by the blue line moving forward in a specific speed rate (**C2.36**).

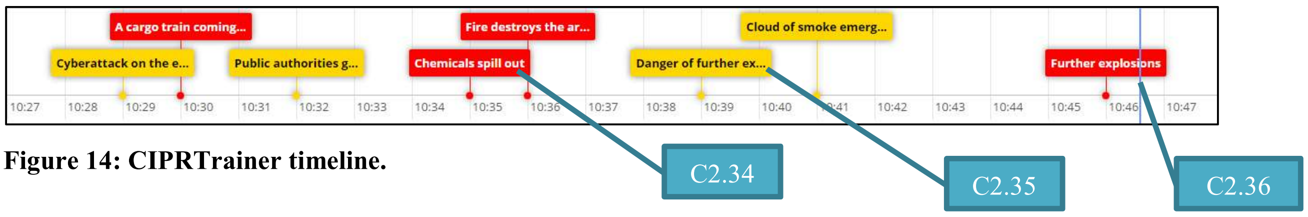
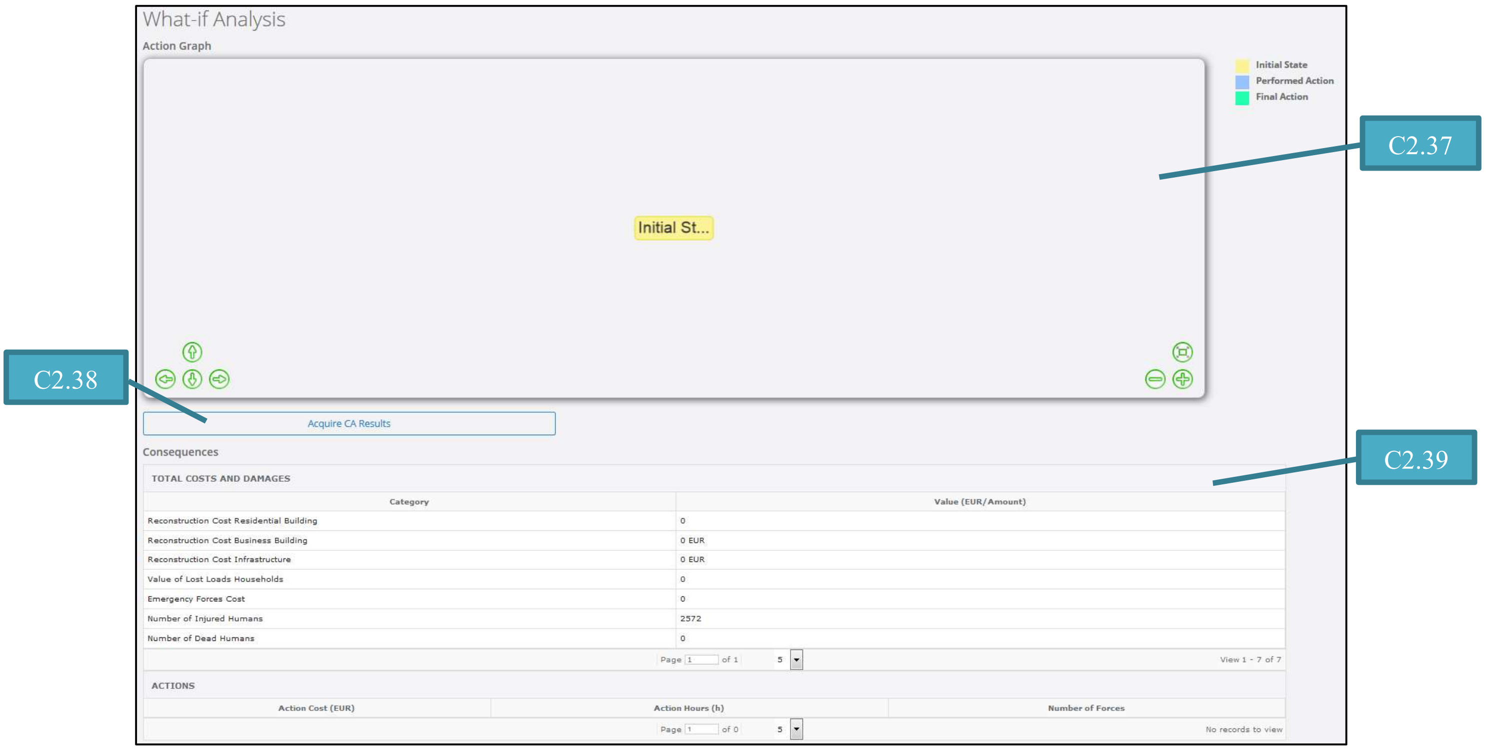


Figure 14: CIPRTrainer timeline.

2.2.10 What-if Analysis view

The 'What-if Analysis' view (WIA view) consists of

- action tree view (**C.2.37**)
- "Acquire CA Results" button for computing the consequences (**C2.38**)
- a list of computed results (**C2.39**)
- a GIS map for showing various kinds of spatial data computed by the Consequence Analyse Module (CAM; **C2.40**)



Consequences

TOTAL COSTS AND DAMAGES

Category	Value (EUR/Amount)
Reconstruction Cost Residential Building	0
Reconstruction Cost Business Building	0 EUR
Reconstruction Cost Infrastructure	0 EUR
Value of Lost Loads Households	0
Emergency Forces Cost	0
Number of Injured Humans	2572
Number of Dead Humans	0

Page 1 of 1 5 View 1 - 7 of 7

ACTIONS

Action Cost (EUR)	Action Hours (h)	Number of Forces

Page 1 of 0 5 No records to view

Figure 15: CIPRTrainer WIA view (upper part) incorporates the action graph and CA result tables.

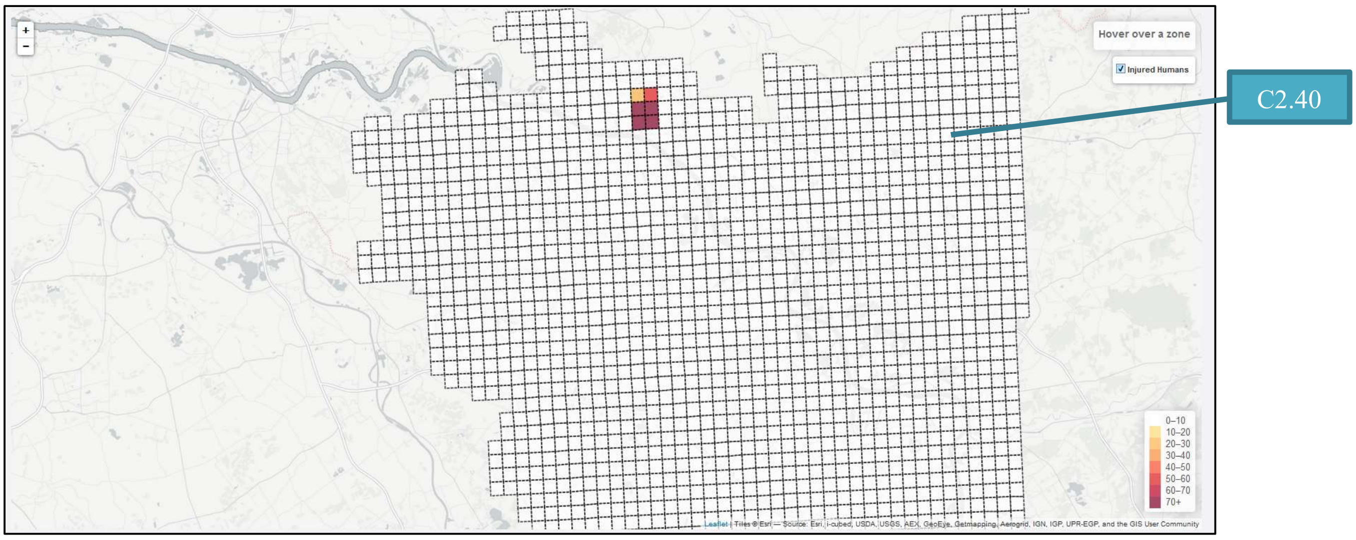

+
-
Hover over a zone
Injured Humans
C2.40
0–10
10–20
20–30
30–40
40–50
50–60
60–70
70+
Leaflet | Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

Figure 16: CIPRTrainer WIA view (lower part) CA result GIS map.

