



Agenzia nazionale per le nuove tecnologie, l'energia
e lo sviluppo economico sostenibile



Ministero dello Sviluppo Economico

RICERCA DI SISTEMA ELETTRICO

Optimization and validation of the CFD modules in the Nurisp Platform

*F. Bassenghi, G. Bornia, S. Manservigi, R. Scardovelli, F. Donato, C. Lombardo, M.
Polidori*



OPTIMIZATION AND VALIDATION OF THE CFD MODULES IN THE NURISP PLATFORM

F. Bassenghi, G. Borna, S. Manservizi, R. Scardovelli – CIRTEN Università di Bologna, F. Donato, C. Lombardo, M. Polidori - ENEA

Settembre 2012

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico - ENEA

Area: Governo, gestione e sviluppo del sistema elettrico nazionale

Progetto: Nuovo nucleare da fissione: collaborazioni internazionali e sviluppo competenze in materia nucleare

Responsabile del Progetto: Paride Meloni, ENEA

Titolo
**OPTIMIZATION AND VALIDATION OF THE
CFD MODULES IN THE NURISP PLATFORM**
Descrittori
Tipologia del documento: **Rapporto Tecnico**
Collocazione contrattuale: Accordo di Programma ENEA-MSE: tema di ricerca "Nuovo nucleare da fissione"

Argomenti trattati: Termoidraulica dei reattori nucleari

Analisi di sicurezza

Calcolo parallelo

Sommario

In questo rapporto sono presentati lo stato di avanzamento nell'integrazione della piattaforma NURISP sul sistema di calcolo CRESCO dell'ENEA e la validazione di alcuni codici della piattaforma.

Sono in particolare discusse le criticità emerse riguardo la compatibilità delle librerie di input/output necessarie all'integrazione dei vari codici. Si dimostra come l'integrazione di codici di tipo open-source all'interno della piattaforma SALOME sia effettivamente ottenibile.

Per quanto riguarda l'attività di validazione vengono presentati i risultati delle seguenti attività:

- validazione del codice CATHARE sull'impianto sperimentale SPES-99;
- simulazione congiunta CATHARE/NEPTUNE_CFD dell'impianto PERSEO;
- validazione del codice TRIO_U su correlazioni empiriche e simulazioni Lattice-Boltzman per test di ebollizione da parete riscaldata.


Note

Questo documento è stato preparato col contributo congiunto del seguente personale di ricerca ENEA e CIRTEN

- F. Bassenghi, G. Bornia, S. Manservigi, R. Scardovelli (CIRTEN - Università di Bologna DIENCA)
Sigla doc. rif.: CIRTEN - Università di Bologna: CERSE RdS/2012/1352-UNIBO
- F. Donato, C. Lombardo, M. Polidori (ENEA)

Copia n. In carico a:

2			NOME			
			FIRMA			
1			NOME			
			FIRMA			
0	EMISSIONE		NOME	F. Donato	P. Meloni	P. Meloni
			FIRMA			
REV.	DESCRIZIONE	DATA		REDAZIONE	CONVALIDA	APPROVAZIONE


Contents

Abstract	5
CRESCO-ENEA GRID platform for thermal-hydraulics simulations	8
1.1 Platform introduction	8
1.2 Library compatibility	10
1.2.1 Data exchange over the platform and MED libraries	11
1.2.2 GUIs over the platform and QT libraries	12
1.2.3 Parallel computations over the platform and MPI libraries	14
CATHARE model validation with the SPES-99 facility	15
2.1 Introduction	15
2.1.1 CATHARE model for the SPES-99 facility	16
2.1.2 Regulated Steady State	17
2.1.3 Reference Steady State	19
2.2 SPES-99 IB LOCA Transient	20
2.2.1 Experimental data	20
2.2.2 Comparison between post-test results and experimental data	22
Validation of the CATHARE and NEPTUNE models for the PERSEO facility	27
3.2 The PERSEO facility test	27
3.2.1 The PERSEO facility	27
3.2.2 PERSEO experimental Test 9	30
3.3 CATHARE-NEPTUNE coupled simulation	35
3.3.1 CATHARE solution of the PERSEO facility	35
3.3.2 Boundary conditions for NEPTUNE	37
3.4 NEPTUNE 3D simulation set up on CRESCO	38
3.4.1 Code setup	38
3.4.2 Initial conditions, boundary conditions and physical proper- ties files	41
3.4.3 Experimental data	43
3.4.4 Results	45

Results obtained on the real geometry	46
Results obtained on the geometry with a straight injector and the nozzle at the real position	49
Results obtained on the geometry with a straight injector in the center of the domain and an additional volume simulating a portion of the HX pool	53
3.4.5 Computational costs	60
Validation of the TRIO_U code with heterogeneous nucleate boiling from a single site	61
4.1 Introduction	61
4.1.1 Experimental data	61
4.1.2 The governing equations	64
4.2 Numerical results	67
4.2.1 Set up of the TRIO_U code	67
4.2.2 Bubble release diameter for heterogeneous nucleate boiling from a single site	71
4.2.3 Bubble release frequency for heterogeneous nucleate boiling from a single site	73
Bubble release frequency as a function of gravity	75
Bubble release frequency as a function of surface tension	79
Bubble release frequency as a function of the contact angle	82
Integration of codes into the SALOME platform	85
5.1 SALOME GUI integration	85
5.1.1 Introduction	85
5.1.2 GUI integration with open-source codes: the SATURNE case	91
5.1.3 GUI integration with NURISP codes: the NEPTUNE and TRIO_U cases	96
5.1.4 GUI integration with in-house codes: the FEMuS case	98
5.1.5 GUI integration with PARAVIEW	101
5.2 Pre-processing and post-processing integration	101
5.2.1 Integration with open-source codes: the SATURNE case	101
SALOME-GEOM application	101
SALOME-MESH application	103
SALOME-SATURNE application	103
SALOME-POST-PRO and SALOME-PARAVIEW applications	109
5.2.2 Integration with NURISP codes: the NEPTUNE and TRIO_U cases	110
5.2.3 Integration with in-house codes: the FEMuS case	110
5.3 Integration with full coupling	112
Conclusions	114


ENEA	Ricerca Sistema Elettrico	Sigla di identificazione NNFISS-LP2-089	Rev. 0	Distrib. L	Pag. 4	di 121
-------------	----------------------------------	--	-----------	---------------	-----------	-----------

