WP 2.2+2.4 Project Deliverable

Selection of developer tools, Specification of planned system capabilities (software)

Project Number	IST-2000-29266
Project Title	Virtual Real Time Fire Emergency Simulator
Deliverable Type	Report
Deliverable Class	Internal

Deliverable Number	D2.2 + D2.4
Title of Deliverable	Selection of developer tools [WP2.2] Specification of planned system capabilities (software) [WP2.4]
Nature of the Deliverable	Report
Contributing WPs	WP2.2, WP 2.4
Contractual Date of Delivery	14. December 2001
Actual Date of Delivery	14. May 2002
URL	www.virtualfires.org
Authors	Sascha Schneider (FIGD), Michael Schmidt (FIGD), Volker Luckas (FIGD), Gunther Lenz (SiTu), Thomas Reichl (SiTu), Wilhelm Brandstätter (CD), Christian Redl (CD), Gert Svensson (KTH), Kai-Mikael Jää-Aro (KTH), Alain Bochon (Apletunnel), René-Michel Faure (CETU), Klaus Schäfer (FDDo), Rainer Koch (FDDo), José Luis Ajuria (EUVE), Christian Redl (CD),
Contact Details	Institute for Structural Analysis / SiTu Research Univ. Prof. Dipl.-Ing. Dr. techn. Gernot Beer Lessingstrasse 25/II 8010 Graz / Austria Tel.: +43 316 8736180 Fax: +43 316 8736185 Email: gernot.beer@ifb.tu-graz.ac.at

Abstract	The summarized results of WP2.2 and WP2.4 are presented in this report. The developer tools are selected and the planned system capabilities (software) are specified to give an overview.

Keywords	Developer Tools, Specification, Software, Simulation, Visualization

Contents

1 Selection of developer tools

Based on the features and capabilities of the available developer tools presented in WP2.1 (cp. delivery report for WP 2.1) the project partners, who are responsible for software programming the VIRTUALFIRES project, decide to choose the following environments for developing.

1.1 Software languages

To guarantee the desired platform independence and good code migration between project partners the developers select the following programming language:

1.1.1 Ansi C++

C++ is the most sophisticated programming language nowadays. It has good performance and distribution together with good documentation. Many dialects are available. For that reason the standardized version of C++ is chosen: Ansi C++. On Windows for example it is possible to install the Intel Compiler Plugin into the Microsoft Visual Studio to get a developing platform supporting Ansi C++. On Unix/Linux it is possible to use GCC to implement in Ansi C++.

1.1.2 Fortran 90/95

Fortran 90/95 is still the most widely used programming language for scientific simulation. Also it so possible to link code written in Fortran with C++ code or even convert existing and new developed Code to C++, Fortran 90/95 will be used in the VIRTUALFIRES Project in the simulation part of the software.

1.2 GUI library

For the implementation of the GUI library the project partners decided to use:

1.2.1 QT 3.0

QT supports all desired features needed in the software developing part of the project. It allows OS and platform independent software development and includes many intelligent features like database integration and OpenGL (cp. 1.3.1) support. For a complete list of features of QT we refer to the delivery report for WP 2.1.

1.3 Graphic Library

The project partners decide to implement the graphic subroutines in the following library:

1.3.1 OpenGL

In professional visualization software development OpenGL is a common standard with good documentation, user and driver support. It allows to implement all modern visualization algorithms using the newest and of course state of the art techniques to realize them. For a complete list of the features of OpenGL we refer to the delivery report for WP 2.1.

1.4 Simulation

The software packages used in the simulation part will be the following:

1.4.1 Fluent

As already explained in the delivery report for WP 2.1 Fluent a general purpose CFD code supporting the integration of user defined functions. Together with its other features it convinced the project partners to use it in VIRTUALFIRES.

1.4.2 ICE

ICE is developed at CD and therefore offers the chance to profit from the knowledge directly available from one of the project partners. It provides the necessary criteria for development needed within VIRTUALFIRES and therefore is chosen to be one of the simulation software packages that will be used.

1.5 Inter-Process-Communication / Threading

For the inter-process-communication and the threading abstraction, the developers of VIRTUALFIRES decided to use MPI and Corba.

1.5.1 MPI

The Message Passing Interface comes as a C++ library which offers the possibility to implement communication operations between threads and applications. It is again platform independent and has a wide field of application areas in the industry. For further details on MPI please refer to the delivery report of WP 2.1.

1.5.2 Corba

Doing the inter application communications with Corba allows the developers to implement their specific code parts independently from the other project partners. Just the communication interfaces have to be defined. Corba is like Ansi C++ standardized and available in many distributions on all platforms.

1.6 Documentation

For the documentation of the developed code within VIRTUALFIRES doxygen was selected.

1.6.1 Doxygen

Doxygen is a documentation system for C, C++ and IDL (Corba, Microsoft and KDE-DCOP flavors). It can generate an on-line documentation browser (in HTML) and/or an off-line reference manual (in LaTeX) from a set of documented source files. There is also support for generating output in RTF (MS-Word), Postscript, hyperlinked PDF, compressed HTML, and Unix man pages. The documentation is extracted directly from the sources, which makes it much easier to keep the documentation consistent with the source code.

Doxygen can be configured to extract the code structure from undocumented source files. This can be very useful to quickly find your way in large source distributions. The relations

between the various elements are be visualized by means of include dependency graphs, inheritance diagrams, and collaboration diagrams, which are all generated automatically.

2 Specification of planned system capabilities (software)

2.1 User Expectations

In this section an overview of the desired (software) system capabilities are specified and described from the end users point of view.

2.1.1 Alpetunnel

The European Economic Interest Group Alpetunnel was set up with the support of European Union to carry out the studies required to define the international segment of the railway link Lyon - Torino.

The project concerns :

- high speed passenger trains,
- heavy freight trains,
- shuttles carrying trucks and their drivers : the rail road highway.

The international segment is essentially a base tunnel between St Jean de Maurienne in France and Susa in Italy. It is 52.7km long. The tunnel will be a twin-bore structure with a cross section of 43sq.m, with a railway line in each tube. The tubes will be interconnected by cross passages each 400m.

The Modane underground emergency station will be located at the tunnel mid-point. There will be three other intervention stations inside the structure providing access to the outside. All the stations are provided with facilities to limit fire with sprinkler systems and for firemen intervention.

<u>The principles of fire protection :</u>

If a fire breaks out when a train is in the tunnel :

- the train goes out of the tunnel,
- if this is not possible, the train goes to the main emergency station or one of the secondary emergency stations,
- if this is not possible, the train stops in the tube and passengers walk to the other tube through the interconnecting cross-passages.

Alpetunnel has carried out investigations to study evacuation of a train on fire in the tunnel section :

- heat, volumes and contents of smokes released,
- smoke propagation in the tunnel depending on tunnel ventilation level,

- risk assessment for any passengers caught in the smoke,

- dimensioning of the ventilation and exhaust units to handle trains on fire in emergency stations.

The method is : pressurised free tunnel and controlled smoke. The system is able to extract the smoke and blow fresh air to protect the passengers and the conductors. The requisite means are four fan stations.

The sprinklers has to limit the fire until the arrival of the firemen.

The more interesting applications of the simulator could be :

- the evaluation of the designed cross passages and other galleries in the underground station,

- the evaluation of the visibility and efficiency of emergency signs,

- the evaluation of the designed sprinkler system,

- the evaluation of the different systems for smoke extraction : one extraction point with 200m^3/s, five extraction points with 40m^3/s.

It would be useful to simulate the different scenarios considered by Alpetunnel : passenger train with weak fire load, freight train or shuttles carrying trucks with high fire load.

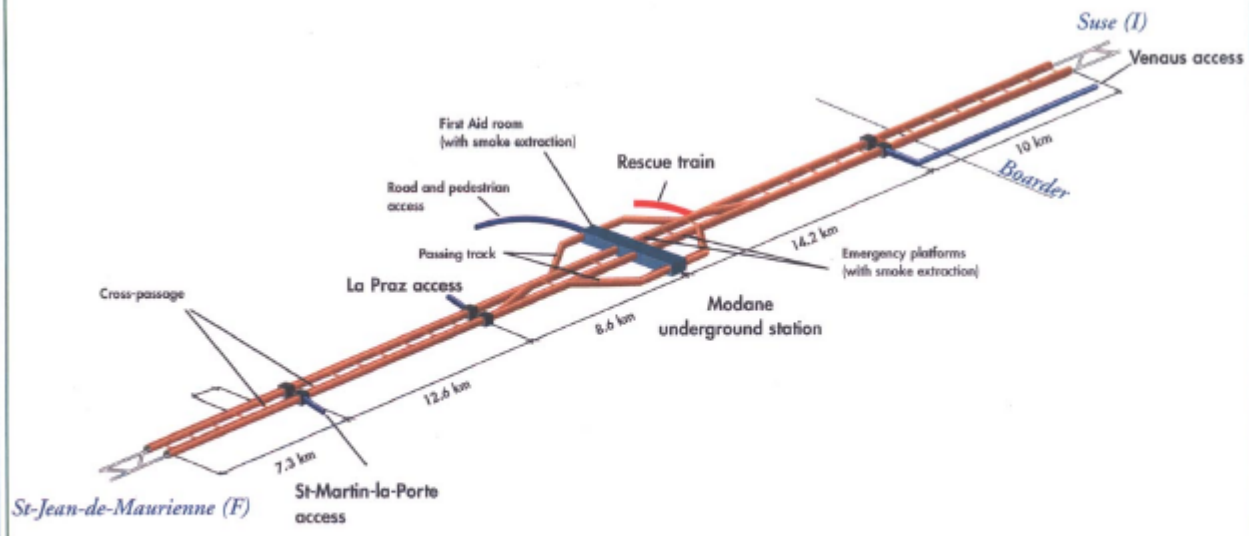
In our case, the designed system has to take charge of passengers and trucks and train conductors. The simulator could help us to understand their behaviour and escape conditions.

Data input :

The geometrical data will be imported from AutoCAD files. The other data will be : fire load, rate and speed of air flow in the railway tunnels.

New transalpine railway connection Torino - Lyon

General base tunnel layout



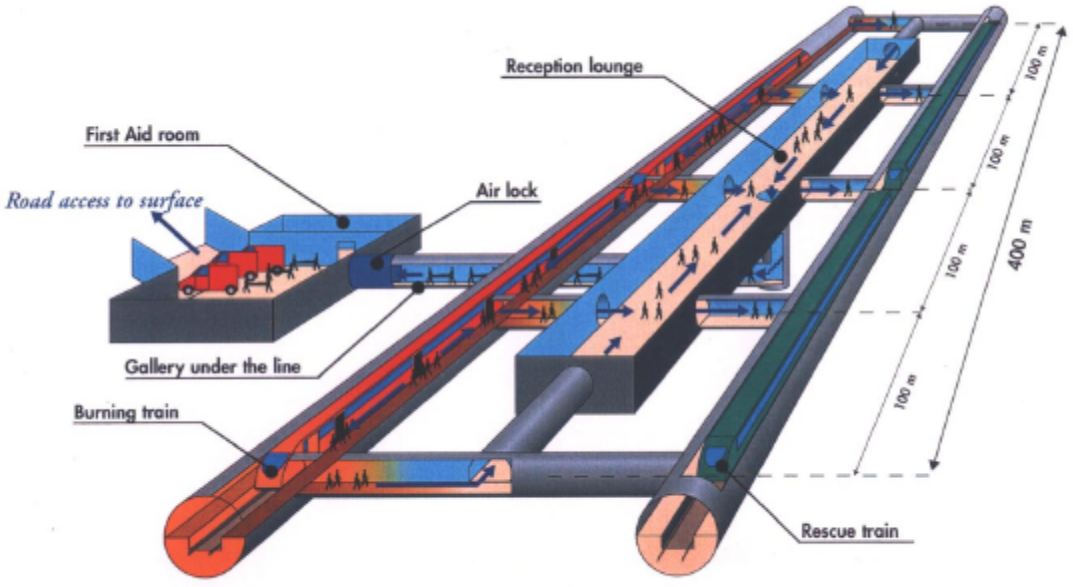
11/09/2000

Alpetunnel

New transalpine railway connection Torino - Lyon

The method used to evacuate passengers from a burning train in the underground station of Modane