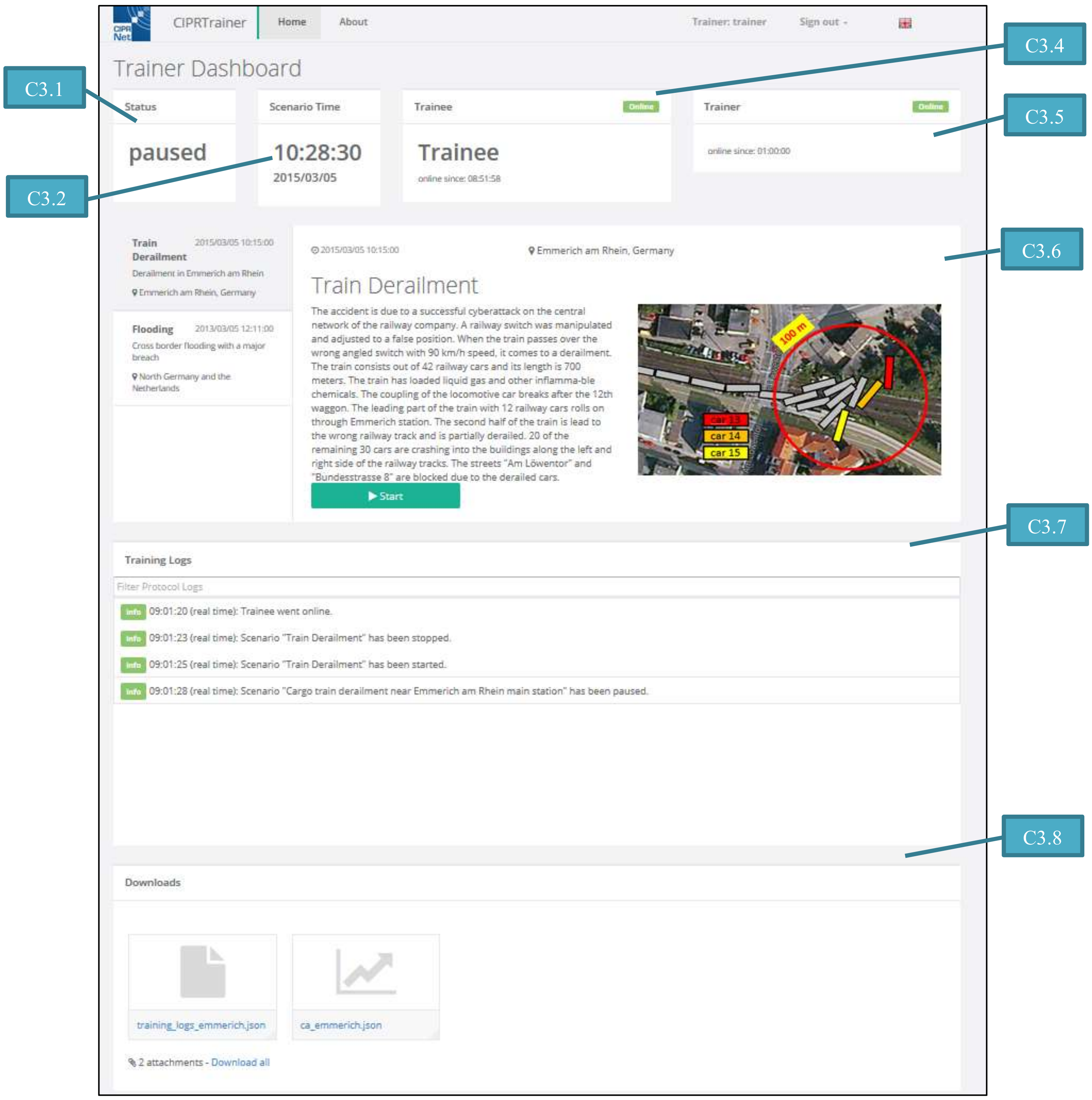
2.3 Trainer module

This section gives a detailed description of the CIPRTrainer UI dashboard components, which is only accessible for the trainer.

2.3.1 CIPRTrainer UI dashboard

The CIPRTrainer UI dashboard consists of

- a status panel depicting the current simulation state (**C3.1**)
- time panel showing simulation time (**C3.2**)
- trainee status including login time, name, and status (online/offline; **C3.4**)
- trainer status (**C3.5**)
- scenario panel including list of possible scenarios to be trained including description text, image, and a button to start and stop training (**C3.6**)
- list of training logs including a filter capability (**C3.7**)
- panel to download computed CA results and logs in JSON-format (**C3.8**)



Trainer Dashboard

Status
paused

Scenario Time
10:28:30
2015/03/05

Trainee
Trainee
online since: 08:51:58

Trainer
online since: 01:00:00

Train Derailment 2015/03/05 10:15:00
Derailment in Emmerich am Rhein
Emmerich am Rhein, Germany

Flooding 2013/03/05 12:11:00
Cross border flooding with a major breach
North Germany and the Netherlands

2015/03/05 10:15:00 Emmerich am Rhein, Germany

Train Derailment

The accident is due to a successful cyberattack on the central network of the railway company. A railway switch was manipulated and adjusted to a false position. When the train passes over the wrong angled switch with 90 km/h speed, it comes to a derailment. The train consists out of 42 railway cars and its length is 700 meters. The train has loaded liquid gas and other inflamma-ble chemicals. The coupling of the locomotive car breaks after the 12th waggon. The leading part of the train with 12 railway cars rolls on through Emmerich station. The second half of the train is lead to the wrong railway track and is partially derailed. 20 of the remaining 30 cars are crashing into the buildings along the left and right side of the railway tracks. The streets "Am Löwentor" and "Bundesstrasse 8" are blocked due to the derailed cars.

Training Logs

Filter Protocol Logs

```
Info 09:01:20 (real time): Trainee went online.
Info 09:01:23 (real time): Scenario "Train Derailment" has been stopped.
Info 09:01:25 (real time): Scenario "Train Derailment" has been started.
Info 09:01:28 (real time): Scenario "Cargo train derailment near Emmerich am Rhein main station" has been paused.
```

Downloads

training_logs_emmerich.json
ca_emmerich.json

2 attachments - Download all

Figure 17: CIPRTrainer trainer dashboard.

3 Using CIPRTrainer's user interface

If all installation procedures are completed successfully (see part B), then the CIPRTrainer should be accessible on `http://host-machine-ip/ciprtrainer`. The first view of CIPRTrainer is the landing page, which contains a navigation bar providing user authentication and a main page describing major features of the tool (Figure 2). This section provides guides for the three modules of the CIPRTrainer: main, trainee and trainer module. The complete index list of user interface (UI) components is located in section 6. It is recommended to take first a look at the CIPRTrainer's UI components in section 2.

3.1 Main module

This section briefly describes how to perform user authentication and language settings.

3.1.1 Authentication

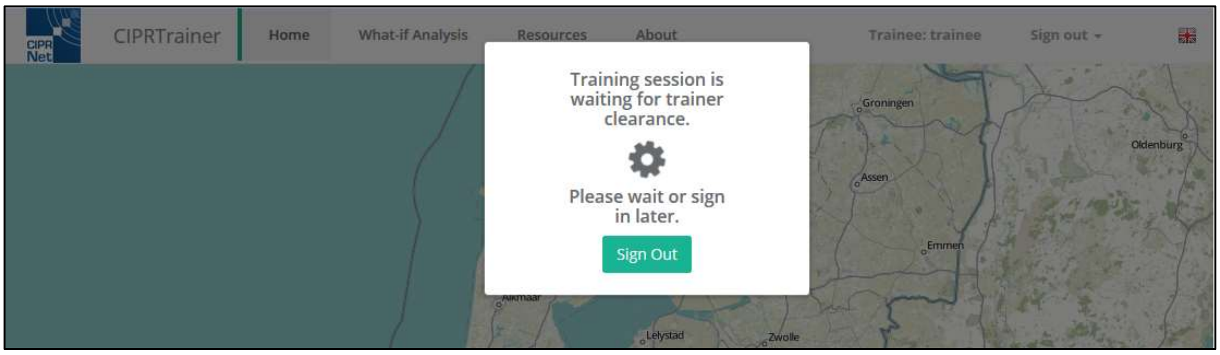
3.1.1.1 Login

To sign in as a user, please follow these steps:

1. Click on the 'Sign In' button (**C1.3**) on the navigation bar
2. Type in your credentials:
    a. username (**C1.5**)
    b. password (**C1.6**)
3. Check or uncheck the toggle button to save or delete the session cookie (**C1.8**)
4. Click the 'Sign In' button to login

Method 1: Signing In.

After signing in as a trainee, if the training session is not ready, a panel will show up pointing out that training session is waiting for trainer clearance. Whenever the trainee starts the simulation, the panel will disappear (Figure 18). While in this state, trainee can only sign out.



Training session is waiting for trainer clearance.

Please wait or sign in later.

Sign Out

Figure 18: Trainee is waiting for trainer's clearance.

3.1.1.2 Logout

To sign out, please follow these steps:

1. Click on the 'Sign Out' button on the navigation bar
2. Confirm the process by clicking on 'Sign Out' button

Method 2: Signing out.

3.1.1.3 Resetting password

Resetting the password requires following steps:

1. Click on the 'Sign In' button (**C1.3**) on the navigation bar
2. Click on the 'Forgot the password?' hyperlink
3. Enter your username
4. Click on 'Reset password' button
5. You will receive an email with a new password
6. To change the newly generated password, see section 3.2.6

Method 3: Resetting the password.

3.1.2 Changing language

Currently, CIPRTrainer provides three different languages: English, German and Dutch (Figure 19). Changing the langue requires following steps:

1. Click on the flag button (**C1.4**) on the navigation bar
2. Click on a desired flag to change the corresponding language

Method 4: Changing the language.