References	116
Notes on the Working Group of the University of Bologna	121

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	5	121

ABSTRACT

The NURISP European project is based on a set of codes that have been developed to simulate the thermal-hydraulic behavior of nuclear power plants and of single components. Each code should be a module inside a larger frame based on the open-source platform SALOME. The platform should be able to upload and download different modules in order to make feasible the study of multiphysics and multiscale problems. Different modules may be used to investigate physical situations at different scales ranging from the boiling cycle on a single nucleation site to the analysis and monitoring of a nuclear power plant. In this work we discuss first the status of integration of different modules inside the SALOME platform, with particular attention to GUI and MPI libraries. Then we focus on the three codes TRIO_U, NEPTUNE and CATHARE and use these modules to validate previous studies and experiments. CATHARE is a lumped-parameter thermal-hydraulic code to investigate the time dependent behavior of complex systems and it is used here to analyze the experimental data from the SPES-99 facility. TRIO_U is a three-dimensional code for two-phase flows, which is based on the single-fluid formulation of the Navier-Stokes equations, and it is here validated against some experimental data and standard correlations involving the release time of single bubbles detaching from horizontal heated walls. Finally, NEPTUNE is a multidimensional code for thermal-hydraulics problems, which is based on the two-fluid formulation of the conservation equations, and it is here used to analyze the experimental data from the PERSEO facility. In the final part of this report we discuss the integration of these and other modules inside the SALOME platform using the open-source code SATURNE as a template.


	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	6	121

Introduction

In recent years ENEA has joined the user group of the European NURISP project funded by the European Union (EU). The goal of this project is to develop a software platform that gathers a wide variety of codes for the simulation of nuclear reactor systems. The members of the user group of the NURISP project do not have access to the source codes of the platform, but they can use these programs under strict agreement in order to validate them against experimental data. The development of the NURISP platform reflects the European policy to create a common set of computational tools to investigate the design of the thermal-hydraulics of nuclear reactors in the presence of a great variety of different physical phenomena occurring at different spatial scales. In particular, the three codes CATHARE, NEPTUNE and TRIO_U have been developed by Électricité De France (EDF) and Commissariat à l'Énergie Atomique et aux Énergies Alternatives (CEA), and are not open-source codes. As a member of the user group of the NURISP project, ENEA has received an executable version of these three codes, but not with a complete functionality. Based on the European NURISP platform model, these codes are modules of a software platform which has been implemented on the CRESCO-ENEA grid system [2, 3].

Moreover, in more recent years some other models, which have been developed in the nuclear field once again by EDF and CEA, have been released as open-source codes and can now be accessed by all developers in a joint effort to improve their usability and the accuracy in modeling light water nuclear reactors (LWR). These new codes can be added to the software platform, improving in this way the possibility to study the very complex behavior of nuclear reactor systems. Indeed, only a large international effort can be effective in investigating very complex multi-physics and multi-scale phenomena which arise in the design and analysis of nuclear reactor systems. Because of these features, these phenomena cannot be studied in a satisfactory way with a single model, but the coupling between different codes is required to achieve accurate results. At present SALOME is an open-source platform and the two codes SATURNE and SYRTHES have also been released by EDF itself with an open-source license. Other visualization tools, such as PARAVIEW and VisIt, and several mesh generation applications are already open-source and can be downloaded from the network with their own license agreement.

This document reports the efforts in the implementation, development and validation of the computational platform on the CRESCO-ENEA GRID with the purpose of studying the thermal-hydraulic behavior of LWR innovative nuclear reactors. The University of Bologna participates in the effort of installing, using, and validating these computational tools and other in-house codes on the CRESCO-ENEA GRID cluster located in Portici (near Naples). This platform has been conceived not only to collect a series of codes that have been extensively used in this field, but also to harmonize them with the aim of solving complex multi-scale problems by exchanging information among different codes over a common platform and on

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	7	121

large multiprocessor architectures.


In Chapter 1 we briefly recall the NURISP platform structure. Library compatibility issues are then discussed, which are of fundamental importance for an effective integration of the various codes. Particular attention is given to the libraries for the input and output of data and their exchange between codes, to the implementation of Graphical User Interfaces (GUI) and to parallel computing.

In Chapter 2 a model based on the system code CATHARE for the simulation of the thermal-hydraulic behavior of the SPES-99 facility is presented. The geometry of the facility is described in detail adopting rules and approaches recommended by the code developers. First, the reference steady-state conditions are attained with the code by means of a regulation algorithm of the main parameters that reproduces the real control loops as accurately as possible. In this way a number of physical parameters are appropriately calibrated and the transient behavior of the SPES-99 facility for an intermediate break loss of coolant accident (IBLOCA) can be simulated. The results can then be validated against the experimental measurements.

Chapter 3 presents the results obtained in the simulation of an experimental test at the PERSEO facility. The simulation is performed by a coupling of the CATHARE and NEPTUNE codes. The facility is first discretized with the one-dimensional (1D) code CATHARE and then the results of the test simulation are used as boundary conditions for the simulation with the NEPTUNE CFD code of a single component of the facility. This component is simulated in greater detail in three-dimensional (3D) geometry and is given by the overall pool and the injector of the PERSEO facility. For the comparison with the experimental data, we consider the discretization of the real 3D geometry and of two simplified geometries.

In Chapter 4 we consider a few validation tests of the TRIO_U code by analyzing the formation, growth and detachment of bubbles on a single site in heterogeneous nucleate boiling. The bubble cycle is studied in terms of the bubble diameter at detachment and the bubble release frequency as a function of gravity, surface tension and contact angle.

Chapter 5 describes the status of integration of the NURISP codes within the SALOME platform. In particular we discuss the embedding of the GUI of each code within the SALOME GUI framework, by considering open-source, closed-source and in-house codes separately. Clearly, the integration is highly facilitated in the case of open-source GUI. We also deal with the harmonization of the codes in the pre- and post-processing stages, with the purpose of improving the data communication between codes and consistent input/output formats.

 Ricerca Sistema Elettrico	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	8	121

CRESCO-ENEA GRID platform for thermal-hydraulics simulations

1.1 Platform introduction

The ENEA platform for thermal-hydraulics has been implemented on CRESCO (Centro Computazionale di RicErca sui Sistemi Complessi, Computational Research Center for Complex Systems) [1], which is located at the ENEA Center in Portici near Naples and consists of a High Performance Computing infrastructure mainly devoted to the study of Complex Systems. The access to the platform can be done through a number of front-end machines (for example `crescof01.frascati@enea.it`) in four different ways: SSH, Citrix, FARO, and NX client. The easiest way is to use a ssh client with a terminal interface which can directly access one of the front-end machines. The access is limited to authorized users with a username and a password that can be used on every machine of the grid. A brief description of CRESCO machines and the platform can be found in [2, 3].