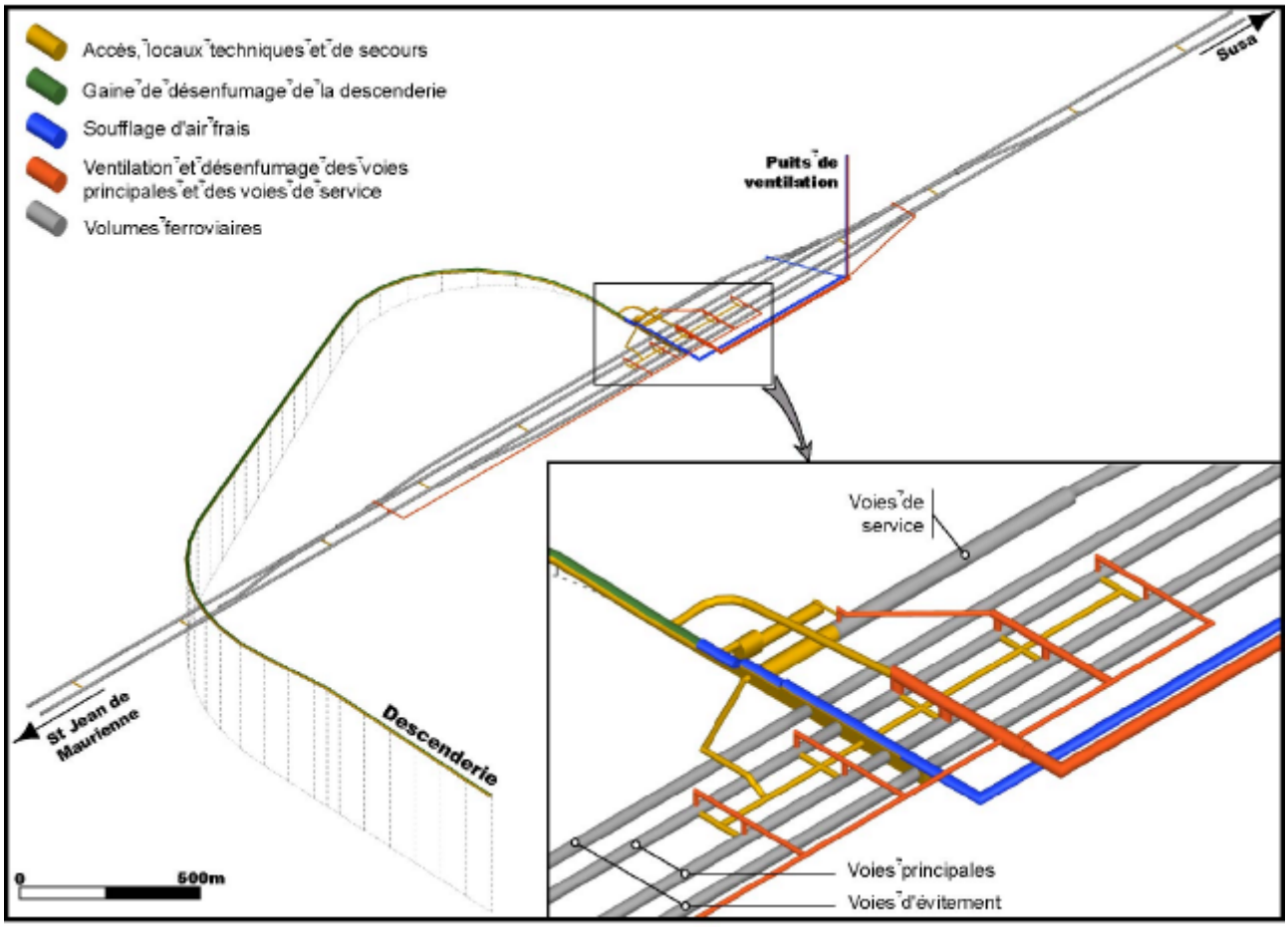
(option with reception lounge)



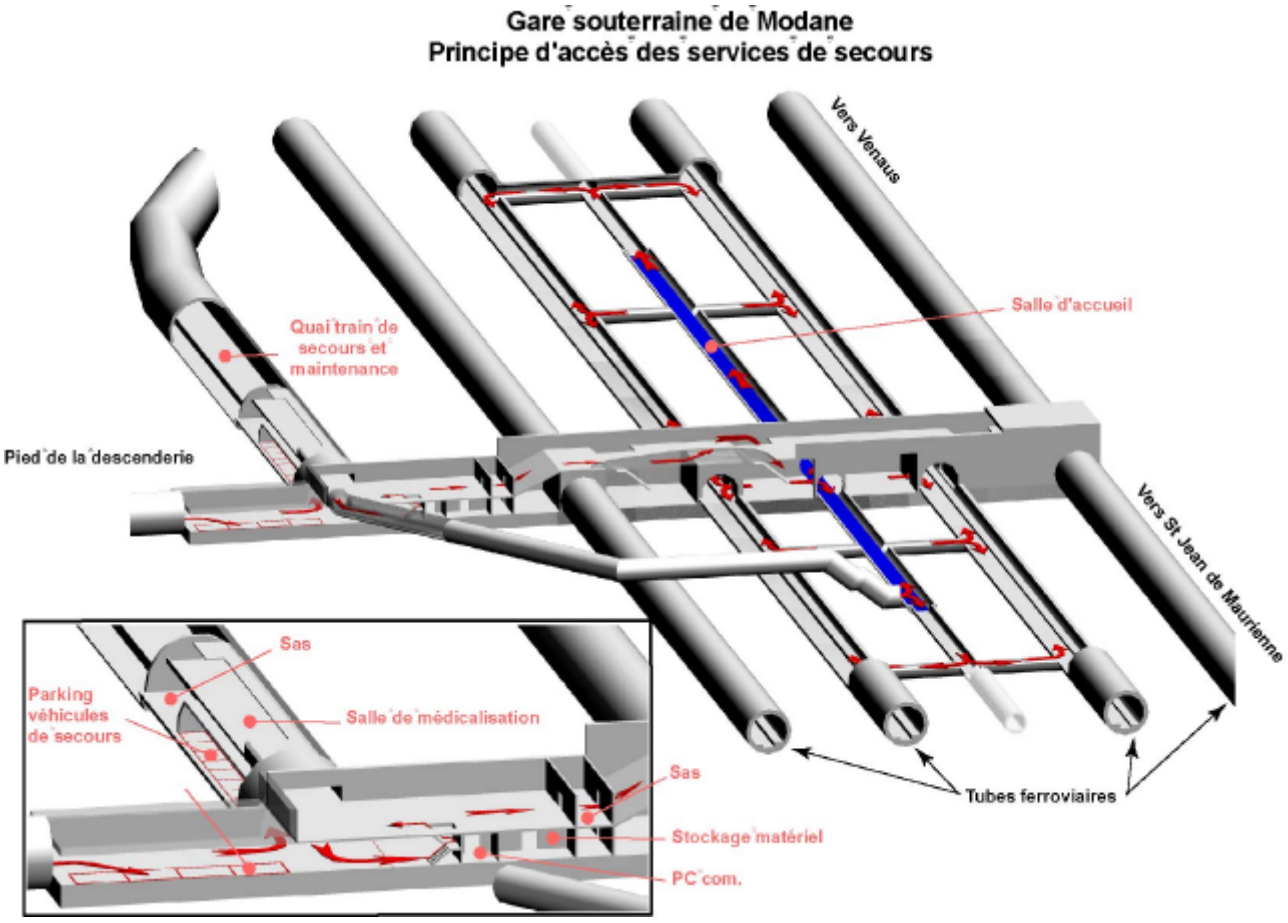
11/09/2000

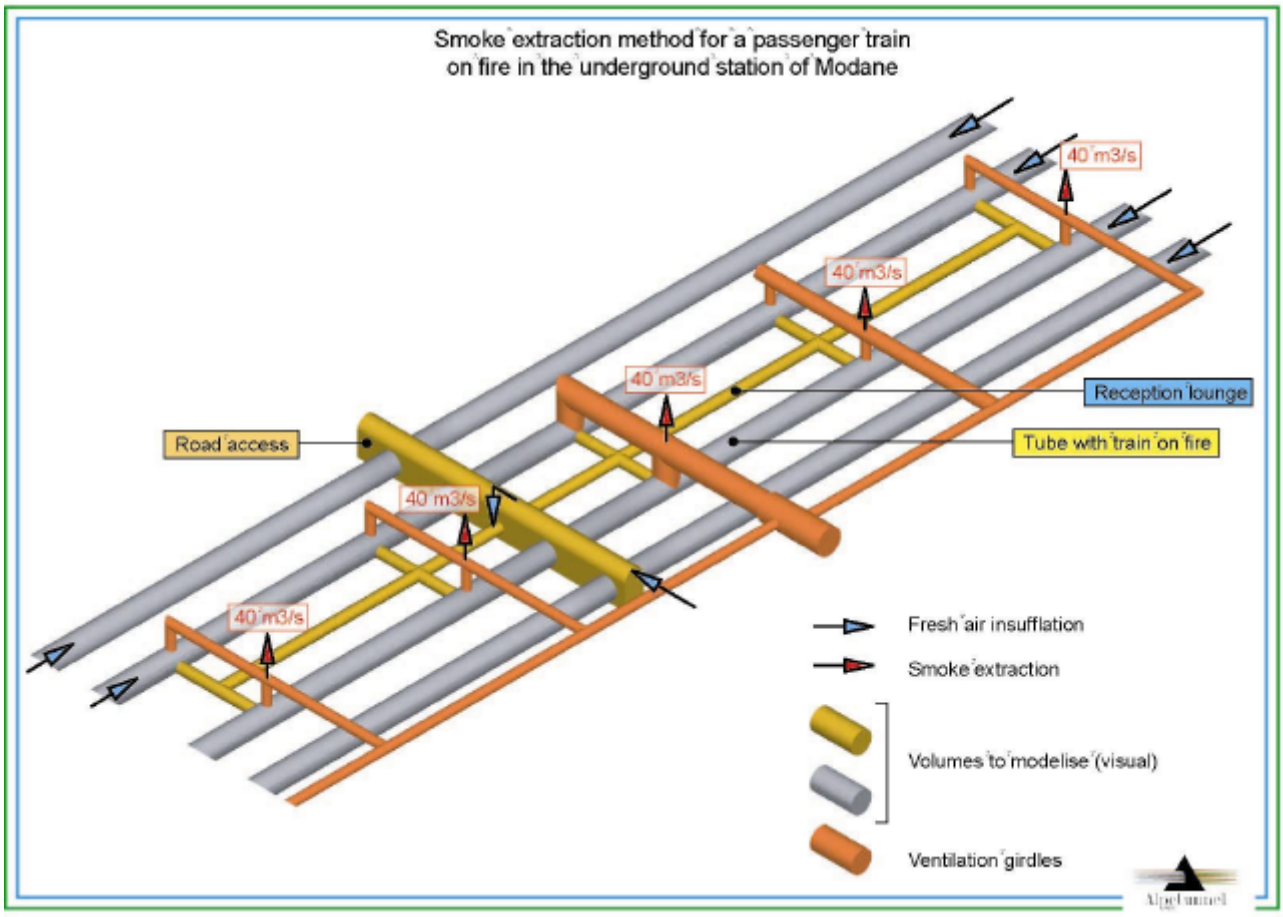
Alpetunnel

General underground station layout



Firemen access in the underground station of Modane





2.1.2 Cetu

The main computer activity of CETU is based on a network of near 100PC, under Windows. Each year about 20PC are changed to the last release. 2 workstations (SGI with Linux) are also used for specific calculation (projects and research).