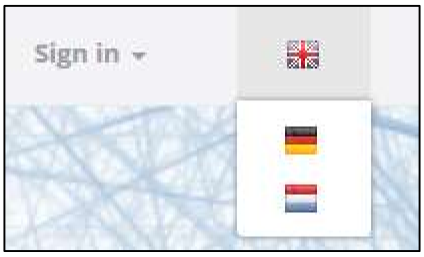


Figure 19: CIPRTrainer provides the languages English, German and Dutch.

3.2 Trainee module

In this section we describe main features of the trainee module and how to use them. Trainees have various options to interact with the system including:

1. Pausing and continuing the scenario
2. Performing various kinds of actions (first responder, crisis management)
3. Performing rollbacks (jumping into a prior state of the scenario)
4. Examining and mapping critical infrastructures
5. Observing and keeping track of the evolution of the scenario

6. Managing user account and customizing CIPRTrainer view components (layer panels, timeline, etc.)
7. Conducting analysis methodologies (CA)

3.2.1 Pausing and continuing the scenario

On the left control panel (**C2.8**) of CIPRTrainer the user can click on the button “Pause/Continue” (**C2.14/C2.15**) to pause or continue the scenario. Other participants like trainees and trainer are notified about this event.

3.2.2 Performing various kinds of actions

CIPRTrainer provides various kinds of actions including first responder, crisis management and CI operator actions (**C2.12**). Currently, three types of actions are offered. These are:

8. First Responder Actions (FRAs)
9. Crisis Management Actions (CMAs)
10. CI-Related Actions (CRAs)

To perform a certain kinds of action, please click on the desired element in the Action List (**C2.12**). A UI for each action type will show up.

3.2.2.1 First Responder Actions

FRAs incorporate resources like police, fire brigades and technical relief services (THW) and can be performed by a trainee. The trainee can order different resources to perform an activity at a predefined location. Forces or resources are spread in the region around the scenario location (in this case Emmerich) as shown in Figure 21. Performing a FRA includes few steps:

1. Click on the element `First Responder Action` on the Action List (**C2.12**) on the control panel
2. Select the desired resource. Possible options are:
    a. Action forces (fire brigades)
    b. Law Forces (police)
    c. Rescue Forces (technical relief organisation)
3. Select the origin (UI: FROM, Zone 1-15)
4. Select the destination (UI: TO, Zone 1-15)
5. Select the Capacity
    a. First Field: Number of leaders
    b. Second field: Number of sub-leaders
    c. Number of forces
6. Select an activity
7. Confirm or cancel the action

Method 5: Performing First Responder Actions.

The capacity within the context of a FRA is defined as a quadruplet and refers to the strength of a specific unit (Figure 20). The user can decide whether to send units with the highest possible capacity to the crisis region or to choose a lower capacity.

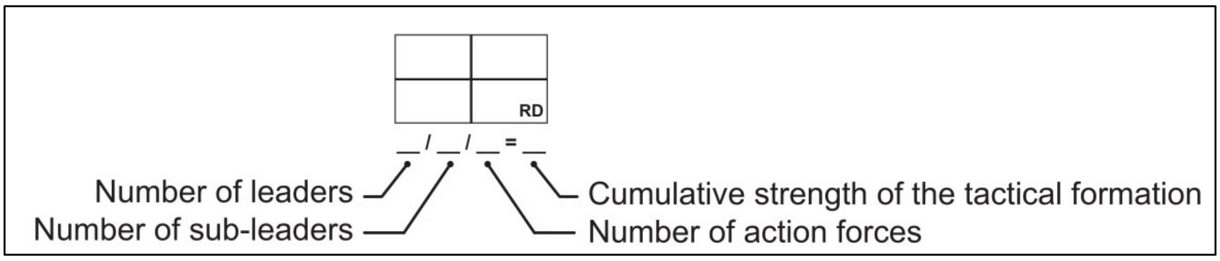
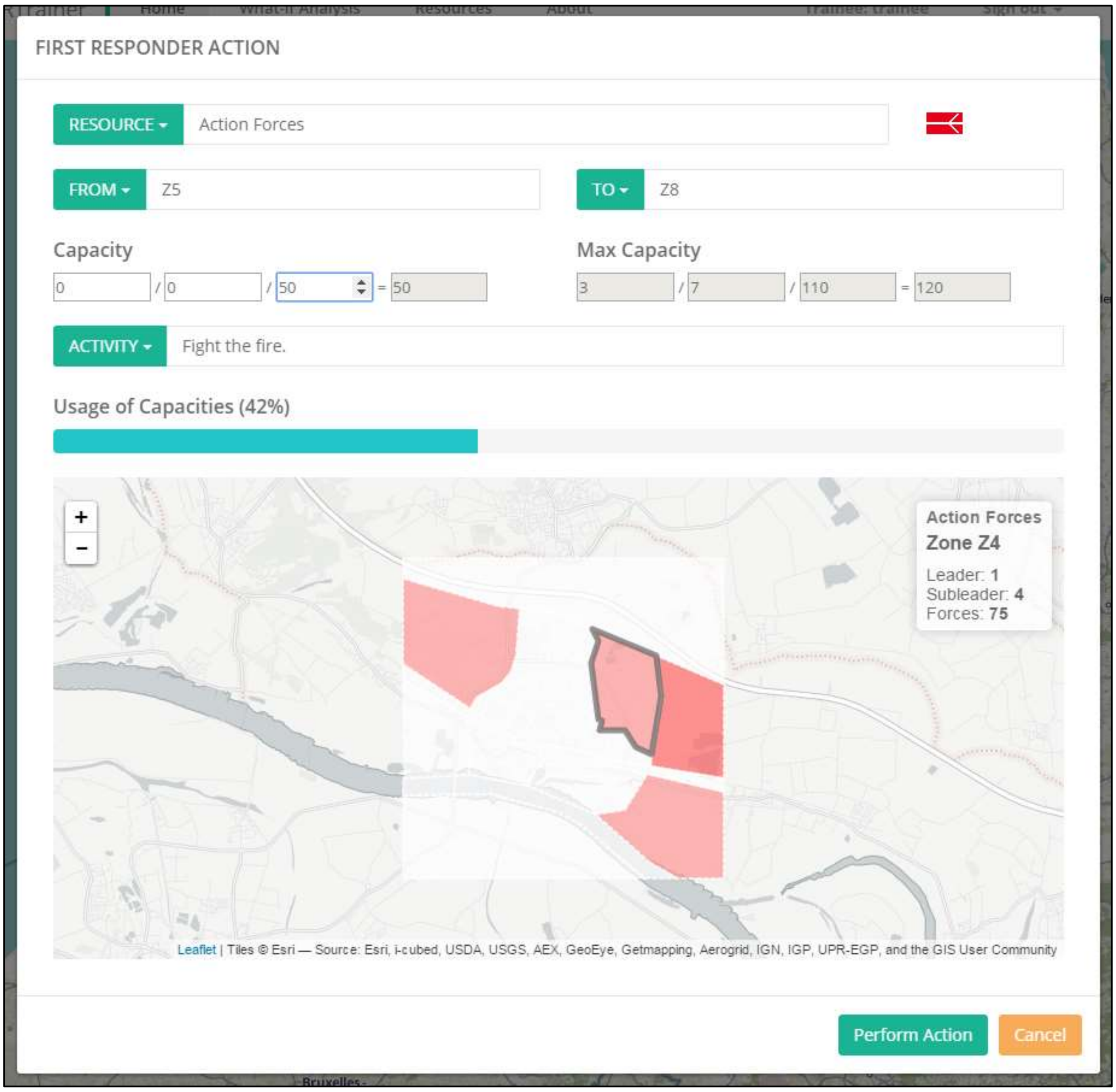


Figure 20: Tactical symbol notation to define strength of units.

When the action is performed successfully, the system pushes a notification that confirms the conduct. Also, the timeline will be updated with a blue label containing information about all action parameters including simulation time.


FIRST RESPONDER ACTION

RESOURCE: Action Forces

FROM: Z5

TO: Z8

Capacity: 0 / 0 / 50 = 50

Max Capacity: 3 / 7 / 110 = 120

ACTIVITY: Fight the fire.

Usage of Capacities (42%)

Action Forces
Zone Z4
Leader: 1
Subleader: 4
Forces: 75

Leaflet | Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, Getmapping, Aerogrid, IGN, IGP, UPR-EGP, and the GIS User Community

Figure 21: Interface for performing First Responder Actions.

3.2.2.2 Crisis Management Actions

CMAs are referred to as the second type of actions that CIPRTrainer provides. This type is dedicated for trainees or crisis managers. A crisis manager is able to alarm public authorities and the general public as well as evacuate critical regions (Figure 22). Each action influences the consequences in the scenario. For instance, when the crisis manager decides to not inform the general public, the number of injured people will rise. On the other side, if the crisis manager performs this action in an early state of the scenario, then less people will be injured, which mitigates the consequences. Each action triggers predefined rules based on the action-function, the time and the underlying socio-economic data. Performing a CMA requires few steps:

1. Click on the category `Crisis Management Action` on the list Action on the control panel
2. Choose a predefined action
3. Confirm or cancel the action

Method 6: Performing Crisis Management Actions.