The CRESCO-ENEA GRID system hosts several codes for the simulation of nuclear components and systems, which are the main modules of the FISSICU platform. Some of these codes are taken from the European NURISP project. The FISSICU project is located in the directory

```
/afs/enea.it/project/fissicu
```

The directory `fissicu` consists of three main sub-directories: `data`, where the data and results of the simulations should be stored, `html`, where the web pages and other helping tools are located, and `soft`, where all the codes are installed. Inside the `soft` directory there are several applications, a few of them are listed in Table 1.1. SALOME provides a common platform for the pre-processing and post-processing of the numerical simulations [9]. It is based on an open and flexible architecture made up of several components. This platform has been conceived not only to collect a series of codes that have been extensively used in the field of thermal-hydraulics of nuclear reactors but also to harmonize them with the aim of solving complex problems by exchanging information among different codes within a common environment and on large multiprocessor architectures. As shown in Table 1.1, the platform contains modules for pre-processing, like the CAD application GEOM, and the mesh


code	type	application
SALOME	open-source	platform
SALOME-GEOM	open-source	CAD
SALOME-MESH	open-source	mesh generator
SALOME-POST-PRO	open-source	data analysis
SALOME-PARAVIS	open-source	data analysis
CATHARE	executable	1D, multiphase thermal-hydraulics
NEPTUNE	executable	2D/3D, multiphase CFD
TRIO_U	executable	3D, two-phase DNS
SATURNE	open-source	2D/3D, single-phase CFD
SYRTHES	open-source	2D/3D, heat conduction and radiation
FEMuS	open-source	2D/3D, two-phase CFD and mechanics

Table 1.1: List of platform codes

generator MESH. With the GEOM application it is possible to create, modify, import, export (IGES, STEP, BREP), repair and fix CAD models, and with the MESH application to edit, check, import and export meshes (in MED, UNV, DAT and STL formats). We refer to them as SALOME-GEOM and SALOME-MESH, since they are fully integrated in the SALOME platform. There are also post-processing tools to analyze the simulation results, like POST-PRO and PARAVIEW. PARAVIEW is an open-source, multi-platform application for data analysis and visualization. The data analysis in 3D can be done interactively or by using its batch processing capabilities [10]. The SALOME-POST-PRO and SALOME-PARAVIS applications are their integrated version inside SALOME.

SALOME can also be used as a platform for the integration of external third-party numerical codes to produce a new application with full pre- and post-processing management of CAD models. The European project for the NURISP platform is planned to be based on the SALOME platform. At the moment the three codes CATHARE, NEPTUNE and TRIO_U codes are independent from the platform, with not fully-compatible input and output formats. These codes have different purposes and licenses as reported in Table 1.1. CATHARE is a Code for the Analysis of Thermal-hydraulics during an Accident of Reactor and safety Evaluation for LWR [11]. It is a system code which is used mainly for PWR safety analysis, accident management and definition of plant operating procedures. It is also used to quantify conservative analysis margins and for nuclear reactor licensing [12, 13]. TRIO_U and NEPTUNE solve the conservation equations in multidimensional geometry [15, 16, 19, 20]. They both consider two-phase flows, but the first code is based on the single-fluid formulation of the evolution equations, the second code on the two-fluid model equations. CATHARE, NEPTUNE and TRIO_U are developed by CEA and EDF and they can be used only under NURISP project agreement.

In recent years some other codes, such as SATURNE and SYRTHES, both de-

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	10	121

veloped by EDF and CEA mainly for nuclear applications, have become open-source and can be accessed by any user in a joint effort to improve the usability and the accuracy in modeling LWR nuclear reactors. SATURNE is a general purpose CFD software. It is based on a collocated Finite Volume approach that accepts meshes with any type of cell. The code works with finite volumes and any type of grid structures such as unstructured or with hanging node geometries. It has the remarkable feature to simulate both incompressible and compressible single-phase flows with several turbulence models [29, 30]. SYRTHES is a finite element code that can be used to handle different kinds of problems such as heat conduction and radiation in solids [31]. These two codes can be coupled to solve conjugate heat transfer problems. SATURNE can also be used together with the structural analysis software CODE_ASTER, in particular within the SALOME platform. CODE_ASTER is also developed by EDF and is distributed under the GNU GPL license. All these codes can be added to the platform to improve the possibility to study the very complex behavior of nuclear reactor systems. The various codes can be used independently to analyze the reactor from different points of view, but a strong coupling among them is necessary to achieve more accurate results.

We are also interested in the integration inside the FISSICU platform of some in-house codes in order to solve particular problems. This is the case for the FEMuS code (Finite Element Multiphysics Solver), which is a finite element code developed at the Laboratory of Montecuccolino of the University of Bologna for the study of thermal-hydraulics, neutronics and structural mechanics problems in both 2D and 3D.

For each code of the platform, a script (`executable_env`) is available in the directory

```
/afs/enea.it/project/fissicu/soft/bin
```

which simply sets the proper environment variables to execute the code. This procedure can be used to run the code directly without any graphical interface on the single node where the user is logged in. This directory should be added to the user environment variable `PATH`. In the `bash` shell the command line is

```
$ export PATH=/afs/enea.it/project/fissicu/soft/bin:$PATH
```

This command line can also be added to the user configuration file `.bashrc`. CRESCO supports the LSF (Load Sharing Facility) job scheduler, which is a suite of several components to manage a large cluster of computers with different architectures and operating systems.

1.2 Library compatibility

In order to harmonize the input and output of all these different codes, a unique library and format should handle mesh input and all data operations. A code is

fully integrated when its GUI, data exchange and MPI computations are performed with the same rules and libraries inside a common framework. Common GUIs allow the user to perform all actions in a single environment. Parallel computations are effective if in the platform a unique MPI library and the same domain decomposition techniques are used. However, a complete integration can only be achieved when the input and output of the different models are managed by the same library with the same formats.