From VIRTUALFIRES we are waiting for two computational levels

1. static simulation where situation = f(t) for a given geometry

CETU uses CFD modelling for ten years and achieved three thesis. For this item CETU needs comparisons with previous calculations. This 1st level may be also "animation" from several static simulations previously computed.

2. dynamic simulation where situation = f(t, men action, material behaviour)

This new possibility may be developed by

- 2.1: scenario to be previously defined
- 2.2: acting any time on boundary conditions

For the second level CETU plans to buy the adequate display system and VIRTUALFIRES is a good opportunity for a choice.

2.1.3 Euve

EUVE's role in the project is validation of the results, this task will consist on checking is desired capabilities are met and explore future possibilities. One of these possibilities come from the aeronautical and railway industry, and some points could be explained in order to expand system capabilities.

Just to offer some classification of capabilities, they could be divided into three categories: data input interface, numerical results and visualisation.

Data input: the definition of the problem to be solved should be the most straight forward from the existing electronic representation. In the case of EUVE, geometrical data could be imported from our mainly used **CAD** program, CATIA. Perhaps, if data from civil engineering companies in Spain are extended to other countries, AutoCAD could be another possibility. For system use, most advanced interfaces will be used, and for instance a **joystick** should be used for movements and a tracking system in the HMD or in the glasses to know head position.

Numerical results: the most important data from will defined from the safety of the user (to prevent injures in the people inside the tunnel and the fire-fighters) and the safety of the construction itself (to prevent collapse). Then, data that will be essential output will be **temperature and concentration of toxic gases**. In case of very busy tunnels, and in the future in trains and planes, it is important to take into account the co-existence with other people and with objects on the ground that will make more difficult the way out. In case of "animated" obstacles like other persons, it will be necessary to define its move inside the tunnel and detect **collisions** with them, and with other objects.

Visualisation: as the main interface of the application will be graphical (and perhaps sound) all data to be displayed will be suitable to be displayed in the CAVE or in the HMD. From the visualisation point of view, some way it should be displayed temperature and toxic gases concentration. At the same time it is very important in order to design evacuation plans to see as the people will see, and then smoke density should affect visibility. It is also desirable to have the capability of showing and hiding the position inside the tunnel, because people will not have this information in case of high smoke density. The collisions detection algorithm will show when it detects it doing impossible to move in the corresponding direction, and perhaps with some sound and lightning the object without revealing its complete form.

Although this facts are desirable, there should be a consensus approach to define final capabilities that are reachable in the scope of the project.

2.1.4 FDDo

Specification of System Requirements in the View of Fire Brigades

2.1.4.1 Introduction

The main applications of the VirtualFires-simulation will be:

- the evaluation and validation of fire fighting measures and tactics inside existing tunnel and subway-systems,
- the risk analysis of planned tunnel and subway-systems,

- the evaluation of fixed fire fighting resources,
- the demonstration of tunnel fires and hazards for training and teaching purposes.

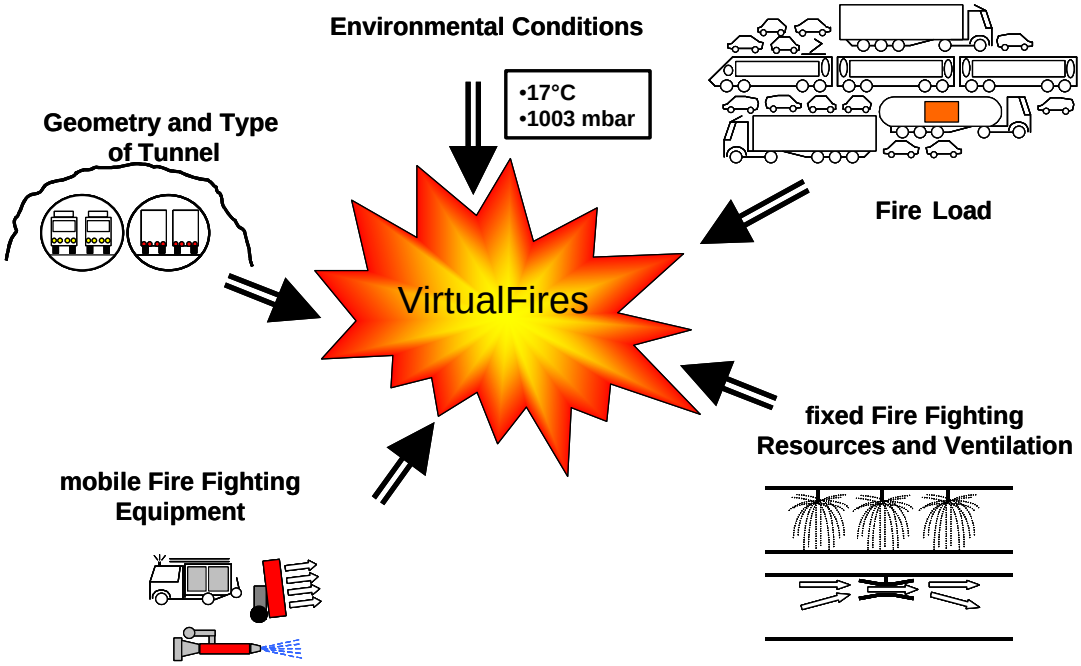


Figure 1: Input Information for Simulation

Therefore it is required to define realistic scenarios (fig. 1), including

- realistic tunnel geometry incl. installations (geometry and textures)
- variable fire loads incl. hazardous goods (which are normally linked to vehicles in the tunnel system)
- the option to define different origins of fire
- and to simulate the impacts of different types of fire fighting resources

A typical process of the simulation has the following steps:

For an existing tunnel with fixed geometry the fire loads and the origin of fire are defined. Based on this the simulation shows the spreading of smoke, fire, gas and heat distribution. The user then can decide to activate fixed fire fighting resources or can define missions for the fire brigade. The impact and effects of these measures modify the further simulation.

2.1.4.2 Structure of simulation

From this the following structure of the simulation can be deduced (fig. 2):

The object of simulation (e. g. a street tunnel or subway-tunnel) is described by the

building itself, installations e.g. traffic signs, built in equipment, and the vehicles. These objects are normally fixed during the simulation, but affected by the fire, and influence the fire simulation by their fire load. So the **Fire Area** can be defined as a dynamic subset of the buildings, installations and vehicles.

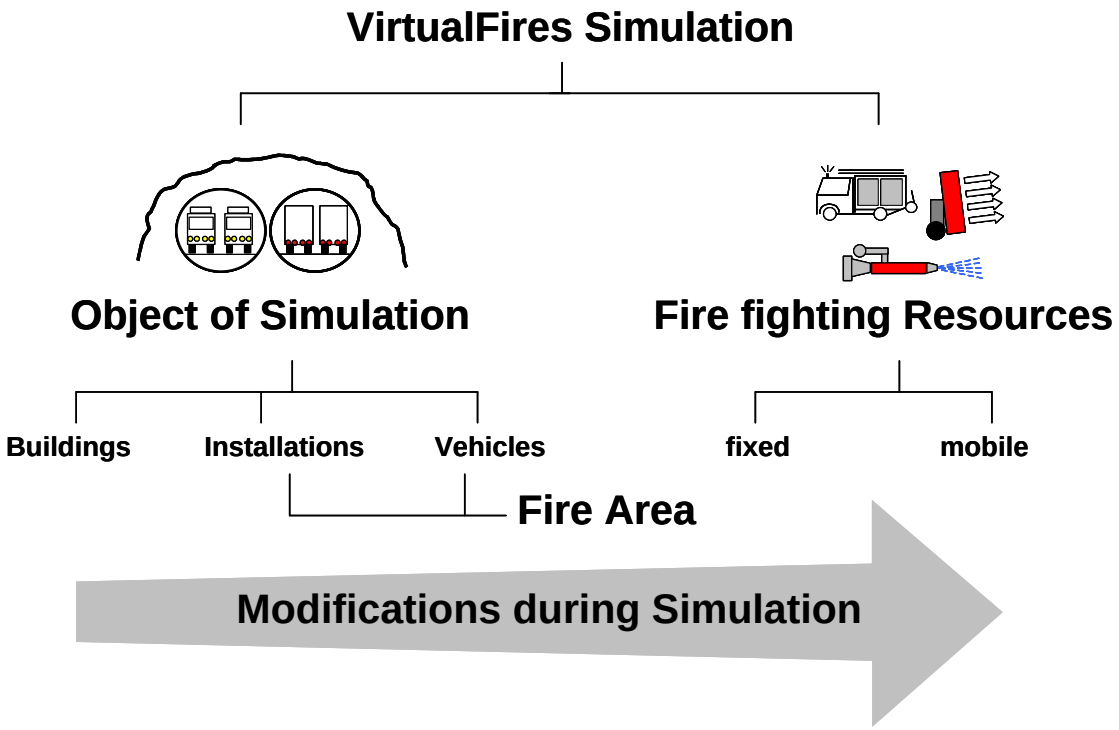


Figure 2: Structure of Objects

Concerning the fire fighting resources there are two different types:

- The fixed fire fighting resources can be activated by the user and have an impact on the fire, they may as well be affected by the fire. Usually they have no or little delay time. Examples are sprinkler-systems, artificial ventilation, robot systems.

- Normally the mobile fire fighting equipment is operated by fire fighters. To activate mobile fire fighting resources the user has to define the mission and the attack-point, where the tasks shall be carried out. From this, a delay time may result. This delay time can be predefined and it depends on the set-up time for the fire fighters´ personal protection equipment and the size and movability of the equipment. An example is given later on. Normally mobile resources are not affected by fire.

During a simulation –especially of existing tunnels and subways - the highest frequency of changes will be the activation of fire fighting resources. These simulations should include the environmental conditions outside the tunnel such as

- temperature,

- barometric pressure and
- wind direction.

These conditions may change during daytime and by this may influence the area included into the simulation.

The objects of the simulation (fig 3.) are defined by their geometry and have textures for realistic visualisations. The parameters describe the behaviour in case of fire (the detailed specification which parameters can be included should come from CD). In this context the installations can be defined as built-in objects, which may influence heat, gas and fire distribution or bring in fire loads, e. g. kiosks, ticket machines in subways. Illuminated signs, which give the direction to the nearest emergency exit, are installations as well.

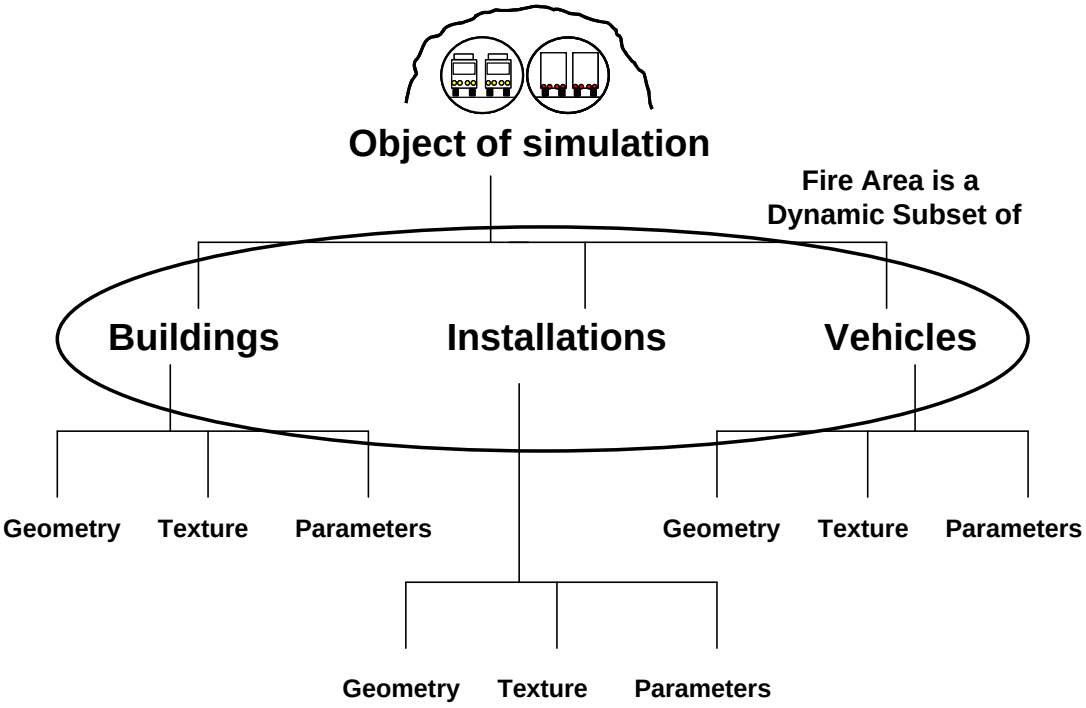


Figure 3: Objects of Simulation and definition of fIre area

Emergency exits are fixed fire fighting resources, because they may influence heat, gas and smoke distribution when the are activated (opened).