Like in FRA the timeline will be updated and other participants (trainer) will be noticed about this event.

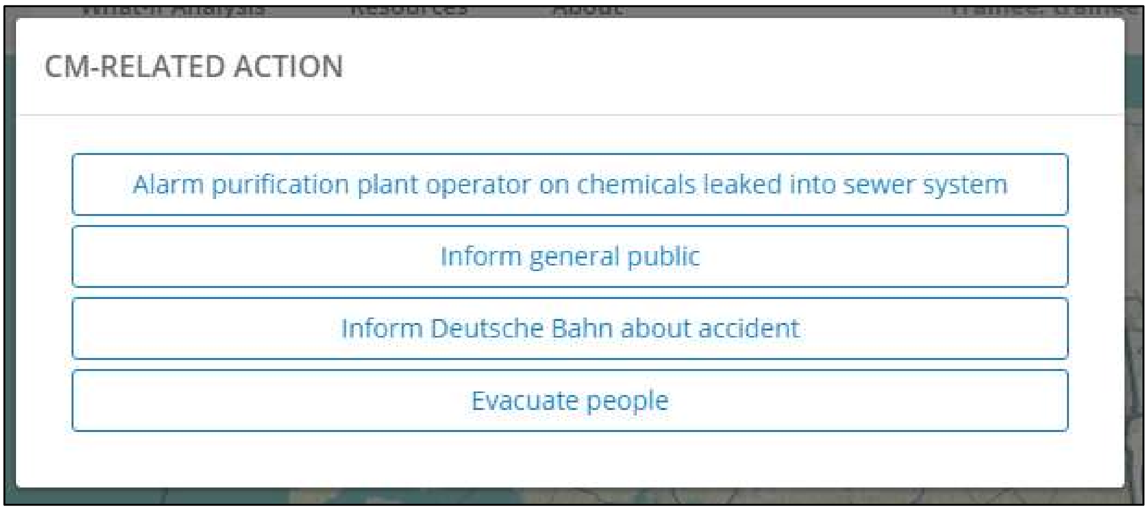

CM-RELATED ACTION
Alarm purification plant operator on chemicals leaked into sewer system
Inform general public
Inform Deutsche Bahn about accident
Evacuate people

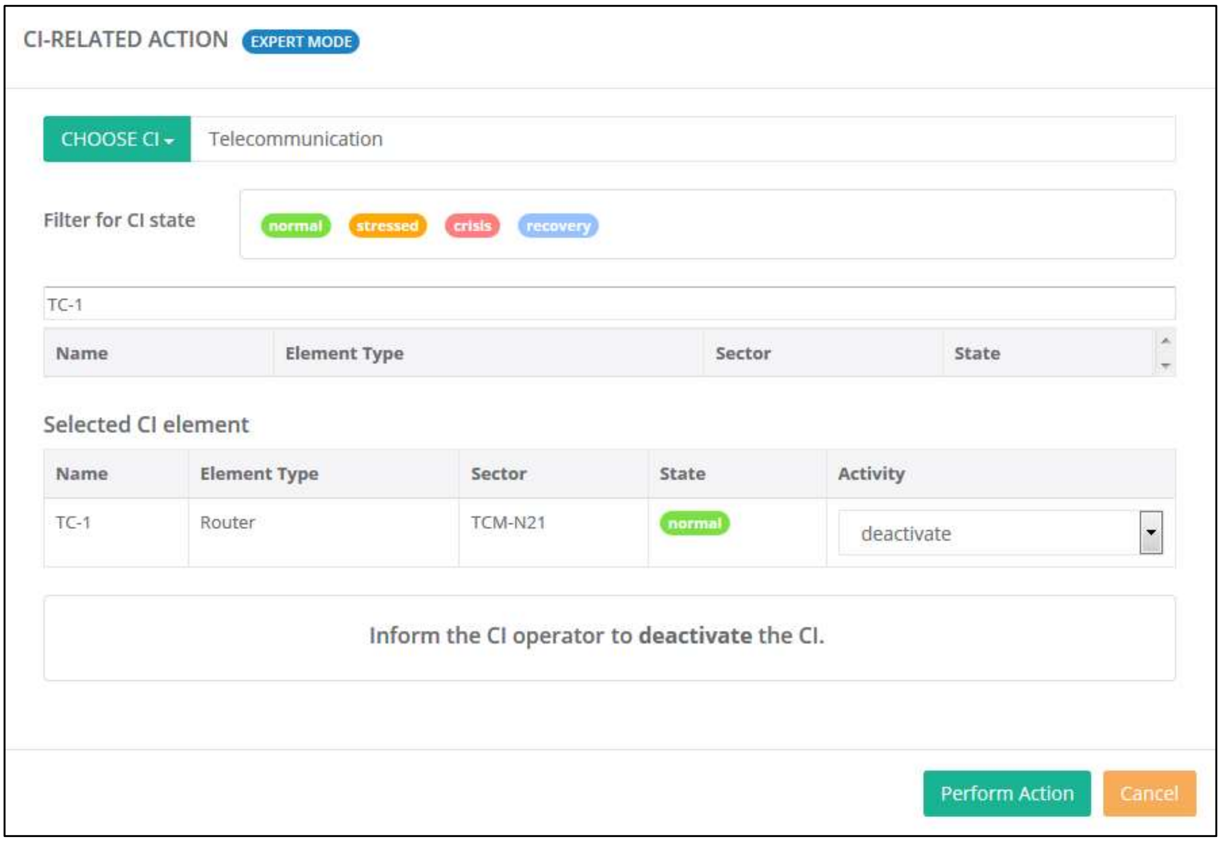
Figure 22: Interface for performing Crisis Management Actions.

3.2.2.3 CI-Related Actions

CRAs are the last kind of actions provided by CIPRTrainer v1.0 and enables the capability to manually activate or deactivate chosen CIs for mitigating concerns (Figure 23). This action is referred to be only accessible for experts to analyse possible consequences on certain states of the scenario. CIs can have four states: normal, stressed, crisis, and recovery. A CI is considered to be in a normal state when there is no impairment or damage. A CI being in the stressed mode means that business or operation serviced are impaired but not shut down complete whereas the crisis mode concludes to an overall shut down of all services including substantial damages. In the recovery mode the CI shuts down all services for mitigating reasons but does not have any damages. Performing CRA requires the following steps:

1. Click on the category `CI Related Action` on the list Action on the control panel (**C2.12**)
2. Choose a CI sector (e.g. telecommunication network, power network)
3. Examine the list of provided CI elements
4. Use the filter capability to query the list of CIs (optional)
5. Choose an element and click on the Activity column to activate or deactivate the chosen CI
6. Confirm or cancel the action

Method 7: Performing CI-Related Actions.


CI-RELATED ACTION EXPERT MODE

CHOOSE CI ▾ Telecommunication

Filter for CI state: normal stressed crisis recovery

TC-1

Name	Element Type	Sector	State

Selected CI element

Name	Element Type	Sector	State	Activity
TC-1	Router	TCM-N21	normal	deactivate

Inform the CI operator to **deactivate** the CI.

Perform Action Cancel

Figure 23: Interface for performing CI-Related Actions.

3.2.3 Performing Rollbacks

To perform a rollback the user needs to decide first on which state the scenario should be re-instated. Therefore CIPRTrainer inserts each performed action in the Action History list (**C2.13**). Performing a rollback requires following steps:

1. Select a desired action that has been performed in the list Action History (**C2.13**)
2. Click on the button “Rollback” (**C2.15**)
3. Confirm or cancel the event

Method 8: Performing Rollback.

3.2.4 Examining and mapping critical infrastructures

CIs can be examined using GIS overlays. The user is able to visualise and hide CI overlays by using the layer panel. Following steps are required to examine CIs:

1. Make sure layer panel is visible
    a. Open the settings panel
    b. Check "Show Layers" toggle (**C2.25**)
2. Click on one more multiple CIs on the layer panel (**C2.31-C2.33**)
3. Examine GIS elements on the map
    a. Click on a desired element to show more information
    b. Click outside of the appeared pop-up window to close it

Method 9: Examining CIs.

To examine an *entire* CI model, for instance power networks, the user can click the list element highlighted with a grey background. By doing so, all components of a power network (e.g. cabins, substations, transformer, etc.) will be checked as well and shown on the map.
CI elements on the map are visualised using icons representing the entity with an underlying LED light on the upper right corner. This light indicates the operational status of the element. Table 1 shows the relation between colour and operational status of a CI element.

Table 1: LED lights on the upper right corner of CI elements indicate the operational statuses.