1.2.1 Data exchange over the platform and MED libraries


The main benefit of a common exchange format is to share computations of nodal connectivity over subregions and arithmetic operations over field solutions for interpolation and coarsening. A common exchange format also reduces the complexity of possible code coupling. The European NURISP project has planned to set the

code	MED compatibility	input MED	output MED
SALOME	yes	yes	yes
SALOME-GEOM	yes	yes	yes
SALOME-MESH	yes	yes	yes
SALOME-PARAVIS	yes	yes	
SALOME-POST-PRO	yes	yes	
CATHARE	no	no	no
NEPTUNE	partial	partial	no
TRIO_U	partial	partial	no
SATURNE	yes	yes	yes
SYRTHES	yes	yes	yes
FEMuS	partial	yes	no
PARAVIEW	no	no	no

Table 1.2: MED compatibility format

SALOME platform at the center of this integration. All data operations in the SALOME platform involve files in MED (Modèle d’Echange de Données) format. MED is a data storage format based on the HDF library [6]. The HDF format is largely used and will probably become the standard in the near future. However, the implementation in MED format is rather unusual. In the SALOME platform a MED module is implemented. The MED module provides a library for storing and recovering computer data in MED format associated to meshes and physical fields, which facilitates the exchange between different codes and numerical solvers. Inside SALOME these data structures are exchanged at the communication level (CORBA or MPI) offering common read/write through MED files.

The MED data model defines the data structures exchanged by codes. The modeled data are geometrical data and the physical fields that are usually defined on

 ENEA Ricerca Sistema Elettrico	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	12	121

the nodes of the mesh. MED supports eight element shapes: point, line, triangle, quadrangle, tetrahedron, pyramid, hexahedron, polygon and polyhedron. Each element has a different number of nodes, depending on the interpolations that can be linear or quadratic. MED defines also numbering conventions, hence a standard for data storage in HDF5 format.

The MED libraries can support three groups of functionalities: MED File, MED Memory and MED CORBA. The first group (MED File) is a C and FORTRAN API that implements mesh and read/write functions into files with a '.med' extension. These files are in HDF5 format. The MED Memory group, which is a C++ and Python API, creates mesh and field objects in memory. Mesh creation can be done using set functions, or by loading a file. Physical fields are also created by loading files or initialization by user-defined functions. Finally the group of functions called MED CORBA facilitates distributed computation inside SALOME [9].

In order to have a working platform, input and output formats should be harmonized. The SALOME platform is based only on the MED format, which is also the data exchange model used inside the platform to transfer data among different components. It manipulates objects that describe the meshes where the scientific computations are performed and the physical fields defined on these meshes. This data exchange can be achieved either through files using the MED File formalism or directly in memory with the MED Memory (MEDMEM) library. In Table 1.2 we report the compatibility of the platform codes with the MED format. This compatibility is required in order to harmonize the codes with respect to input/output files.

1.2.2 GUIs over the platform and QT libraries

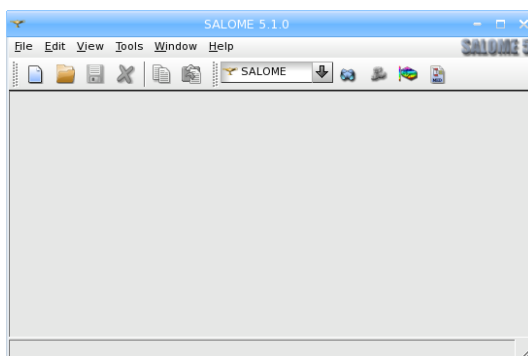


Figure 1.1: SALOME User Interface Toolkit for loading GUI applications

It is fundamental for the users to be able to operate through a common Graphical User Interface. If different applications must be used it is a desirable feature that they can be integrated inside the same framework. In order to load the GUI of each single application inside SALOME a compatibility with the Python-QT or QT libraries is required. If this is the case The SALOME GUI can be adapted

to load different GUI applications. The SALOME GUI is based on the SALOME User Interface Toolkit (SUIT) that allows a very flexible and powerful interaction with the SALOME tool components and QT exceptions or QT signal processing. With this approach the SALOME multi-desktop dockable-windowed user interface is available at the highest level. The Graphical User Interface of each application should have a plug-in structure, so that each GUI module is placed in a dynamic library which is loaded on demand. All applications can add their own menu items,

code	Py-QT compatibility	native GUI libraries
SALOME	yes	Py-QT
SALOME-GEOM	yes	Py-QT
SALOME-MESH	yes	Py-QT
SALOME-POST-PRO	yes	Py-QT
SALOME-PARAVIS	yes	Py-QT
CATHARE	yes	Py-QT
NEPTUNE	no	own
TRIO_U	no	own
SATURNE	yes	Py-QT
SYRTHES	yes	Py-QT
FEMuS	yes	Py-QT
PARAVIEW	yes	Py-QT

Table 1.3: QT library GUI compatibility

buttons in toolbar and windows.

The SALOME GUI provides a common shell for all components, which can be integrated into the SALOME platform, and some basic GUI functionalities, common to all modules. The **File** menu puts in common the actions of creation, saving, loading and editing files. The **Edit** menu gives access to Copy/Paste commands to transfer the objects from one application into another one. The **View** menu provides desktop management functionalities. Basically, it shows or hides toolbars, activates and manages application windows. The **Help** menu gives access to the help of SALOME-integrated applications. The appropriate help page can also be called from any operation dialog via the Help button. These are common functionalities, but specific pages for each application can be loaded as well. By importing widgets the original GUI of any application can appear in a particular window of the SALOME framework and load its own pages. In Table 1.3 we report the compatibility of the GUI of different codes with the QT libraries. With the exception of the NEPTUNE and TRIO_U codes [15, 19], the GUIs are all compatible with the QT libraries. The GUI of CATHARE is called GUIHARE, that is based on the SALOME GUI and its input deck includes the data block used to define each elements, the geometry and the topology of the reactor. However, the members of the NURISP user group have access only to the executable version of CATHARE, hence only the developers

can fully integrate it into the SALOME platform [11].


1.2.3 Parallel computations over the platform and MPI libraries

code	MPI local	MPI support on CRESCO
SALOME	OPEN/MPI	yes
SATURNE	OPEN/MPI	yes
SYRTHES	OPEN/MPI	yes
TRIO_U	CH/MPI	partial
NEPTUNE	LAM/MPI	yes
CATHARE	OPEN MP	partial
FEMuS	OPEN/MPI	yes
PARAVIEW	OPEN/MPI	partial

Table 1.4: Implementation status of the MPI libraries

Parallel computations are effective on a common platform only when a unique MPI library is implemented and the same domain decomposition algorithm is used. The Open MPI library is a good candidate to this end. Open MPI represents the merger between three well-known MPI implementations: FT-MPI from the University of Tennessee, LA-MPI from Los Alamos National Laboratory and LAM/MPI from Indiana University. The Open MPI library also integrates the contributions from the PACX-MPI team at the University of Stuttgart. The Open MPI development team consists of these four institutions.