2.1.4.3 Description of Objects

The FDDo will apply the Virtualfires simulations for street tunnels as well as for subway systems. For the street tunnels the following data can be given:

- The cross-sections of the tubes can be circular, oval or rectangular.
- The street tunnels consist of two or three tubes with two carriage ways.
- The length is up to 6.5 km (planned).
- Usual installations are emergency exists to an additional rescue tube, traffic signs and sign-posting.

The subway systems of course are more complex. for realistic simulations not only the tubes but also the stations should be included into the simulations. The following coarse description can be given:

- different types of stations
    - single level stations with only one platform and two connected tubes and stairs and elevators
    - three-level crossing stations with up to three platforms on one level, multiple connections and installations
- different types of tubes
    - circular, oval or rectangular cross-sections
    - additional sidings, where trains can be parked
    - additional emergency exits connected to the tubes

The average distance between two stations is 600 m, the maximum distance is about 1,3 km. If the distance exceeds 300 m an emergency exit is built in between the stations.

For realistic simulations of fires in tubes at least two stations have to be surveyed, because the fire brigade can access from both directions through the stations or an emergency exit.

If an origin of fire within a station is simulated at least three stations have to be included in the simulation:

- the station, where the origin of fire is and
- the neighbouring stations

The number of neighbouring stations included depends on the specific situation. In general, all stations and sidings connected by tubes have to be included, additionally all levels connected by stairs and escalators. Another influence comes from the environmental conditions outside the tunnel. E. g. if smoke, heat etc. are driven into one direction by strong flow the nearest and the nearest but one station has to be surveyed.

To sum up, the simulation should cover an subway area of about 5 to 5 km, where

different types of stations and connections can be included. So these requirements will not exceed the requirements of ALPETUNNEL, but the requirements concerning the photo-realistic presentation may be higher.

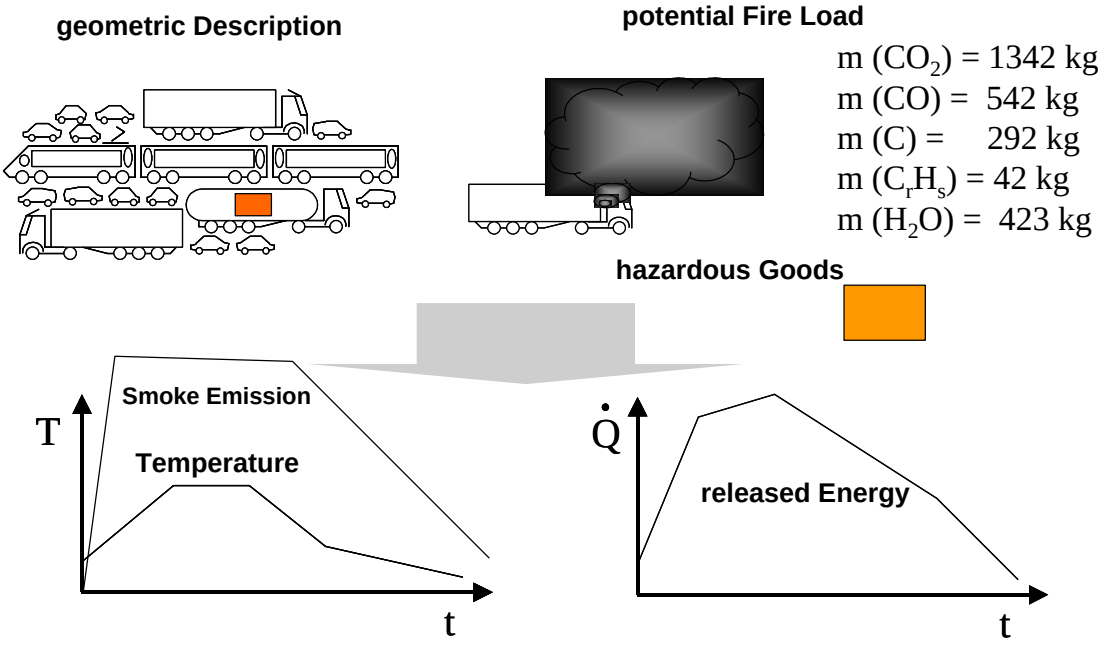


Figure 4: Example of fire loads

For all the objects potentially affected by the fire, the fire loads and the resulting temperature and energy release have to be defined (fig. 4). For street tunnels the data of hazardous goods may additionally be included.

2.1.4.4 Simulation of Fire Fighting

Main point of the simulation in the view of fire brigades is the evaluation and optimisation of fire fighting tactics and measures as well as fixed fire fighting resource. These fixed or semi-mobile resources can be characterised e.g. by

- their locations,
- their coverage,
- the physical impact,
- the way of activation and
- the delay or set-up time

As mentioned before mobile fire fighting resources usually are operated by fire fighters. So the user has to define so called **missions**, which describe

- the point of attack,
- the measures,
- the personal protection and
- additional equipment.

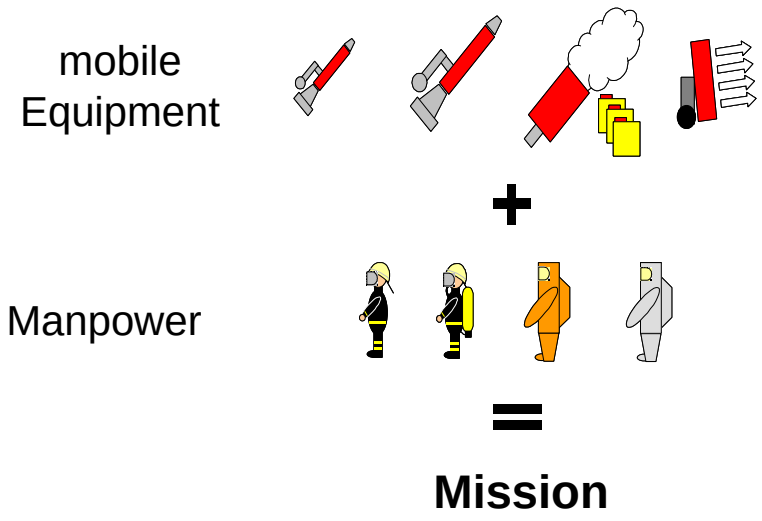


Figure 5: Defining mobile missions

From these data the impacts and duration of a mission can be calculated during a simulation, provided that the toughness of the personnel and the fire fighting equipment are described by adequate parameters (s. following figs 6 and 7)

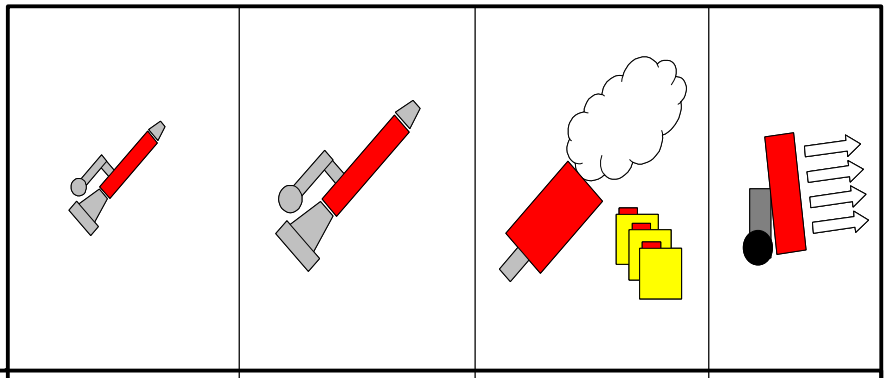
				
medium	water	water	foam	ventilation
flow rate	100 l/min	400 l/min	25000 l/min	500 m³/min
coverage	15 m	30 m	> 5 m	-
impact of the measure	X	Y	Z	V

Figure 6: Parameters of mobile fire fighting equipment

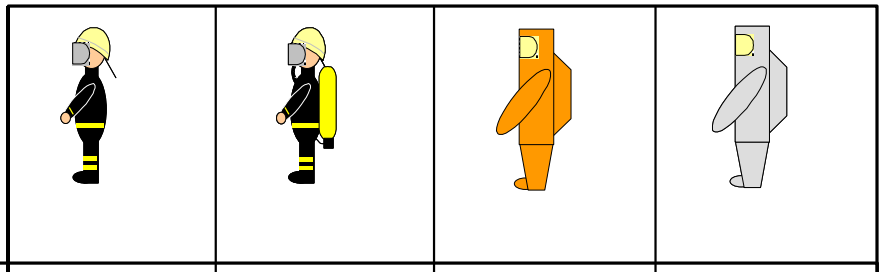
				
breathing protection	partial	full	full	full
temperature limit	< X °C	< X °C	< Y °C	< Z °C
movability	usual	reduced	limited	limited
speed of motions	high	average	low	low
chemical protection	no	no	full	low
operating time	long	average	low	low
heat protection	average	average	low	high
set-up time	low	average	high	high
. . .				

Figure 7: Parameters of fire fighting personnel

In general, the fire fighting equipment defines the impact of measures and the protective clothing and equipment of the personnel define the operating time. The set-up time is influenced by the fire fighting equipment as well as by the equipment of the personnel.

The following figure gives an example of the definition of a mission: The types of missions can be predefined (e.g. “Cool down” (objects or smoke)). Additionally the required fire fighting equipment and details for its application have to be selected or given. From this the impact of the mission can be calculated. Additional parameters describe the personnel and their personal equipment.

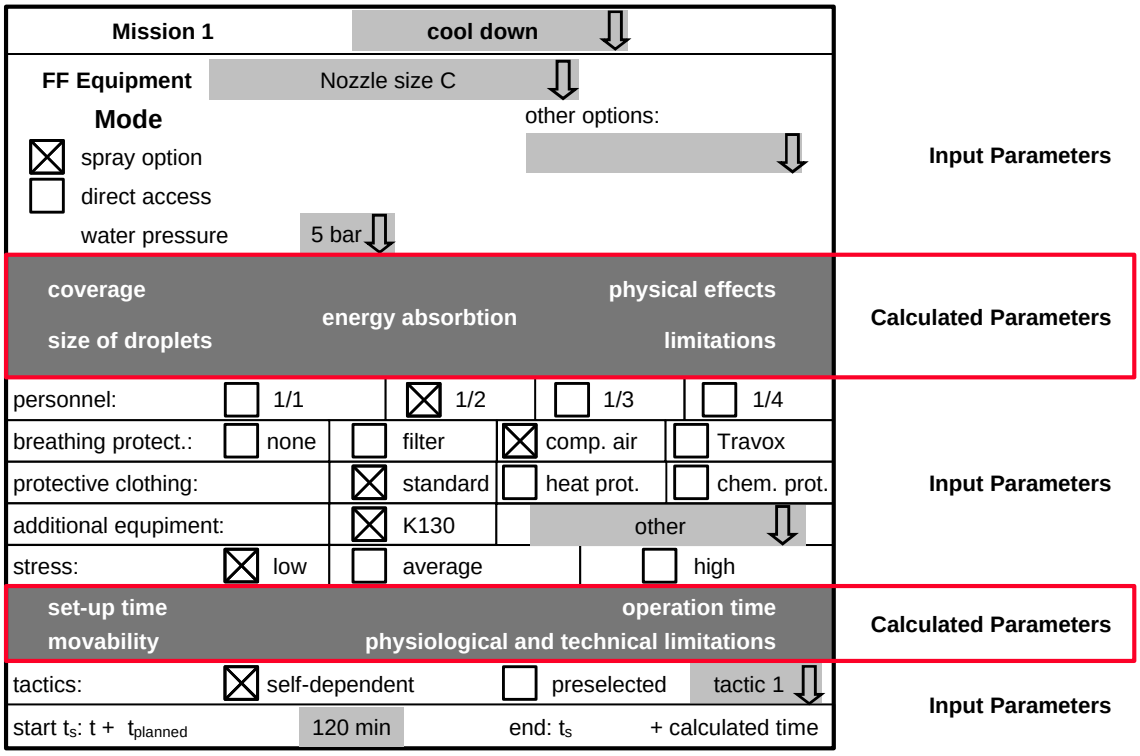


Figure 8: Defining a mission scheme

From these data describing equipment and personnel set-up times, operating times and limitations and the operating speed can be estimated. The point of attack additionally should be defined by graphical input, so that the whole mission can be calculated.