Colour		CI State
Green	🟩	Normal state; service up
Yellow	🟨	Service partially shut down; no substantial damages
Red	🟥	Service completely shut down due to damages
Blue	🟦	Service completely shut down, but currently no damages

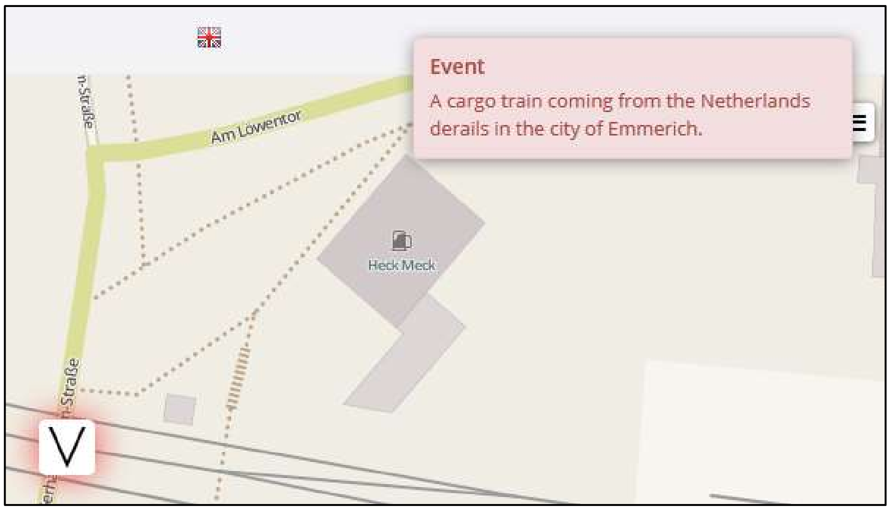
3.2.5 Observing and keeping track of the evolution of the scenario

The user has three options to observe and keep track of the evolution of scenario, namely using:

11. GIS map
12. Timeline
13. Notification Logs

3.2.5.1 Using GIS map

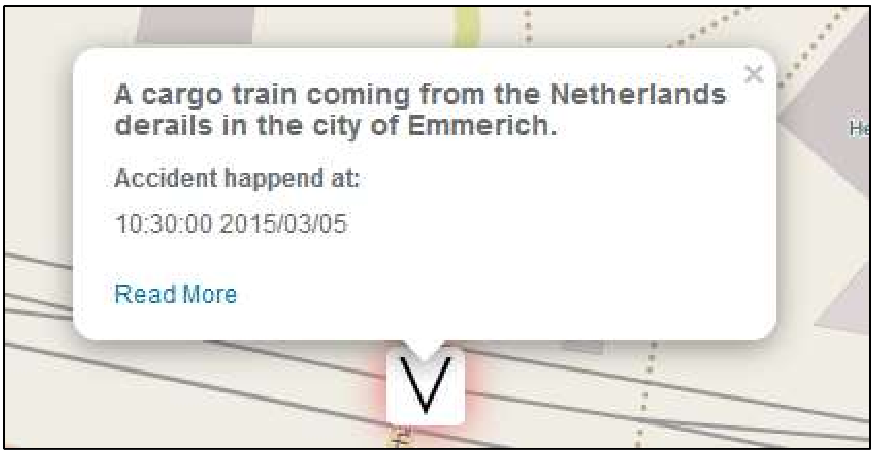
CIPRTrainer GIS map visualises storyline events and status reports as GIS elements using the corresponding tactical symbol on the map. When CIPRTrainer receives an event by the Scenario Executor, the user will get a notification showing up on the upper right corner containing a short description of the event. Simultaneously, the event appears on the map, too (Figure 24). Clicking on the item reveals a pop-window with more detailed information showing a short description and accident time. To gain more information, the user can click on the 'Read More' button. A new window pops up including all information of this event (Figure 26). To close the window pop-up, the user can push key `ESC` or click outside the message box. To hide the all events on the map, the user can toggle the list element 'Critical Events' on the layer panel.



Event

A cargo train coming from the Netherlands derails in the city of Emmerich.

Figure 24: Event notification label on the upper right corner depicts a short summery of the event.

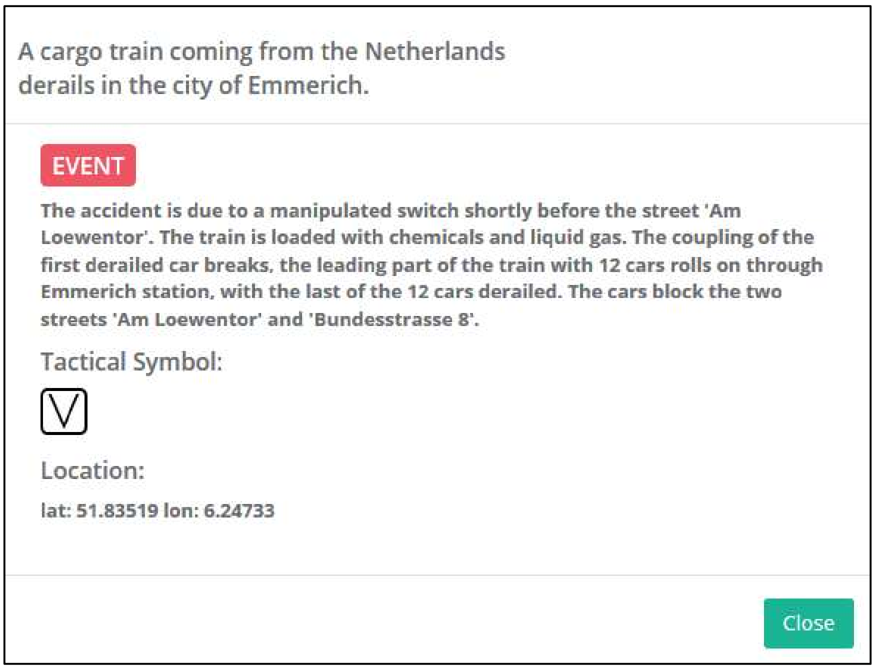


A cargo train coming from the Netherlands derails in the city of Emmerich.

Accident happend at:

10:30:00 2015/03/05

Figure 25: Information panel of a GIS element appears by clicking on it.



A cargo train coming from the Netherlands derails in the city of Emmerich.

EVENT

The accident is due to a manipulated switch shortly before the street 'Am Loewentor'. The train is loaded with chemicals and liquid gas. The coupling of the first derailed car breaks, the leading part of the train with 12 cars rolls on through Emmerich station, with the last of the 12 cars derailed. The cars block the two streets 'Am Loewentor' and 'Bundesstrasse 8'.

Tactical Symbol:

Location:

lat: 51.83519 lon: 6.24733

Figure 26: Window panel containing all information shows up by clicking on the element on the timeline or "Read me" button.

3.2.5.2 Using timeline

CIPRTrainer timeline (Figure 27) shows various kinds of events in a chronological order. Different event types have different colours (Table 2). To know more about the event, the user can click on the label. A window pop-up shows up similar to section 3.2.5.1 and can be closed by pushing the key `ESC` or click outside of the window.

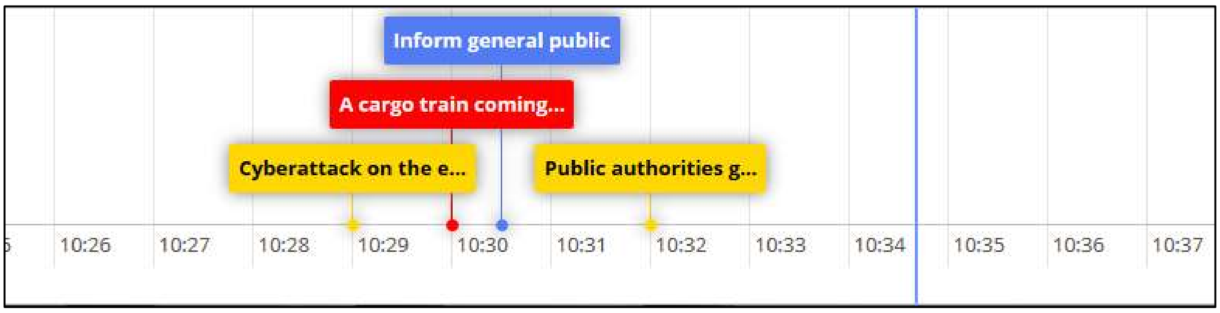


Figure 27: Timeline depicting various types of events.

Table 2: CIPRTrainer incorporates various types of events: action events, events defined in the SDL, events produced by the federated simulators and general non-georeferenced events.

Event Type	Colour		Location
Action events (performed by trainee)	blue	🟦	Partially
Events defined in the scenario	red	🟥	Yes
Events produced by the federated simulation	red	🟥	Partially
Non-georeferenced events and other events	yellow	🟨	No

3.2.5.3 Using notification logs

To examine certain events of any kind of events, following steps are required:

1. Click on the Notification Icon (**C2.9**)
2. On the filter input field type in the desired type of events (optional) to filter the list; possible types are:
   a. Action
   b. Simulator
   c. Scenario
   d. Others
3. Click on the desired list element; a window shows up with detailed information

Method 10: Using and filtering notification logs.