The implementation status of the codes of the FISSICU platform is described in Table 1.4. For each code, we report under the labels “MPI local” and “MPI support on CRESCO” the possibility of using the MPI libraries provided with each code over a local workstation or the CRESCO GRID, respectively. Some applications have been developed with their own MPI libraries (for example a specific version of LAM/MPI) and this can lead to difficulties in the integration on the platform due to specific features. Furthermore, the version of the CRESCO MPI libraries is not the same as that used by many applications, therefore the executable version of a few codes cannot run in parallel on CRESCO. Finally, we remark that a complete integration can be achieved only when the input data and the simulation results are managed by the same library with the same formats.

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	15	121

CATHARE model validation with the SPES-99 facility

2.1 Introduction

This section is devoted to a simple validation of the CATHARE code and it can be considered an extension of the work presented in [3]. The present work concerns the development of a geometrical model based on the system code CATHARE and the simulation of the thermal-hydraulic behavior of the SPES-99 facility, including the comparison between the experimental data and the simulation results.

The geometry of the facility is described in great detail by considering rules and approaches for the simulation of the different components as recommended by the code developers. The reference steady-state conditions have been attained with CATHARE by means of a regulation algorithm of the main parameters in order to reproduce the real thermal-hydraulic loops as accurately as possible. The stability of the calculated steady state has been verified through a simulation period without any regulations. The comparison of the CATHARE results with the experimental data has requested an additional iteration on the facility model in order to reduce the uncertainties related to the thermal coupling between the primary and secondary circuits as well as to calibrate the pressure drops calculated along the primary circuit. The results of the simulation have highlighted the good capability of the model to predict the behavior of the facility also in transient conditions, although the comparison with the experimental data has suggested some possible improvements of the model. This report summarizes the final work and presents several considerations that were rather preliminary in [3].

We recall briefly the geometry of the SPES-99 facility which is described in more details in [3]. The facility consists of two primary loops. Each of these two loops includes a Hot Leg (HL) and two Cold Legs (CL). The hot leg connects the reactor pressure vessel to the steam generator (SG). The two cold legs detach from a vertical discharge line of the single primary coolant pump. A device simulating a 10-inches-equivalent break is mounted on the cold leg 2 of loop B to carry out the intermediate break test.

The reactor pressure vessel is composed by the lower plenum, the riser, where the rod bundle is placed, the upper head and the down-comer. The down-comer consists

Cooling Fluid	water
Number of loops	2
Number of pumps	2
Design Primary Pressure [MPa]	20
Design Secondary Pressure [MPa]	20
Primary Design Temperature [°C]	365
Design Secondary Temperature [°C]	310
Maximum Power [MW]	9
Height Scaling factor	1:1

Table 2.5: Main characteristics of SPES-99

of an annular section, where the four cold legs and the two Direct Vessel Injection (DVI) nozzles are attached, and an outer pipe connecting the annular section to the lower plenum.

The rod bundle is electrically heated with a maximum power of 7 MW and consists of 97 skin-heated Inconel rods. The pressurizer consists of a cylindrical flanged vessel equipped with two immersed heaters and six external ones. It is connected to the loop hot leg. There are two centrifugal, single-stage pumps, with an horizontal shaft. The suction line is horizontal and the delivery is vertical, discharging downwards into a pipe upstream of the two cold legs. The facility has two identical steam generators to transfer thermal power from the primary to the secondary circuit. The SG primary side consists of a 13 Inconel600 tube bundle, which are assembled in a square lattice, and inlet/outlet plena. The secondary side has full elevation up to the top of the steam separator and the secondary separators (dryers) are located at the top of the steam generator.

The main characteristics of the SPES 99 facility are reported in Table 2.5. For more details one can see [3].

2.1.1 CATHARE model for the SPES-99 facility

The latest version V2.5.1 of CATHARE 2 has been adopted to simulate the SPES-99 facility. The facility nodalization has been developed by respecting the geometrical dimensions of the different parts and components as well as the topology of the circuits [27]. In particular, the modular structure of the code has allowed to find a good compromise to preserve heights, flow areas and fluid volumes. It is worth reminding that the facility is scaled 1:1 in height and 1:397 in volume respect to the AP-600 reactor. In Figures 2.2 and 2.3 the nodalization schemes of the primary vessel and of loop A, that includes the pressurizer, are reported.

Some choice on the nodalization of the facility can strongly influence the results of the simulation. The vessel annular downcomer has been represented with a zero-dimensional (0D) component (DWC_ANN) in order to easily describe the

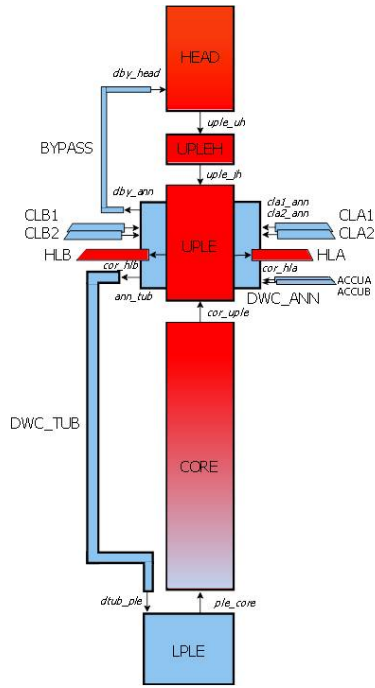


Figure 2.2: Vessel nodalization

high number of connections in this part of the circuit (4 cold legs, downcomer-upper head bypass, accumulators injection, tubular downcomer). However, this 0D component does not allow to take into account the inertial forces (the internal velocity is neglected) and other possible multidimensional effects. Even if the utilization of axial or 3D components is not considered in the present work, their effects will be probably evaluated in the future with sensitivity analyses.

Since the thermal capacity of the wall structure could affect the gas expansion and therefore the water injection rate, the accumulators have been simulated by means of a detailed description of the discharge line.

2.1.2 Regulated Steady State

The CATHARE code has an algorithm (PERMINIT) that provides the initial steady-state conditions of the thermal-hydraulic parameters starting from a set of guess values and introducing the necessary corrections to achieve the calculation convergence. This is the first step to obtain the reference steady state that was recorded on the facility before running the Intermediate Break LOCA test.