2.1.4.5 General Requirements

With a look at the time consuming operations and processes a fully real-time simulation of a tunnel fire is questionable. As seen in actual accidents the fire fighting can last for several hours or even days.

Additionally the following figure 9 gives an idea of the whole operating sequence:

- The time of detection can be estimated up to 5 to 10 minutes.

- The time for the approach of the fire fighting unit is planned up to 8 min. or more, depending on the location and the traffic situation.
- After the arrival several missions maybe defined nearly parallel.
- The typical intrusion speed can be estimated at 10m/min. in a tunnel filled with smoke.

From this the implementation of fast motion, break points and slow motion functionalities are recommended. By this the user could define scenarios and missions to find out critical situations via fast motion. These situations and places can be inspected in detail with the help of slow motion. Additionally the simulation can be rewound to breakpoints, e.g. starting points of missions, to redefine a mission. From this it is necessary to include multiple mission into the simulation (fig.10).

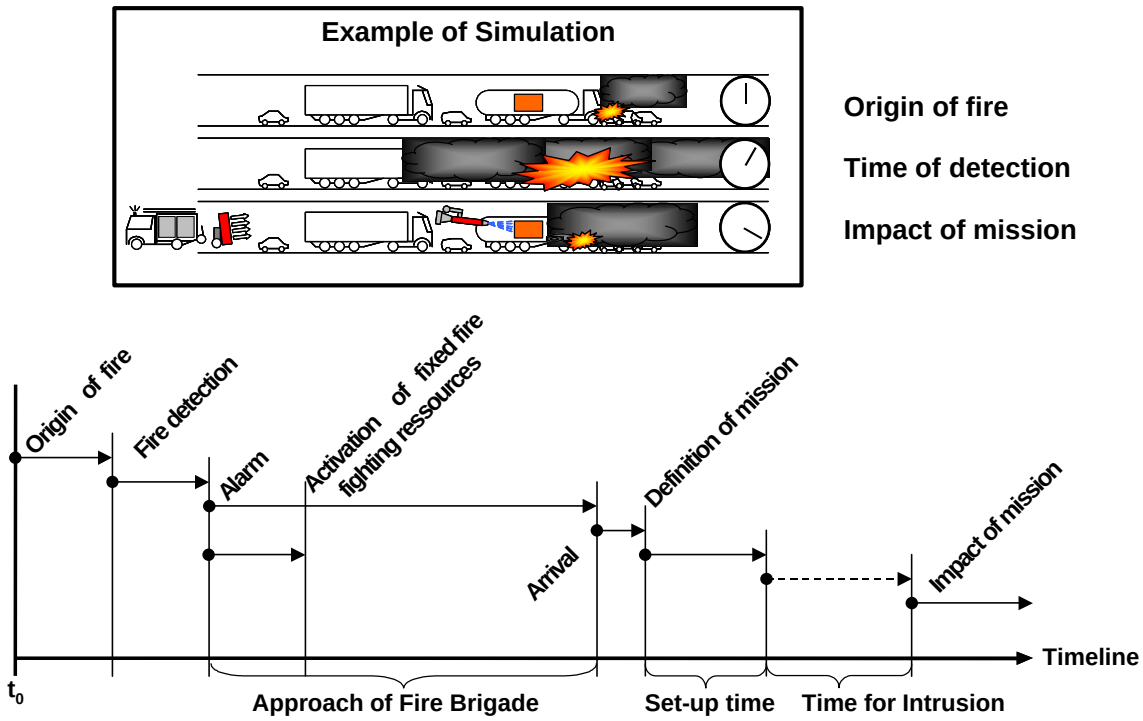


Figure 9: Sequence of fire fighting and rescue

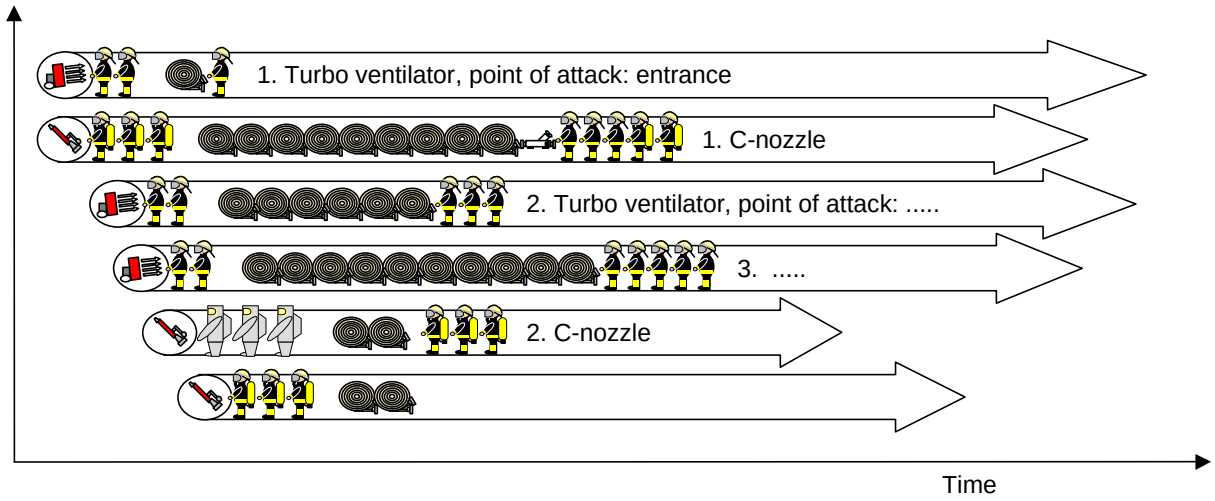


Figure 10: Multiple fire fighting missions

As pointed out in the Graz meeting in November three types of applications can be seen:

- the detailed evaluation of buildings, fixed fire fighting resources, risks and fire fighting tactics,
- the demonstration of tunnel fires and the related measures for preparatory training of fire fighters,
- a general information for laymen, e. g. drivers.

For the detailed evaluation the user will define different scenarios, equipment and missions to assess the risks of specific objects. This will be done by one person or small teams. The focus is on the scientific visualisation (colour mapping,...) of heat, smoke and gas distribution and a symbolic visualisation of missions. Additionally the already mentioned highly interactive functionality for setting breakpoints, slow motion etc. is required and the system should produce reports of the measure respectively show the state of missions when they are identified interactively. This system for the all-day use should be high-end PC-based with adequate graphical output devices (HMD, Shutter glasses, small projection system). For specific situations (very large objects, high risks, approval processes) CAVE-implementations are useful.

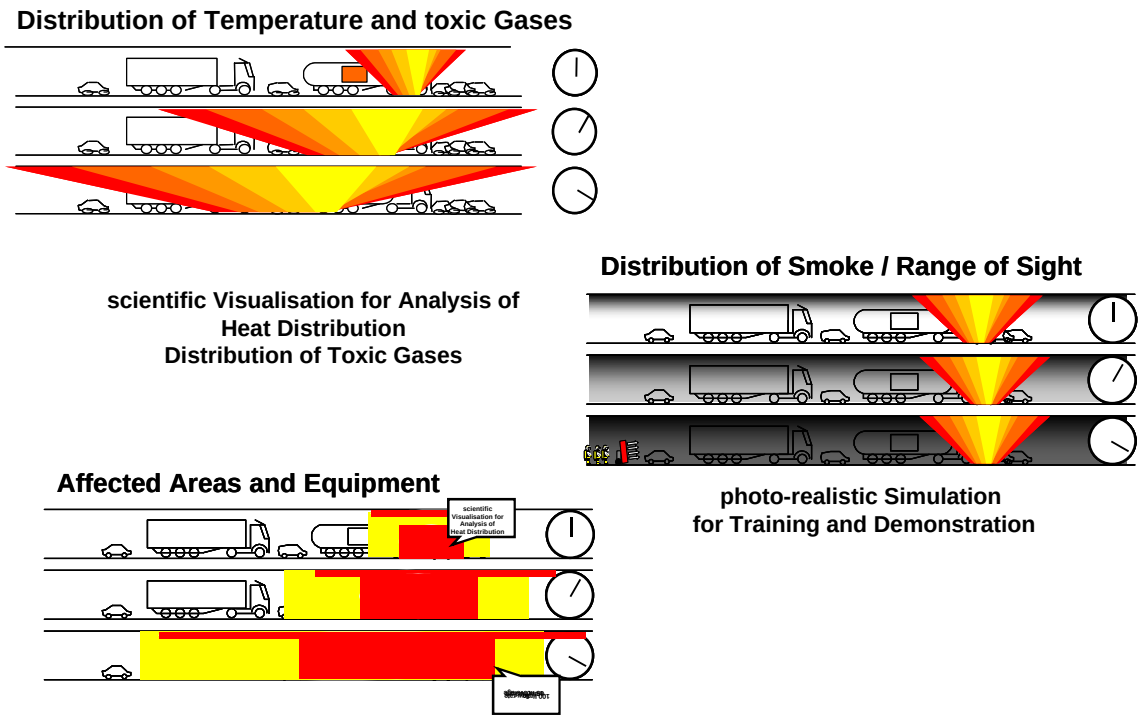


Figure 11: Different views during simulation

For demonstration purposes photo-realistic simulations are required. These demonstrations normally will be predefined with predefined view points and selected animated sequences. The focus is on the impressive visualisation of the scenarios and processes including a close to reality visualisation of missions and equipment.

2.1.4.6 Test Cases

As pointed out before, the simulation of missions and the impacts of fixed fire fighting resources are of great importance for the fire brigades. The Lyon meeting clarified, that simulations like this have not been done until now. From this it is recommended, that at first test cases should cover this problem, and that we turn to the simulation of complex structures later on, when the simulation of the test cases have been successful.

From this first a large tunnel with a simple structure should be the basis. Within this tunnel different fire loads can be placed and missions can be defined.

To evaluate the missions a closer look to an area of about 20m to 40 m should be possible. in the first draft the missions can be defined by approximation of geometric features and physical impacts of fire fighting resources and equipment. The effects of multiple missions could be simulated by scaling and multiple applications of the defined base-missions and resources (s. Appendix 3). The descriptions of the equipment and the related physical impacts and the layout will be corresponding to the needs of the CFD-simulation and the visualisation.