3.2.6 Managing user account and customising CIPRTrainer view components

The trainee can change first name, last name, email address and password in the settings panel (**C2.11**). This requires following steps:

1. Open the settings panel (**C2.11**)
2. Enter or update information. Possible input fields are:
    a. first name
    b. last name
    c. email address
    d. password (this includes entering old password, new password and conformation)
3. Click on 'Save' button to confirm or 'Cancel' button to discard changes

Method 11: Managing user account.

To customise CIPRTrainer, the user is enabled to show or hide certain UI components, of which there are:

14. Base Layers
15. Layers
16. Resources
17. Legend
18. Timeline

To show or hide UI components, open the settings panel, check or uncheck the desired components on the list (**C2.25**), respectively.

Method 12: Customising CIPRTrainer view components.

3.2.7 Conducting analysis methodologies

For using the Consequence Analysis Module (CAM), first click on the hyperlink 'What-if Analysis' (**C2.2**) on the navigation bar. The first section contains information about performed actions and rollbacks (**C2.37**, Figure 28). The graph is an $\boldsymbol{n}$-dimensional tree containing $\boldsymbol{x}$ nodes, where $\boldsymbol{n}$ reflects the number of performed rollbacks and $\boldsymbol{x-1}$ the number of performed actions. The root node (marked as yellow) illustrated the initial status of the scenario. Green nodes represent final actions, on which the user can perform the CAM. To examine the graph, you can:

19. Click on the nodes to read more details about the actions
20. Use the navigation controls of the network panel (Table 3)

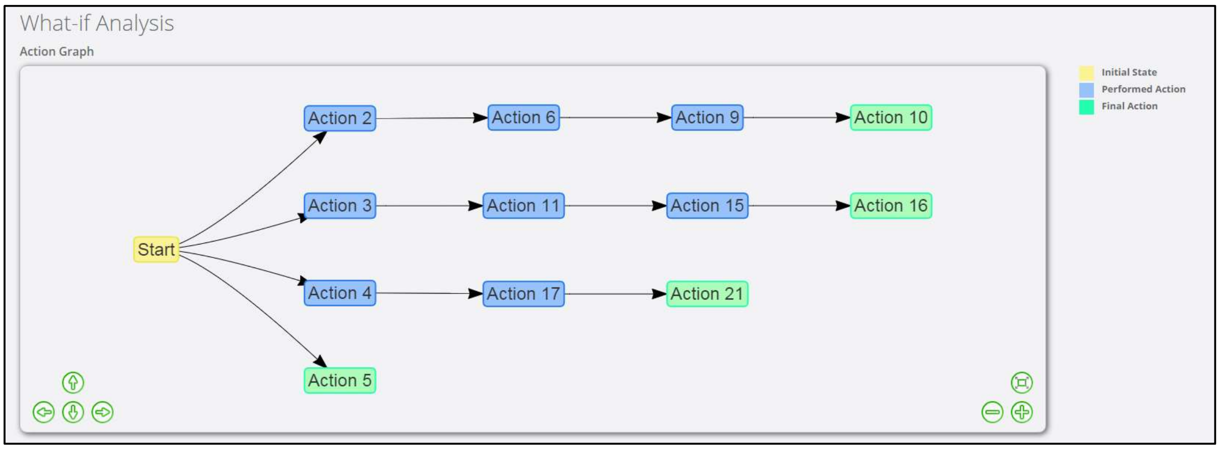


Figure 28: Example of action graph. Each node of the n-dimensional tree refers to an action. The rollback capability creates an additional branch, on which following actions can be added. Final actions are marked as green nodes. The root node corresponds to the initial state of the scenario. In this example, the end-user performed four rollbacks.

Table 3: Navigation controls for examining the action graph[1].

Icon:	⇧	⇩	⇦	⇨	⊕	⊖	⊡
Keyboard:	Up arrow	Down arrow	Left arrow	Right arrow	"="	"-"	None
Description:	Move up	Move down	Move left	Move right	Zoom in	Zoom out	Zoom extent

Please note, whenever the WIA view is active the simulation automatically pauses, if it was running, or remains in the state *stopped*, if trainer stopped the simulation before. To apply the CAM, follow these instructions:

1. Click on a green node in the action graph panel (**C2.37**)
2. Click on the "Acquire CA Results" button (**C2.38**)
3. Repeat **Step 1** for other green action nodes (optional)

Method 13: Examining WIA action graph.

When the "Acquire CA Results" button is clicked, several computation processes are stated in the backend. To get more details about how the CAM works and how to read the output, please take a look at deliverable D6.4 "Implementation of the integrated CIP MS&A based 'what if' analysis". All computation processes are conceptually described there. CIPRTrainer shows CAM results in two ways:

[1] http://visjs.org/examples/network/other/navigation.html

21. Tables (**C2.39**)
22. GIS map (**C2.40**)

The tables contain information about various kinds of damages without geospatial context, whereas the GIS map depicts several spatial-related damages using carefully chosen colour schemes to support map diagnostics provided by "ColorBrewer"[2].

3.2.8 Filtering CIs

Filtering CIs (KNN filter) requires following steps:

1. Open the filter panel (**C2.9**)
2. Select a CI or resource layer element to be filtered (KNN info panel **C2.20** will be updated)
3. Choose parameter $K \in \mathbb{N}$ (**C2.21**)
4. Apply filter by clicking on "Apply KNN Filter…" (**C2.22**)

Method 14: Filtering CIs.

3.3 Trainer module

This section describes trainer's possible interactions with the system. In addition to the capability of monitoring scenario and trainee's interactions, the trainer is able to set up scenarios and download computed results.

3.3.1 Choosing, starting and stopping scenarios

The trainer can choose a scenario using the Scenario List (**C3.6**). This requires few steps:

1. On the left side, explore possible scenarios
2. Read the description on the main panel
3. Click on "Start" button to start the simulation, whenever trainee is ready
4. Click on "Stop" button to stop the simulation, whenever the training session is consider to be done

Method 15: Choosing, starting and stopping scenarios.

Please note that the scenario cannot be started when the trainee is offline. Also, a training session theoretically does necessarily terminate, since rollbacks can be repeated any number of times. However, it is recommended to not perform more than 12 rollbacks due to performance issues. Any training session can be started and stopped as often as desired.

[2] http://colorbrewer2.org

3.3.2 Downloading consequence analysis results and training protocol

For downloading the CA results and training logs, please navigate to the download panel (**C3.8**) and click on:

1. `training_logs_`*scenario_city*`.json` to download the training logs in JSON-format
2. `ca_`*scenario_city*`.json` to download the CA results in JSON-format
3. Hyperlink “Download all” to download a single JSON file with both results CA and training logs

Method 16: Downloading CA results and training protocol.

The variable ***scenario_city*** contains the name of the city of the designed scenario. A complete result file contains meta-information, CA results and a list of all kinds of events, as the following excerpt shows:

```
{
 "meta": {
  "scenario id": "sce01",
  "trainee": {
        "id": "13",
        "username": "trainee",
        "email": "trainee@ciprnet.net"
        "first name": "Ute",
        "last name": "Mustermann"
  },
  "trainer":  {
        "id": "42",
        "username": "trainer",
        "email": "trainer@ciprnet.net"
        "first name": "Max",
        "last name": "Mustermann"
  },
  "start": "7841361201",
  "created": "7841630092"
 },
 "consequence analysis module": [...],
 "training protocol": [...]
}
```

PART B: DOCUMENTATION

4 How to get started

This section describes briefly the requirements including exact version numbers and a guide for installing components of the CIPRTrainer.

4.1 Requirements

The CIPRTrainer is developed on various distributed systems. The tools and software used for developing CIPRTrainer are listed below.

4.1.1 Operating systems

Windows:	Windows 7 Enterprise, Service Pack 1, 64 Bit
Linux:	Ubuntu 14.04.2 LTS

4.1.2 Databases and extensions

PostgreSQL:	Version 9.3.6
PostGIS:	Version 2.1

4.1.3 GIS tools

MapServer:	Version 6.4.1

4.1.4 Simulators

SINCAL:	Version 3.12.12
ns-3:	Version 3.24
OpenTrack:	Version 1.8.2
Flooding Simulator:	Version 0.3.1

4.1.5 Complex Event Processor

Java:	Version 8u91

4.1.6 CIPRTrainer application (v1.0)

Node:	Version 2.3.4

Node Modules: body-parser:1.15.1, bower:1.7.9, client-sessions:0.7.0, cookie-parser:1.4.1, ejs:2.4.1, express:4.13.4, gulp:3.9.1, gulp-concat:2.6.0, gulp-insert:0.5.0, gulp-less:3.1.0, gulp-mocha:2.2.0, gulp-plumber:1.1.0, gulp-rename:1.2.2, gulp-typescript-compiler:1.0.1, gulp-watch: 4.3.5, morgan:1.7.0, pg:4.5.5, socket.io:1.4.6 timer.js:1.0.4, winston:2.2.0

4.1.7 Web-server

Nginx:	Version 1.10.0

4.2 Installation

This section describes how to deploy the CIPRTrainer on a Windows, Linux and Mac OS X machine. There are two approaches on how to do so: Either by installing all components from scratch or deploying the components by using Docker tool suit.