In order to get thermal-hydraulic conditions near the reference steady state ones, a regulation procedure of the main parameters has been adopted that has reproduced the real regulations of the facility as close as possible. These conditions has been achieved according to a procedure recommended for CATHARE [8] which requires the use of the transient calculation algorithm (TRANSIENT) and the control of the

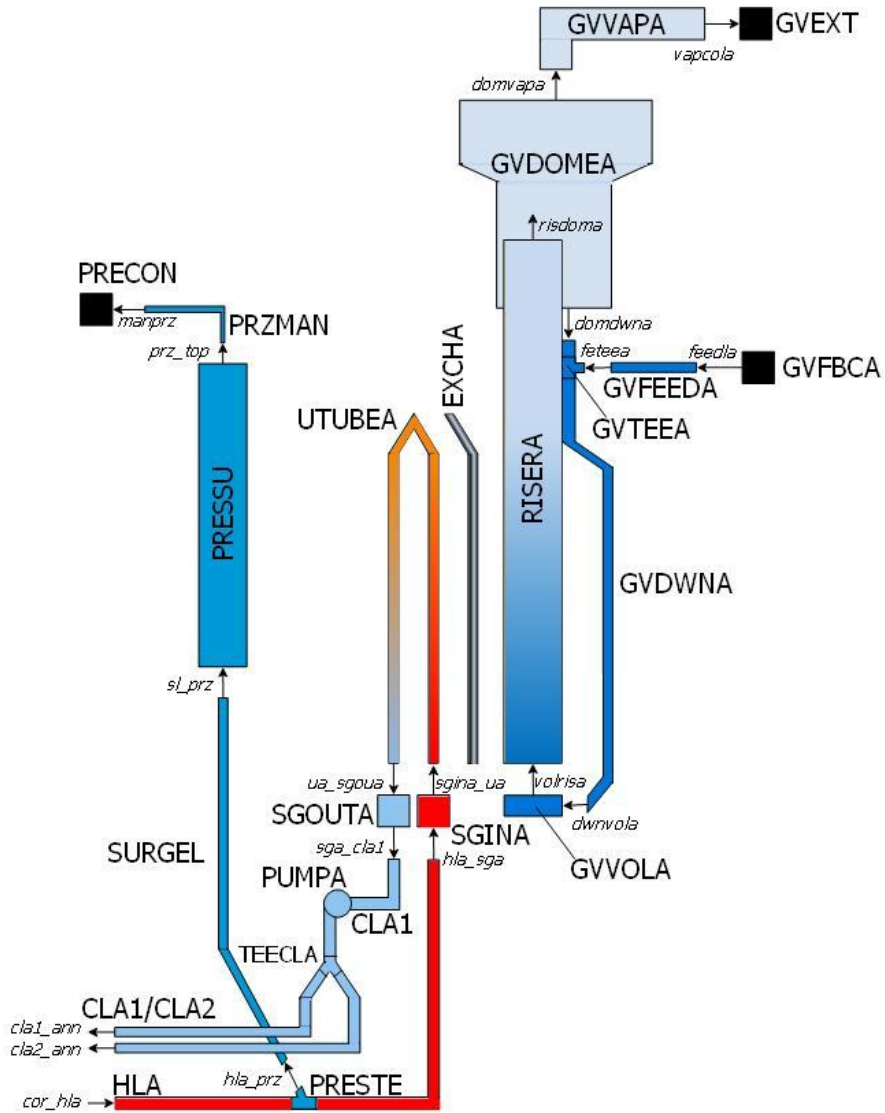


Figure 2.3: Loop A nodalization

main parameters directly from the input-deck. To this purpose, the action is on the boundary through a proportional-integral regulation.

Some parameters were considered in the transient regulation calculation, namely primary mass flow rate (MFR), liquid level in the pressurizer, primary pressure, fluid inventory in the steam generators and feed-water mass flow rate. These parameters have been controlled for 3500 seconds until the acceptance criteria have been satisfied. In the following 1500 seconds of calculation there is no external regulations in order to verify the stability of the achieved steady state. The results of this regulated calculation are compared with the experimental values in Table 2.6. Important deviations are related to the temperatures in the primary circuit that are about 4 °C lower than the measured values, and the rotation velocity of the pumps

that is 20% slower than the experimental values.

Steady State Conditions	Experimental Values		Calculated Values	
Heater Rod Power (MW)	4.97		4.92	
Pressurized Pressure (MPa)	15.37		15.36	
CL Temp. (A1, B1) [°C]	279.7	277.6	273.8	272.9
CL Temp. (A2, B2) [°C]	279.4	277.6	273.8	272.9
Core Inlet Temp. [°C]	277.9		273.4	
Core Outlet Temp. [°C]	320.4		312.7	
HL Temp. (A, B) [°C]	315.8	316.9	312.5	312.5
Core MFR [kg/s]	23.55		23.55	
CL MFR (A1,B1) [kg/s]	6.04	5.56	6.23	5.78
CL MFR (A2,B2) [kg/s]	6.24	5.82	6.05	5.65
Pump speed (A, B) [rpm]	3057	2769	2453	2299
DC-UH bypass MFR [kg/s]	0.13		0.13	
Pressurizer level [m]	3.77		3.72	
SG pressure (A, B) [MPa]	4.97	4.94	4.96	4.96
SG Dome level (A, B) [m]	0.8	0.8	0.8	0.8
SG FW Temp. (A ,B) [°C]	225.6	226.9	226.0	226.0
SG Dome Pres. (A ,B) [MPa]	5.16	5.08	4.97	4.97
SG MFR (A ,B) [kg/s]	2.00	2.20	1.41	1.32

Table 2.6: Main parameters after the regulation phase

2.1.3 Reference Steady State

With a further tuning on the experimental data has it has been possible to reduce the uncertainties related to certain phenomenologies in order to obtain a reference steady state in better agreement with the experimental results.

The lower values of the primary temperatures calculated by CATHARE indicate an over-prediction of the heat transfer in the steam generators that may be justified by the uncertainties in the heat transfer coefficients and in the fouling degree of the SG tubes. In order to solve this discrepancy the heat exchange area of the SG tubes has been reduced by 15% to match the experimental average primary temperature.