2.1.4.7 Dictionary

English	**Deutsch**
Mobile Fire Fighting Equipment	Mobiles Löschgerät
Environmental Conditions	Umgebungsbedingungen
Geometry of Tunnel	Tunnelgeometrie
fixed Fire Fighting Resources	stationäre Brandbekämpfungsanlage
Fire Load	Brandlasten
Origin of Fire	Entstehungszeitpunkt des Feuers
Fire Fighting	Brandbekämpfung
Barometric Pressure	Luftdruck
Hazardous Goods	Gefährliche Stoffe und Güter
Released Energy	Freigesetzte Energie
Artificial Ventilation	Belüftungsanlage
Semi-mobile Fire Fighting	Eingebautes, bewegliches Löschgerät
Foam	Löschschaum
Coverage	Reichweite
impact of the measure	Wirkung der Einsatzmaßnahme (z. B. Wärmebindung)
set-up time	Rüstzeit
Nozzle	Strahlrohr
Nozzle size	Strahlrohr- Düsengröße
Distribution of Temperature	Temperaturverteilung
Distribution of toxic Gases	Ausbreitung giftiger Brandgase
Distribution of Smoke	Rauchausbreitung
Range of Sight	Sichtweite
Affected Areas and Equipment	Schadensstelle

2.2 Specification from the developers point of view

In this section the required system capabilities (software) are described from the developers point of view

2.2.1 CD

2.2.1.1 General Overview

The first task of CD is to carry out flow and combustion simulation in tunnels with various ventilation systems using the commercial CFD code FLUENT.

The second and primary task is to develop the real time simulation program based on the Lattice Boltzmann method.

2.2.1.2 Simulation

The commercial CFD program FLUENT is taken to simulate various fire hazards within tunnels. The resulting data can be used for visualisation purposes. Using FLUENT one will be able to vary the size and location of different fire loads. It is possible to change the operating conditions of the ventilation system.

The Lattice Boltzmann based flow solver ICE will be used for the real time simulation. Due to the tremendous computational effort it will be able to calculate about 100.000 computational cells in real time. This yields a spatial resolution of about 20 [cm] and an extension of the computational domain of about 10 [m] in each direction.

The final target is to allow concurrent interaction of the user with the flow solver. The user should be able to activate fixed fire fighting devices, change ventilation system operating conditions, etc..

2.2.1.3 Input data

Design drawings or a CAD files of the tunnel cross section is required in order to set up the computational mesh which is the basis for the simulation. Additionally detailed information about the ventilation system (location and size of fresh air inlets and exhaust air outlets, amount of fresh air, ...) and fixed fire fighting equipment is needed. The activation time of the fire fighting equipment may be predefined by the user.

For the user interaction during a concurrent simulation one has to define an interface which is suited for both the HMD and the CAVE. The specification of this interface will be done in correspondence with the developers of the visualisation system.

2.2.1.4 Output results

The simulation programs deliver information about flow velocities, temperature and smoke concentration. These are written to an output file in CGNS format. These CGNS files can be processed directly by the virtual reality system or can be processed to be stored in a database for future reuse.

2.2.2 FIGD

2.2.2.1 General Overview

From the FIGD point of view, the general layout of the whole System should be designed like the structure in fig 12.

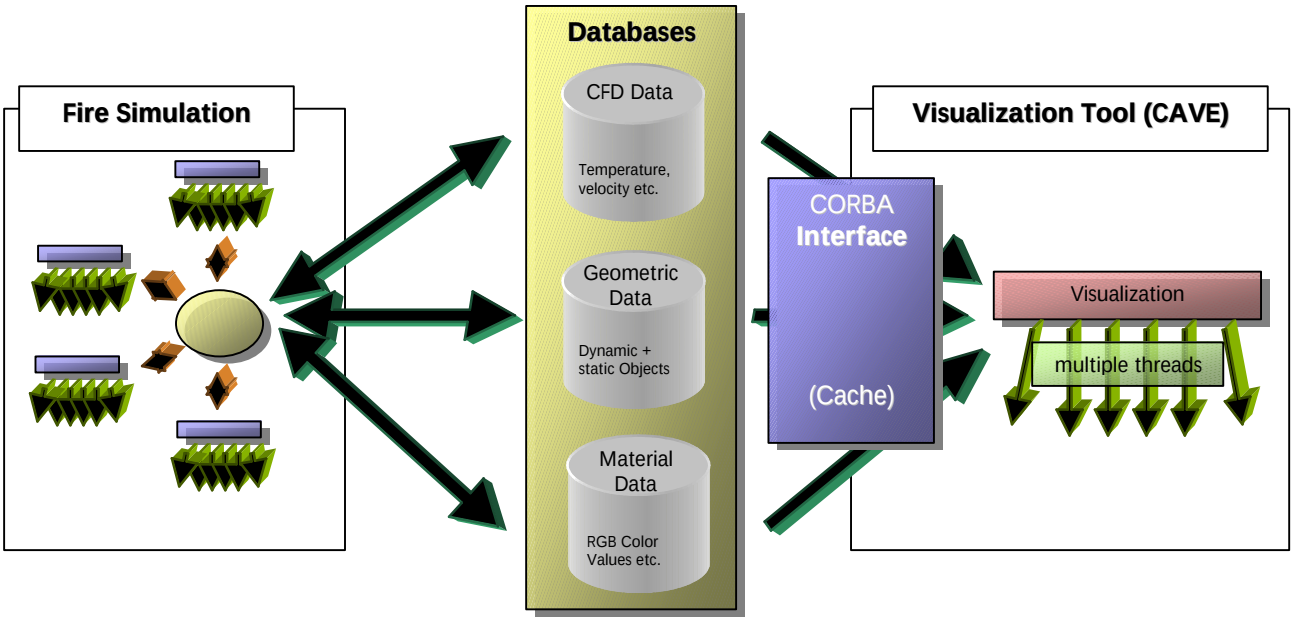


Figure 12: general system layout

2.2.2.2 Simulation

The simulation is docked to a database server which holds the necessary information like the geometry (tunnel, cars, etc.) and the material data. The results of the simulation ("CFD data") like temperature or gas and smoke density are stored in another database. Furthermore the simulation part needs a more or less complex interface for the user data which has influence on the simulation (as can be seen in section 2.1.4).

2.2.2.3 Data-server + Interface

The data server(s) hold the necessary information for the simulation and the visualization and will be reachable through an abstract interface (implemented in Corba as specified in WP 2.3). Together will modern caching techniques it will reach the required performance and abstraction levels. The abstract (Corba) interface will guarantee the independency of the visualization from the simulation and the data storage (server).

2.2.2.4 Visualization

The main part of FIGD in this project is the massive parallel visualization of the simulation results which can be seen in the right part of fig. 12. All necessary data for the visualization are transmitted from the database servers through the interface mentioned in the former section. Then the visualization can start to process the received data and to visualize it. For this task scientific and realistic visualization methods are (re)implemented in a massive parallel way. During runtime the user will have the possibility to chose the desired visualization methods. The selected methods are stored are derived from a generalized

class and stored in a central container abstraction which supports massive parallel processing as well. The calculated visualization objects are then projected to the chosen display (i.e. CAVE, HMD or Monitor).

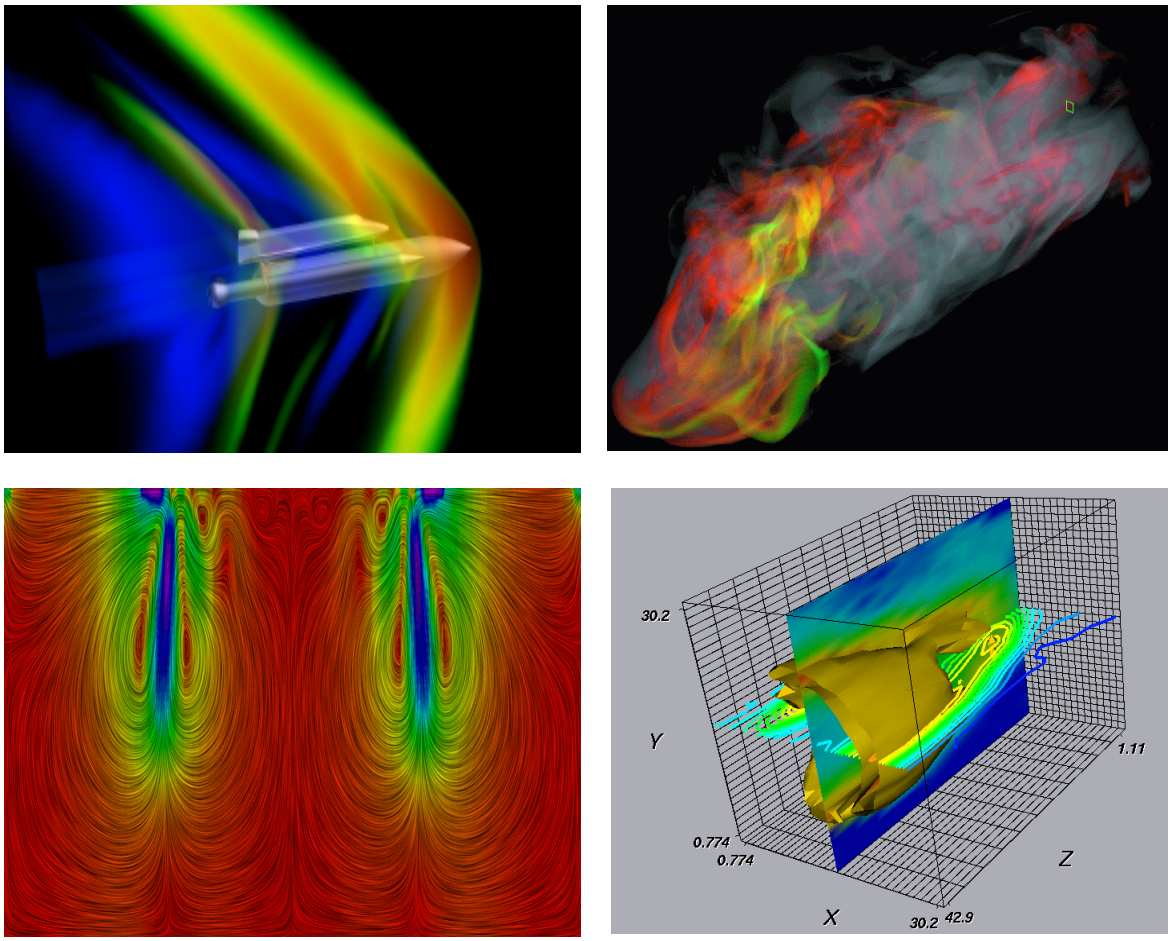


Figure 13: some visualization methods (upper left to lower right):
volume rendering, smoke + heat visualization, line integral convolution (LIC) and iso surfaces

2.2.3 SiTu

2.2.3.1 General Overview

SiTu's main part of the development will be the design and implementation of the database for the simulation environment. It will perform the following tasks:

- provide services for storage and retrieval of the required data for the CFD calculation
- compress and filter the resulting CFD data to preserve storage volume and transmission bandwith
- provide the services for retrieval of the required scene data for the VR system.

2.2.3.2 Scenarios

2.2.3.2.1 realtime simulation

Within this scenario a user of the system can not only watch the spreading of the fire from its starting point, but is also able to set measures against it and watch their reactions immediately. Due to the large amount of data that is produced during an interactive run of the simulation and its demands for low latency a direct coupling of the CFD calculation and the VR system is preferred. In an active simulation the boundary conditions for the CFD-calculation are changed interactively and are forwarded directly to the CFD-solver. The results of the CFD calculation are transmitted back to the VR System.

2.2.3.2.2 a-priori simulation

Here the total timeline of actions is predetermined in advance and stored into the database. In this scenario there is a timely decoupling of the CFD calculation and its visualisation in the VR system. The CFD solver performs its calculation based on predefined data. The resulting dataset for all timesteps is stored into the database. The VR system fetches the model of the tunnel for an initial display and a part of the CFD-data surrounding the users position and renders it according to the selected visualisation mode.