4.2.1 CIPRTrainer GUI application from scratch

The first approach is to install the CIPRTrainer from scratch. First, each operating system needs to install the latest version of Node[3]. Then, having the latest release of the CIPRTrainer pulled from the Git[4] repository, the user has to navigate to the root directory of the git-project and enter the following commands in the Terminal or Command Line Tool:

```
# root directory of the ciprnet
$ cd src/ciprtrainer
$ ./install.sh # if unix-based OS, else $ ./install.bat
$ ./start.sh   # if unix-based OS, else $ ./start.bat
```

The first command changes the current directory to `src/ciprtrainer`. Then, the second command executes a shell or batch-script that contains the following instructions:

```
$ npm install
$ bower install # make sure bower is installed⁵
```

The command `npm install` installs a package and all the dependencies that it depends on in the local folder `node_modules`. The package (JSON file) looks like this:

```
{
  "name": "CIPRTrainer",
  "version": "1.0.0",
  "dependencies": {
    "express": "^4.13.3",
    "gulp": "^3.9.0",
    "socket.io": "^1.3.6"
  },
  "devDependencies": {
    "karma": "^0.13.9",
    "mocha": "^2.2.5",
    "should": "^7.0.4"
  },
  "scripts": {
    "start-production": "node ./app.js"
  }
}
```

The second command (`bower install`) installs packages to the folder `bower_components`. All dependencies like AngularJS or BootstrapJS are configured in `bower.json`. Bower components are mainly used in the browser. The shell script `start.sh` compiles and runs the CIPRTrainer. It contains the following instructions:

[3] https://nodejs.org

[4] https://git-scm.com

[5] Make sure bower is installed. If not, run the command `$ npm install bower -g`. If bower asks for the version of AngularJS, then choose the latest stable version.

```
$ gulp build # run gulpfile.js
$ sleep 5s # sleep 5 second to wait for the gulp task completion
$ export NODE_ENV = production # set the node environment in pro-
 duction mode
$ node app.js # run the application
```

Gulp is used for compiling the front-end dependencies for the CIPRTrainer (`ciprnetlib.js`). Since this requires some time, a sleep-instruction is added for 5 seconds. The environment variable is set to “production”. The last instruction starts the node application.

4.2.2 CIPRTrainer GUI application using Docker

The second approach to install the CIPRTrainer is using Docker. It enables the user “to package an application with all of its dependencies into a standardized unit for software development”[6]. Filesystem, system libraries, and tools are provided by the Docker container. The specific dependencies (libs/bins) for the application can be pulled from DockerHub, a public repository. These dependencies have to be defined into the Dockerfile.

The file (plain text file) contains all necessary configurations and dependencies for installing and running the CIPRTrainer:

```
# Dockerfile.txt
FROM node:4 # loads node dependencies from DockerHub (node dependen-
 cies)

RUN useradd -ms /bin/bash node # create user, create home directory,
 use bash

ADD . /home/node/ciprtrainer # copy src to new directory

RUN chown -R node:node /home/node # asign user to src

# load npm, bower and gulp
RUN npm install -g npm
RUN npm install -g bower gulp
RUN apt-get update

USER node # set the user node when running the image

EXPOSE 3005 # inform Docker container to listen on a specified network
 port at runtime

WORKDIR /home/node/ciprtrainer # change work directory

# install ciprtrainer
RUN npm install
RUN gulp build

CMD node app.js # run ciprtrainer
```

[6] https://www.docker.com/what-docker

To run Docker, Docker[7] needs to be installed. Docker provides a specific terminal for Windows machines. The user has to launch the Docker terminal. It starts with a configuration process, which assigns an `NAT-IP` address to the Docker Engine. Then, the user has to enter the following instructions:

```
# navigate to git project ciprnet
$ cd src/ciprtrainer # contains Dockerfile
$ docker build -t ciprtrainer .
$ docker run -p 80:3005 -t ciprtrainer
```

The first instruction navigates to the source files of the CIPRTrainer. The second instruction builds a Docker image, which contains execution parameters within a container runtime. The last command creates the Docker container, which is a runtime instance of the Docker image. The tag `-p` maps the exposed port of the container to the port on the Docker Engine. The CIPRTrainer is now accessible on the `NAT-IP` address of the Docker Engine on port `80`.

4.2.3 Setting up adaptors and simulators

The CIPRTrainer federated simulation environment describes the whole system which includes functionality for the coupling and orchestration of various CI-simulators. It is implemented through the *Simulator Adaptors* and the *Complex Event Processing* engine. Both components are described in detail within the deliverable D6.4 "Implementation of the integrated CIP MS&A based 'what if' analysis" [CIPRNetD64].

The current implementation of the CIPRTrainer comprises Simulator Adaptors for the CI-simulators PSS®SINCAL [SINCAL], ns-3 [NS-3] and OpenTrack [OpenTrack]. The architecture of the CIPRTrainer federated simulation environment allows an extension of the system with other simulators in principle. In this case, an implementation of a dedicated Simulator Adaptor that accesses the simulation software would be needed.

A Simulator Adaptor in the process is a single Java Application that implements a REST endpoint, which enables it to receive incoming events. Therefore, each adaptor needs at least to be configured and initialized with a receiver URL and a receiver port. Furthermore, the adaptor needs to know the URL and port of the CEP Engine, in order to establish a connection to this component for sending events. These parameters are provided to the application through a JSON-based configuration file. Depending on the actual simulation adaptor used, the configuration file can include more parameters. For the adaptor to the SINCAL simulator, a valid configuration file is defined through:

```
{
 "receiver_URI" : "localhost",
 "receiver_port" : 30666,
 "event_system_URI" : "localhost",
 "event_system_port" : 6666
 "sincal_database_uri" : ".\database.mdb"
 "sin_file_uri" : ".\Emmerich Distribution V26.sin"
}
```

[7] https://www.docker.com

Within the build environment of each simulation adaptor, an Apache Ant[8] build script is included. The build script includes a target for the start configuration of the according adaptor. During the application start-up, the URI of the configuration file is passed as a command line argument to the application. The following call will for example start the SINCAL adaptor:

```
$ ant -Dcfg_file=config_A.json SincalConnector
```

The operation of the federated simulation environment with the simulation adaptors presupposes the according simulators and simulation models available and installed. This means for the current state of the implementation that the CI-simulators PSS®SINCAL, ns-3 and OpenTrack need to be installed and configured properly.

The installation of the PSS®SINCAL software requires a Microsoft Windows operating system and is through the provided installer straightforward. Depending on the licensing model chosen, some additional steps to unlock the software might be required. The Simulation Adaptor accesses the calculation functions of the software through a COM interface. To activate this functionality in PSS®SINCAL, the COM servers need to be registered in the operating system. This can be done with the *PSS Tool* application, which comes with the PSS®SINCAL software package.

The telecommunication simulation environment ns-3 is a Linux based application. The software is distributed as source code, which need to be installed and compiled on the target system. A comprehensive introduction and installation instruction is provided on the ns-3 website[9]. The simulator for the CIPRTrainer's Emmerich scenario is provided as Tarball and needs to be installed on top of the ns-3 installation. After unzipping the archive, one will get

- a 'ciprnet' directory with thrift service definition
- an updated 'wscript' (to link all necessary external libraries)
- a "scratch" directory containing the simulation (`ciprnet_v3.cc`)

The ns-3 simulation is accessed through *Apache Thrift*, which needs to be downloaded and compiled[10]. After this, the `build.sh` script within the 'ciprnet' directory needs to be executed; this will generate a `libciprnetns3adapter.so` shared library. Before starting the application, the `LD_LIBRARY_PATH` environment variable has to be extended to refer to the 'ciprnet' directory within the ns-3 installation directory. The simulation can be started by invoking

```
$ ./waf --run "scratch/ciprnet_v4 -ThriftPort=8080"
```

The Port on which the Apache Thrift server listens for commands can be freely chosen, the port only needs to be reflected in the configuration file for the ns-3 Simulation Adaptor.

The railway simulator OpenTrack is a Microsoft Windows based software, which can be easily installed on any Windows Machine. For the CIPRTrainer software system, the OpenTrack API is used to implement the intercommunication between the OpenTrack simulator and the

[8] http://ant.apache.org

[9] https://www.nsnam.org/docs/tutorial/html/index.html

[10] https://thrift.apache.org/tutorial/

corresponding Simulation Adaptor. The OpenTrack API provides an interface to OpenTrack via SOAP over HTTP. To send commands to OpenTrack and receive commands via the API, OpenTrack needs to know about the TCP ports involved. By default, OpenTrack listens on port 9002 for incoming commands, and sends messages to applications listening on port 9004. Port 9004 would in the case of the CIPRTrainer software then be the port on which the OpenTrack Simulation Adaptor listens for incoming messages by OpenTrack. The user can change these ports within OpenTrack. Similar to the PSS®SINCAL and ns-3 Simulation Adaptors, the configuration of the ports needs to be reflected in the corresponding configuration file.