The under-prediction of the pump speed at the end of the regulation phase is due to an underestimation of about 20% of the total pressure drop in the primary circuit. The comparison between the pressure drops that are measured and calculated in the different parts of the circuit has revealed that the larger discrepancies are obtained in the annular downcomer (some internal structures implemented to represent the pressure drops expected in the AP600 are not simulated in the model), in the connections between upper plenum and hot legs and in the steam generators. These last two differences can be attributed to the uncertainties in calculating the

geometrical pressure drops at the entrance and exit of the large plenum. Moreover, in the U-tubes of the steam generators the fouling phenomena could be responsible for an increase of the distributed pressure drops with respect to the ideal situation considered in the calculations.

The following geometrical coefficients used to calculate some concentrated pressure drops have been re-calibrated on the experimental data: connection between cold-legs and annular downcomer, entrance of hot-legs from the upper plenum, connections between hot/cold-legs and SG plena.

The final results of the reference steady state calculation are reported in Table 2.7.

Steady State Conditions	Experimental Values		Calculated Values	
Heater Rod Power (MW)	4.97		4.93	
Pressurized Pressure (MPa)	15.37		15.36	
CL Temp. (A1, B1) [°C]	279.7	277.6	277.0	276.0
CL Temp. (A2, B2) [°C]	279.4	277.6	277.0	276.0
Core Inlet Temp. [°C]	277.9		276.5	
Core Outlet Temp. [°C]	320.4		315.2	
HL Temp. (A, B) [°C]	315.8	316.9	315.1	315.1
Core MFR [kg/s]	23.55		23.68	
CL MFR (A1,B1) [kg/s]	6.04	5.56	6.16	5.70
CL MFR (A2,B2) [kg/s]	6.24	5.82	6.14	5.67
Pump speed (A, B) [rpm]	3057	2769	2723	2603
DC-UH bypass MFR [kg/s]	0.13		0.13	
Pressurizer level [m]	3.77		3.72	
SG pressure (A, B) [MPa]	4.97	4.94	4.96	4.96
SG Dome level (A, B) [m]	0.8	0.8	0.8	0.8
SG FW Temp. (A ,B) [°C]	225.6	226.9	226.0	226.0
SG Dome Pres. (A ,B) [MPa]	5.16	5.08	4.97	4.97
SG MFR (A ,B) [kg/s]	2.00	2.20	1.41	1.32

Table 2.7: Main parameters after the calibration phase

2.2 SPES-99 IB LOCA Transient

2.2.1 Experimental data

The Intermediate Break LOCA test consisted in a 10-inches-equivalent break in the cold leg B2 starting from full power and pressure conditions. The test was defined by a working group constituted by ENEA, ANPA, JRC Ispra, ANSALDO, Pisa University and SIET [28]. The experimental boundary conditions of the test are reported in Table 2.8 together with the relevant thermal-hydraulic events.

Test Boundary Conditions	Specified Time (s)	Actual Time (s)
Break opening	0	0
Steam Line A closure	P<12.41 MPa + 2s	5.
Steam Line B closure	P<12.41 MPa + 2s	5.
Scream Signal	P<12.41 MPa + 3.5s	6.5
FW A Closure	P<11.72 MPa+ 2s	7
FW B Closure	P<11.72 MPa + 2s	8
PRZ empty	-	8
UP in saturation cond.	-	4.5
Pump A trip (coastdown)	P<11.72 MPa + 16.2s	25 (4)
Pump B trip (coastdown)	P<11.72 MPa + 16.2s	25 (8)
SG A PORV open/close	P (SG) = 6.30/6.20	13/20
SG B PORV open/close	P (SG) = 6.30/6.20	14/17
Two-flow at break	-	17
First dry-out (starting)	-	38
Secondary P \downarrow Primary P	-	46.8
ACC A Injection start	P (DVI)<4.8 MPa	74.5
ACC B Injection start	P (DVI)<4.8 MPa	73
ACC A Injection stop	-	725
ACC B Injection stop	-	646
Second dry-out (starting)	-	1908
Electrical Power shut-off	Rod Clad Temp \downarrow 664 °C	2600
End of test	-	2712

Table 2.8: Test boundary conditions

The main phases of the transient can be summarized in the following steps:

- a fast depressurization after the break;
- the first heater rod heat-up due to DNB;
- the accumulator intervention with quench of the first heat-up;
- the continuous loss of mass from the break and the second heat-up due to dry-out of the core;
- the end of test for electric power supply interruption at the cladding temperature of 664 °C.

The trends of the main quantities with the indication of the events are shown in Figures 2.4 and 2.5.