2.2.3.3 Data Flow

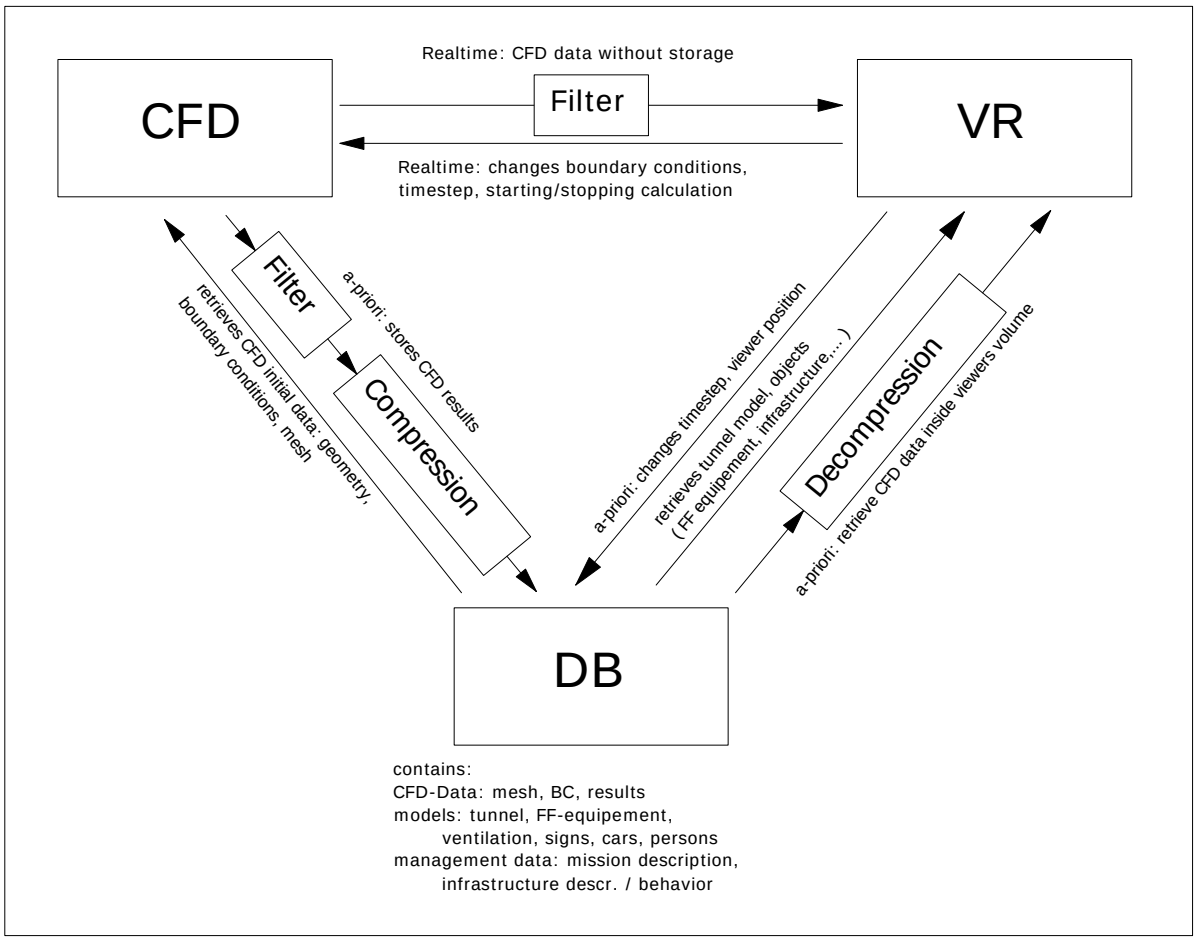


Figure 14: interaction between CFD, VR and DB

Figure 14 shows the interaction and dataflow between the main modules. Depending on the following listed usecases there are several paths possible.

2.2.3.3.1 Generation of a new simulation

For a new simulation the user stores the 3D model of the tunnel and all of its infrastructure, the firefighting equipment models and descriptions into the database. From this 3D model a suitable model with a reduced level of detail for the CFD-calculation will be generated and stored into the DB. This file will be (manually) processed with a meshing tool to get a useable mesh. Also the initial boundary conditions are defined within it. The resulting files build the base for further CFD-calculations and are stored back into the DB.

2.2.3.3.2 Realtime simulation

Initialisation

The VR System loads the model from the database and establishes a direct connection to the CFD-calculation. The CFD solver retrieves its initial dataset (computational domain and boundary conditions) from the database. The RT solver will construct its mesh at the beginning of the calculation.

Running the simulation

While in the interactive simulation mode the boundary conditions for the CFD-calculation will be changed by user interaction like activation of ventilation systems or fire fighting equipement. These changes are forwarded directly to the CFD-solver and contribute to the results for the next timestep. To reduce the amount of data to be transferred the results of the CFD calculation are filtered before being transmitted back to the VR System. Due to the huge amount of data being generated by the CFD it seems not useful to store these data into the database for later usage.

2.2.3.3.3 A-priori simulation

Precalculation

The boundary conditions for the whole simulation period are predefined and stored into the database. The CFD solver performs its calculation based on this predefined data. The resulting dataset is being filtered, compressed and finally stored into the database.

Inspecting the results in the VR system

The VR system fetches the model of the tunnel for an initial display. A part of the CFD-data belonging to a volume around the current viewing position of the user from the simulation is fetched from the database and rendered.

2.2.3.4 Content

The content of the database will be divided into the following main parts.

2.2.3.4.1 CFD-Data

The required geometrical and topological description for the CFD calculation. From this data a suitable mesh will be generated by the preprocessor of the CFD solver if needed. The initial boundary conditions and the results of a-priori calculations.

2.2.3.4.2 models

Data required by the VR system. It incorporates the photorealistic models of the tunnel and its infrastructure, the firefighting equipment, cars and persons.

2.2.3.4.3 management data

Data that is needed for configuring the simulator. This contains the description of fire fighting missions, of the behaviour of infrastructure and its physical parameters for the simulation.

2.2.3.5 Interfaces

As decided in WP 2.3 all parts of communication with the database will be provided as Corba interfaces

2.2.4 KTH

2.2.4.1 Use scenarios

Three different use scenarios can be identified for the VIRTUALFIRES system:

1. Planning authorities need to assess the likely outcome of a fire in a planned or existing tunnel. They may need to demonstrate the results of this assessment to laymen.

2. Fire departments need to simulate the effects of taking given actions in a fire situation.

3. Fire fighters and tunnel operators need to practice their behaviour in the case of a fire.

We now attempt to discern what requirements the different scenarios place on the software.

Scenario 1

Target group:

Planners, can be expected to be familiar with visualisation software. Most likely do not have access to VR equipment.

Assumptions:

Need to be able to see:

- Development of fire
- Spread of smoke over time
- Spread of toxic gasses over time
- Gas flow
- Visibility of emergency (possibly lit) signs

Functionality:

- Import of geometry dataset
- Choice of camera position
- Turning on and off display of
    - smoke (Sign visibility should be implicit through the smoke display.)
    - gas
    - gas velocity

    - temperature
    - ventilation settings

- Adjusting iso-surface levels
- Stepping back and forth through time

At a later stage:

- Placement of fire
- Placement of cars, lorries, trains in tunnel
- Setting properties of vehicles (loads of lorries, etc)
- Adjusting the ventilation settings during the course of a simulation
- Setting weather conditions at tunnel openings (including ventilation shafts!)
- Stopping the simulation, changing parameters, restarting

Scenario 2

Target group:

Fire fighters, presumed familiar with visualisation equipment.

Assumptions:

Need to be able to see:

- The effects of extinction efforts
- The effects of fans brought into tunnel in addition to the points under scenario 1 (with higher priority to temperature and smoke)

Scenario 3

Target group:

Fire fighters, need not be familiar with computer equipment. In addition to fire fighters, tunnel operating staff may also be involved to the extent that they can control fixed equipment, such as ventilation, doors, etc.

Assumptions:

This is the actual training scenario. It can be developed at several levels, but currently it is considered impractical to do a full simulation of a fire situation and some compromises in realism are acceptable. (Based on discussions with FDDO.)

It seems important to practice cooperation between fire fighters, and presumably between fire fighters and tunnel operators. However, discussions with FDDO and ALPETUNNEL

indicate that coordination between remotes sites are not a high priority, therefore we can assume all users to be located in the same building.

For training, the simulation does not have to be scientifically correct, real-time performance being more important.

Two extreme cases of training method can be envisioned:

1. A completely desktop-based interface where each fire fighter uses a mouse-based interface in order to manipulate various fire extinguishing equipment, as in any computer game.

2. A semi-realistic simulation where suited-up fire fighters move more or less as they would on a real fire site.

Comments from the Fire Department in Graz suggest that an important part of training is getting used to the physical stress of heat, equipment weight and oxygen breathing. These can be simulated to some extent, though it is unclear if this particular application will *require* it; in any case the interface should not a priori preclude the use of (instrumented) protection suits, should these later on be deemed to be important.

The visual representation is interesting - an initially clear tunnel can rapidly be enveloped in impenetrable smoke, and this rapid transition is important to demonstrate to the trainees. The subsequent lack of vision would indicate the need for haptic interfaces, but these would be beyond the scope of the project to create.

Functionality:

- Import a geometry dataset

- Record a training session and replay it from different viewpoints

- Allow a trainer to place vehicles and other obstacles, and place fires.

- Fires can allowed to develop from stereotyped models, or from actual simulations, if these can be run at sufficient speed.

- Break a session, change parameters and restart.

- Detect if firefighters are subject to dangerous temperatures, smoke or gasses.

Suggested interface:

Case 1:

The trainees sit by ordinary workstations. A maximally simple interface should be used where the trainees use a point-and-click interface to navigate their avatars, choose tools and apply the tools to the environment (e g, indicate where to direct a hose).

Case 2:

For multiple persons training, a CAVE environment is non-optimal, as it would be unlikely to afford sufficient physical separation of trainees. An HMD interface can be extended to

an arbitrary number of participants, given constraints of inter process communication delays and physical space for training. On the other hand an HMD may conflict with the wearing of smoke-helmets - a market study to determine whether an HMD with suitable profile exists should be undertaken.

For each trainee, their head has to be tracked as well as a prop, corresponding to the hose or other tool in use. While fire fighters do not move very long distances, they still tax the limits of available tracking systems, so some kind of virtual motion has to be indicated. A method requiring a minimum of hardware is walking motion detection (virtual treadmill) [Slater et al, 1994].

If the trainees are shouting to each other, there might be a problem with the physical positions of the trainees not corresponding to the virtual ones. In a situation where the trainees can see each other in the virtual environment, visual capture would tend to alleviate this, but if the trainees cannot see each other directional audio cues may in fact be important. In this case speech and other sounds from each trainee may have to be picked up, spatialised and retransmitted to the other trainees.

Electromagnetic trackers can handle a large number (> 30) of receivers, but not over a large volume, so it may be necessary to use some kind of optical tracking system.

In both cases, the trainer should be able to work from a desktop workstation in order to set the scenario, monitor the progress of the trainees and stop, adjust and restart a scenario when needed.

On the assumption that tunnel operators already work at a desktop environment, a copy of this should be possible to use for their interaction with the environment.

<table>
<tr><th>Testing</th><th>Planning</th><th>Demonstration</th><th>Training</th></tr>
<tr><td>Fire authorities
Tunnel operators</td><td>Builders
Fire authorities
Tunnel operators</td><td>Financers</td><td>Fire fighters</td></tr>
<tr><td colspan="3">Correct physics</td><td>Qualitatively correct</td></tr>
<tr><td colspan="2">Smoke and fire</td><td colspan="2">Smoke and fire</td></tr>
<tr><td colspan="2">Sci Viz methods</td><td></td><td></td></tr>
<tr><td colspan="3">CAVE presentation</td><td></td></tr>
<tr><td></td><td></td><td colspan="2">HMD presentation</td></tr>
<tr><td colspan="4">Worskstation presentation</td></tr>
</table>

Figure 15: overview of the different settings and user groups

2.2.4.2 User interface

While the software is intended to be used on multiple display systems, CAVEs, HMDs and standard monitors, it seems unlikely that the identical interface can be used in all instances. While several of the relevant visualisation packages allow similar types of button-and-menu interfaces to be used both on the desktop and in an immersive environment, the latter alternatives are often sub-optimal, especially in the case of a HMD-based solution.