4.2.4 CEP Engine

The Complex Event Processing Engine is a Java project developed by Fraunhofer IAIS. It is started from a Linux machine that acts as host and uses port 3007 by default. All dependencies are already included in the compiled package so there are no necessary install requirements except for Java.

To start the CEP Engine copy the directory “EventProcessing” into your local home directory. First change the directory and file dependencies to match your current user:

```
$ chown -R $USER EventProcessing
$ chmod 755 /EventProcessing/start-cep.sh
```

Then navigate into the directory and start the CEP Engine:

```
$ cd EventProcessing
$ sh start-cep.sh
```

This will run the program on port 3007. If you need to use a different port just edit the start-cep.sh script and change the port that is given as parameter to the java command.

4.2.5 PostgreSQL and PostGIS

PostgreSQL is an open source Relational Database Management System (RDBMS) with advanced support for SQL. It is designed with an extensible architecture in mind. Extensions or plugins can be developed for the PostgreSQL providing extended features like spatial data management, foreign data wrapper, and cryptographic functions, etc. PostGIS is one of those extensions that provide sophisticated spatial data types and the corresponding functions to manage the data. Similar to the installation of MapServer, it is recommend as well updating the cached package index before starting the installation process:

```
$ sudo apt-get update
```

To install the PostGIS binary from the package repository, simply run the following two commands:

```
$ sudo apt-get install --no-install-recommends -y postgresql-9.3-
 postgis-2.1
$ sudo apt-get install --no-install-recommends -y postgis
```

After the binaries are installed, the PostgreSQL database server will run on port 5432 by default. A database server is capable to maintain multiple databases simultaneously. PostgreSQL extensions need to be activated for each database. Therefore if you want to enable the

PostGIS support in your database, it is still necessary to enable that extension by sending the following command to the database server:

```
$ echo "create extension postgis;" | sudo -u postgres psql
```

In the example above we use the client psql to communicate with the PostgreSQL server. You can also use any other client tool as you want. Important is that in order to successfully execute this command, the "super user" privilege is needed. Normal database users with different privileges are not allowed to run this command for security reasons.

The same to the installation of MapServer, it is also possible to install PostgreSQL and PostGIS from the source code. The whole source code repository can be downloaded with Git from the official website. After that checkout the version that you want to build and run the following command:

```
$ ./configure
```

If you want to build the client program psql, which uses readline, under Debian-based you can install it:

```
$ sudo apt-get install libreadline
```

The SQL query executor in PostgreSQL uses Bison and Flex for parser generation:

```
$ sudo apt-get install bison flex
```

After that running the make program will start the building process:

```
$ make
```

After the building process is finished, the executable file under src/backend/postgres will be produced and ready for use.

4.2.6 MapServer

MapServer is a software tool that focuses on efficient implementation of different web map protocols like Web Map Services (WMS) and Web Feature Services (WFS), etc. To provide an efficient implementation, it uses the C/C++ programming language, which in general has better runtime performance than other virtual machines based languages like Java. Unlike other mapping generation tools like GeoServer, MapServer itself does not have a built-in Web component to handle HTTP request and response. Instead, it relies on other dedicated Web Servers like Apache or Nginx that are capable for Common Gateway Interface (CGI) or Fast-CGI communication. It is the classic way for C-based programs, which are not primarily designed for Web-based communication, to interact with the Web clients.

Our production environment is Linux-based. Various pre-built binary distributions are available in most of the Linux-distributions. As an example, the Debian-based distributions are used to show how to install MapServer. To start the installation, it is always a good practice to update the index files of the binary packages by running:

```
$ sudo apt-get update
```

The GNU Make system is needed to perform several pre-installation tasks for MapServer binary installation, therefore we need to install it before installing the MapServer binary by running the following command:

```
$ sudo apt-get install --no-install-recommends -y make
```

After the dependent tool is installed, the binary packages and corresponding configuration files can be fetched by issuing the following command in the command line shell:

```
$ sudo apt-get install --no-install-recommends -y mapserver-bin
```

Another option is to install from the source code. This requires however advanced techniques in software development and system administration. The benefits are the configuration flexibility during the building process, i.e. a dedicated version of the produced binary files that meet the requirements can be created. For instance, if it is clear that no WMS support is need, a binary version without WMS support is more compact and produces better runtime performance.

5 Conclusion

This deliverable introduced a documentation and user manual of a new interactive web application, CIPRTrainer v1.0. CIPRTrainer realises one of CIPRNet's new capabilities, namely 'what if' analysis for exploring different courses of action in simulated crisis situations. CIPRTrainer is dedicated for training crisis managers in the area of civil protection / disaster relief, making them familiar with the behaviour of Critical Infrastructures under severe perturbations.

The implementation of CIPRTrainer integrates modern Web 2.0 technology and best practices in GIS. This manual illustrated CIPRTrainer's functions for exploring complex scenarios, performing WIA, and managing the user profile. Example scenarios were designed specifically for crisis managers. The situation display incorporates national standard tactical symbols from Germany (as ruled by the German BBK, the Federal Office for Civil Protection and Disaster Assistance) and the Netherlands, for a better comprehension of the displayed information. The main view of CIPRTrainer incorporates a GIS map, two information panels (left and right), one navigation-bar and one timeline. It was designed to provide an easily manageable feeling for the trainee.

This deliverable also describes the installation of the CIPRTrainer system and its components. Complementary to this manual, we have developed a demonstration movie for illustrating the features of CIPRTrainer. The forthcoming deliverable D4.5 will describe a web-based service for demonstrating CIPRTrainer's essential new WIA capability to a wider audience.

6 User interface component index

UI number	Description
C1.1	Landing page
C1.2	Hyperlink to About view
C1.3	Sign-In panel button
C1.4	Language button
C1.5	Sign-in username input field
C1.6	Sign-in password input field
C1.7	Sign-in button
C1.8	Toggle for saving or deleting session cookie after training
C1.9	About view
C2.1	Hyperlink to Home view
C2.2	Hyperlink to WIA view
C2.3	Label for user role and username
C2.4	Sign-out panel button
C2.5	GIS map
C2.6	GIS element
C2.7	Pop-up window
C2.8	Hyperlink control panel
C2.9	Hyperlink filter panel
C2.10	Hyperlink notification panel
C2.11	Hyperlink settings panel
C2.12	Action list
C2.13	Action history list
C2.14	“Continue/pause” button
C2.15	Rollback button
C2.16	Label simulation time
C2.17	Label real time
C2.19	Filter information panel
C2.20	KNN filter table
C2.21	KNN parameter input field
C2.22	“Apply KNN filter” button
C2.23	Filter input field
C2.24	Notification list
C2.25	Layer list
C2.26	Label username (not changeable)
C2.27	Input field first name, last name, email
C2.28	Input field old password, new password, confirmation password
C2.29	“Save” and “Cancel” button
C2.30	Base layer list
C2.31	Resource layer list
C2.32	CI layer list for power networks
C2.33	CI layer list for telecommunication networks

C2.34	Scenario event on timeline
C2.35	Other event on timeline
C2.36	Current simulation time
C2.37	WIA action graph
C2.38	“Acquire CA result” button
C2.39	CA result table
C2.40	CA result GIS map
C3.1	Label scenario state
C3.2	Label simulation time
C3.4	Label trainee status
C3.5	Label trainer status
C3.6	Scenario panel
C3.7	Notification/training logs panel
C3.8	Download panel

7 Index

8 References

[CIPRNetD61]	EU FP7 CIPRNet, Fraunhofer, Deliverable D6.1 – Conceptual design of a federated and distributed cross-sector and threat simulator, 2014.
[CIPRNetD62]	EU FP7 CIPRNet, CEA, Deliverable D6.2 – Application Scenario, 2014.
[CIPRNetD63]	EU FP7 CIPRNet, Fraunhofer, Deliverable D6.3 – Federate CI Models, 2015.
[CIPRNetD64]	EU FP7 CIPRNet, Fraunhofer, Deliverable D6.4 – Implementation and integration of the federated and distributed cross-sector and threat simulator, 2016.
[CIPRNetD756]	EU FP7 CIPRNet, ENEA, Deliverable D7.5+D7.6 – Integration of meteo-climatological simulators, flood forecasts, earthquakes data and analysis tool and Interface to the technical demonstrator for federated CIP MS&A, 2016.
[CIPRNetD83]	EU FP7 CIPRNet, UCBM, Deliverable D8.3 – Textbook: Training Material on DSS and Training Material on CIP MS&A based 'what if' analysis, forthcoming 2016.
[NS-3]	ns-3 home page http://www.nsnam.org (last access on 16/07/2015)
[OpenTrack]	The OpenTrack Home Page http://www.opentrack.ch (last access on 16/07/2015)
[SINCAL]	The SIEMENS PSS$^{®}$ SINCAL Home Page http://www.sincal.de (last access on 16/07/2015)