For the desktop version, standard graphical user interfaces will work well, but would be enhanced by support for stereo goggles and/or SpaceMouse or similar device, as these will simplify the interpretation and manipulation of data.

For an immersive interface the ideal would be a speech- and gesture-based interface, which allows generating visualisations by pointing at positions and speaking operations, rather than requiring floating menues and buttons for operation. A project currently underway at KTH aims to make it possible to use a PDA for interaction in a CAVE environment, which would then make it possible to have most of the benefits of a 2D interface (though limited to a much smaller screen) whilst still being able to work in an immersive space. In particular this will make it easy to set iso-surface levels, fire load settings and other such items which require the input of numerical values.

A nice feature, used in e g raypaint (http://www-graphics.stanford.edu/~cek/rayshade/), is to perform approximate computations and display these at a coarse resolution, and then let the user "rub" or otherwise indicate an interesting region of space to force a more precise computation to be performed for that region. (This assuming that the simulation algorithms can manage such a region-of-interest-based refinement.)

The models of the tunnel contain moving parts (doors, vents, etc) that affect the progress of a simulation and they must therefore be movable by the user. The data format must somehow indicate if objects are movable, as well as the (at least the geometric) constraints for this motion, and the user interface must be able to handle this information. At the least these objects can be defined at their extreme positions (fully open, fully closed), and the interface be able to select between these alternatives. Which position is selected must then be reported back to the simulation, possibly along with parameters indicating what the effect of the one or other setting is.

Likewise, it must be possible to set weather parameters for all openings of a tunnel - for multiple vent openings it must be possible to just apply a standard set. Conceivably the metaphor should be a fill-in form with all relevant parameters of which copies can be dropped by the openings.

Placement of vehicles and fire sources in the tunnel must be interactive. One can equip the users with a palette of vehicles that can be inserted into the environment. The contents of this palette should preferably be stored in the database described before, so that they are properly integrated with the computational modules and can be accessed by any interface developed. The description of the objects should then also contain iconic representations of the objects, as well as 3D models that can be inserted into the environment. It would be useful to be able to create new items to be stored in the database, perhaps based on already existing objects, but this can be left for later. The actual placement should use collision detection so that vehicles do not penetrate each other or the walls of the tunnel. In a desktop interface this would also help in placing objects in 3D space, as they could be slid along the floor or walls of the tunnel. However, the extremely high polygon count of the tunnel models will then require a highly optimised collision detection method which can quickly select a likely subset of polygons to test for collision. In an immersive interface on the other hand, collision detection need not necessarily operate in real time, but can instead be invoked by an explicit command or run in a lower-priority thread.

Data have to be recorded over time so that the progress of a given session can be replayed at a later time without having to redo the computation (particularly relevant for sites that

do not have supercomputer resources, but still need to see the results of simulations). There should be a convenient way to indicate and annotate breakpoints at any point in a time flow and allow the user to explore alternatives, e g insert a fan at a given time, follow the development from that time on, go back and run the simulation from the same point in time but without the fan and then be able to compare the results of the two cases. It seems reasonable to use a tree-branching display for the timeline. On a desktop display this is trivial to render and interact with. In an immersive display there must be some way of separating the time display from the rest of the environment; possibly one could adjust the hidden-surface removal algorithm in such a way that certain objects, such as interface widgets, will always be rendered on top and not be obscured by the data displays. (How this will affect depth perception remains to be seen, but empirically it tends to be quite robust to similar anomalies in depth, such as those caused by a user's body parts obscuring virtual objects that actually are visually closer.) A timeline can profitably be rendered in three dimensions rather than as a flat object, allowing rotation of branches into the third dimension - which *may* help alleviate clutter. "Gravity"-based interaction, where active regions of space will attract the interaction point, will hopefully simplify for users to access the intended parts of the timeline. The aforementioned PDA-based interface will allow the writing of annotations, but an HMD user is in practice required to have a non-immersed assistant who can type for her, unless we use audio annotations, but these are difficult to overview at a glance (of course, in an HMD text cannot in general be read anyway). In addition to written and/or spoken annotations, small thumbnails of the visualisation at the given points of time may help searching for a specific setting of parameters.

It is unclear if there should be any specific support for the creation of paths through a visualization, so that a high-quality offline rendering can be made, or possibly just to replay a particular scenario for an audience. There are standard methods for setting waypoints and traveling along them, though it has to be said that they are not always very convenient to use.

2.2.5 Final Plan

The global aim of the project will be to develop a tool which allows the user to interact with the simulation (as fine/fast as possible) like it was specified in the proposal. After the WP 2 meeting held in Stockholm on May 12, the project partners decided to realize the system in the following specification.

2.2.5.1 Several prototypes

Several prototypes should be realized to approach the final product in three main steps. The first prototype should be capable to simulate and display a tunnel with fixed geometry and a fire load in it. The second one should be able to include fixed fire fighting measurements. In the third one the fire fighting measurements should be moveable.

2.2.5.2 Description of tunnel & geometry

The tunnel geometry & texture should be stored in a standard 3D format (like *.obj for example). The meta information will be saved in an extra file, which is written in XML language. It describes the "special" objects, which can be placed in the tunnel and have a direct or indirect influence on the fire (fire fighting measurements, burnable objects etc.). The different geometry and texture states ("fresh", "burned", etc.) of these "special" objects will be stored, according to the tunnel information, in (several) 3D files (*.obj) for each state. The whole scene information (which tunnel geometry, which meta objects at which position) will be stored in another XML file ("scene" file). In the first run it will perhaps be tried to safe the placement of the "special" objects in the scene-geometry itself using the tunnel geometry file by placing simple 3D Objects (like cubes for example) with special tag names as place. It will be considered if it is useful to store the whole information of the scene including meta objects in one file, in later versions.

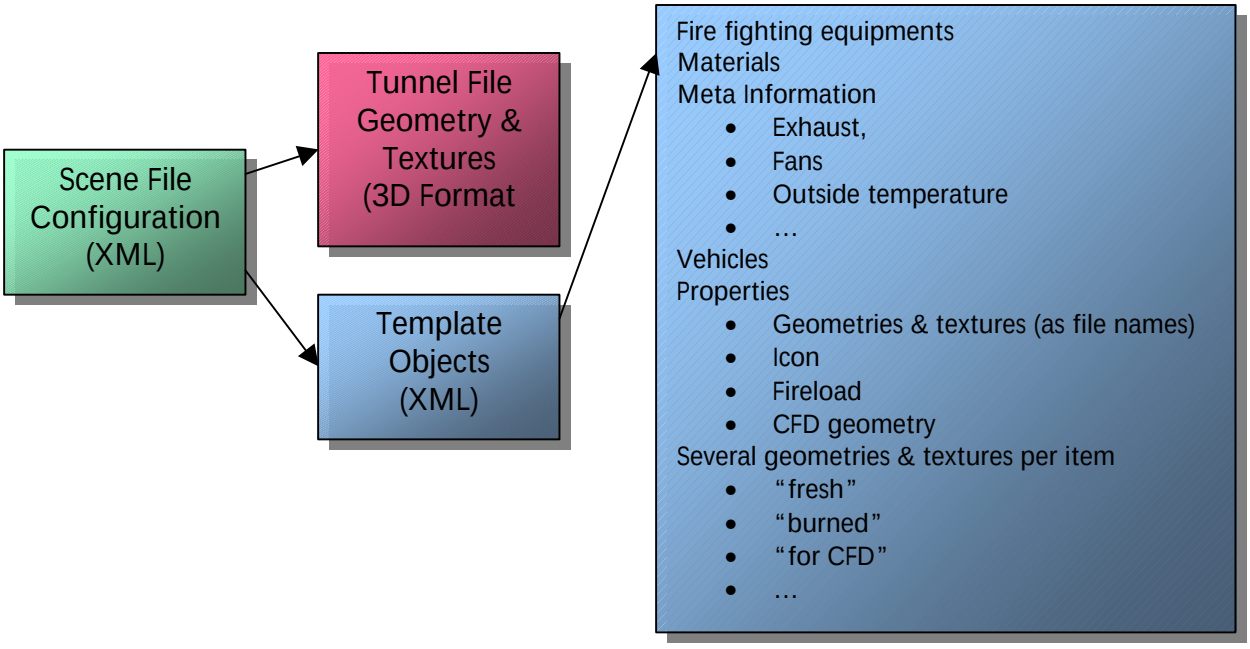


Figure 15: File Layout

As can be seen in figure 15 the scene can be described using several XML and 3D format files. "Special" resp. "template" objects carrying meta data can be things like the tunnel entrance, air pressure at certain positions or fire loadable items (ticket machine, car, train, …) and cars for example.

The perspective to display the scene will be a 3rd person perspective which can be zoomed & rotated in a 3D environment to a 1st person perspective easily. The decision which state each special item has ("burned", "fresh", etc.) will be made by the simulation part of the software during the simulation.

2.2.5.3 Simulation

At the current state of the development the simulation part is able to process about 100.000 cells which corresponds to about 10m³ in the desired precision per frame in the real-time mode. Unfortunately this is not even enough to include just a simple truck in the calculations. Therefore it has to be considered if the planned data rate of about 25 simulation frames per second can be reduced to a more reasonable value. By keeping this amount of frames and grid points per second, at least the network traffic rate will reach a critical value. The first prototypes will help to investigate this problem.

As the end users suggested, the interactive "rougher" simulation can be used in the final application to find the interesting parts of the tunnel and the important simulation time intervals. Afterwards the precise (offline) simulation can be started using these values and its results will then be presented as post visualization.

2.2.5.4 User interfaces

Several user interfaces will be needed in the final product. The real-time simulation and visualization needs a GUI which can be displayed in the CAVE and which has direct influence on the underlying simulation. Furthermore an interface for the simulation itself is needed (at least for the prototypes) to put in the values, scenes and template objects as specified in 2.3.2. For example the users should be able to put out fans, modify smoke distribution rates, velocities, temperatures, smoke concentrations etc.. Furthermore he should be enabled to change ventilation and adjust the parameters of the fixed fire fighting measurements. In the end of the project it would be nice to have the user being able to modify / place mobile fire fighting measurements in the virtual scene and modify them as well.

Additionally a user interface for the visualization itself is needed (turn on/off several visualization methods and adjust their parameters) – see 2.3.6.

2.2.5.5 Data storage & communication

For a performant visualization a resampling of the simulation output data is needed. At the present point of the development it seems clear that the visualization will make use of regular grids (equidistant or octree for example). Therefore the software needs to decouple the grids of the simulation and of the visualization. The best place for this computation will be the database (~interface). By doing so the simulation will still be able to safe its whole simulation output into the database and on the other hand the visualization will be able to request just the information which is really needed to display the scene as exact as needed. This way bandwidth and computing power can be saved on the visualization side to guarantee a performant rendering of the result. For this reason the (space) interpolation which is needed for resampling will be done in the database.

Time interpolation will not be needed. The regular case will be a real-time application or even an accelerated display of the simulation results. Therefore it will only be necessary to have the possibility to drop frames.

Two abstract interfaces have to be designed around the database itself. One between the simulation and the data storage and the other one between the data storage and the visualization.

In the first shot all the interfaces will be realized using Corba.

2.2.5.6 Visualization

At the end the visualization should be able to display smoke, temperatures, velocities and toxic gases (which are normally invisible) using several special realistic and scientific visualization methods. The concurrent visualization will have an interface so that the user is able to have a direct influence on the simulation parameters (e.g. turn on fans, place fire fighting equipment, etc.). For the post visualization only an interface to modify the parameters for the visualization methods themselves is needed.