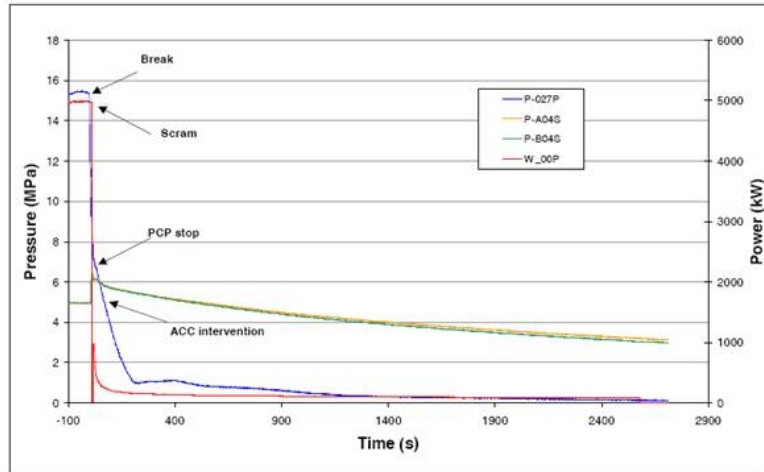


Figure 2.4: PRZ, SG-A, SG-B pressure and core power

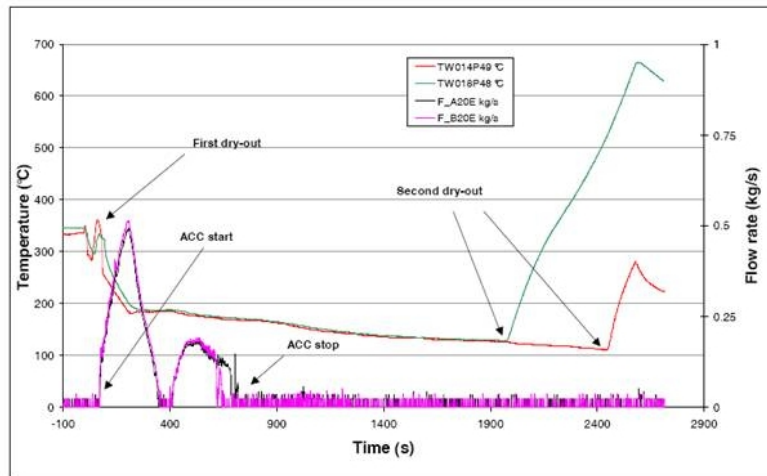


Figure 2.5: ACC-A/B flow rate and heater rod temperature

2.2.2 Comparison between post-test results and experimental data

The IB LOCA test has been calculated with the CATHARE model, as described in the previous section, and with the RELAP5 model. For the CATHARE transient the calculation is started from the reference steady-state conditions achieved after the calibration phase (Table 2.7). The boundary conditions reported in the second column of Table 2.8 as well the power curve provided in the experiment, shown in Table 2.9, have been imposed by means of the transient algorithm of CATHARE.

The main calculated parameters are compared with the experimental data and the RELAP results in Figures 2.6 to 2.11.

Time (s)	Power (kW)	Time (s)	Power (kW)
0	4991.54	27.5	401.28
5.75	4991.54	30	381.70
5.76	978.74	35	347.45
12.38	978.74	40	318.09
14.5	978.74	45	293.62
15	753.63	50	278.94
17.5	636.18	70	234.89
20.82	548.09	100.	195.78
22.5	450.22	200	156.59
25	425.75	500	127.24
...	...	1000	107.66

Table 2.9: Power curve

Figure 2.6 shows the fast depressurization of the primary system. This behavior is well predicted by CATHARE in the first seconds of the transient while in the following ones the calculated depressurization is faster than in the experiment. The slightly anticipated intervention of the accumulators in the CATHARE calculation highlighted in Figures 2.7 to 2.8 is a direct consequence of the faster depressurization of the loop, whereas the prediction of an interruption in the injection behavior is due to the presence of the short re-pressurization. The anticipated intervention of the accumulators could be related to the rough geometrical description of the downcomer.

In the same figure one can see that the depressurization of the primary system predicted by RELAP simulation is in better agreement with the experimental data, when compared to the CATHARE predictions.

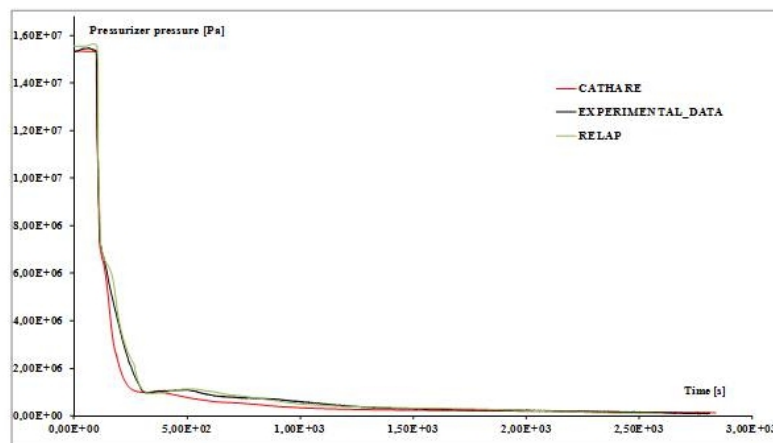


Figure 2.6: Pressurizer pressure

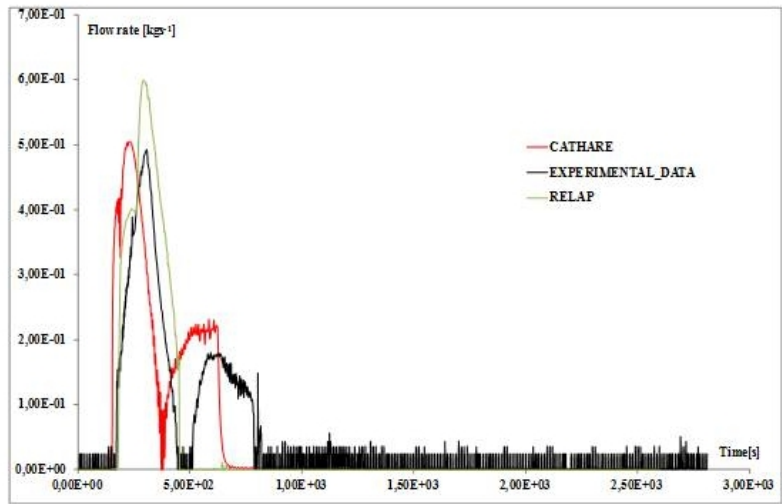


Figure 2.7: Accumulator A injection mass flow rate

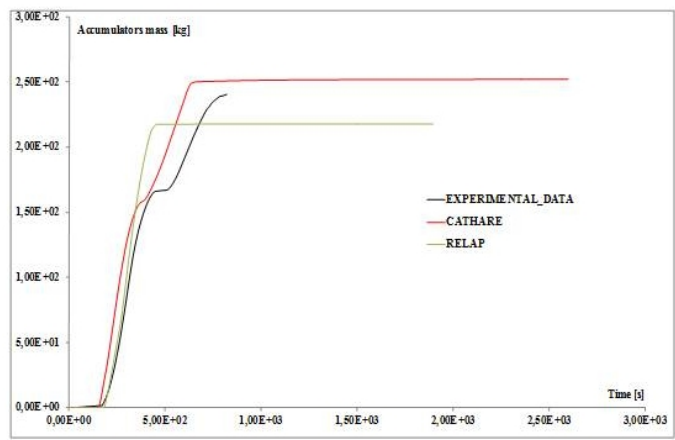


Figure 2.8: Integral mass injected by the accumulators

In order to improve the simulation of the fast depressurization phase of the transient a more detailed model than the present 0D volume should be adopted for the downcomer. The utilization of 3D components, which are able to take into account the inertial forces and multidimensional effects, will be investigated in the future.

In spite of these discrepancies, the occurrence of DNB phenomena is well predicted by the CATHARE code, in particular at the middle level the rod heat-up period is precisely calculated, as shown in Figure 2.9. The clad peak temperature is higher than in the experiment and the effect of mesh refinement in the power channel should be investigated. In the same figure it is clearly see that the RELAP simulation doesn't present the DNB phenomenology.

Figures 2.9 to 2.11 report the rod clad temperature at three different heights and

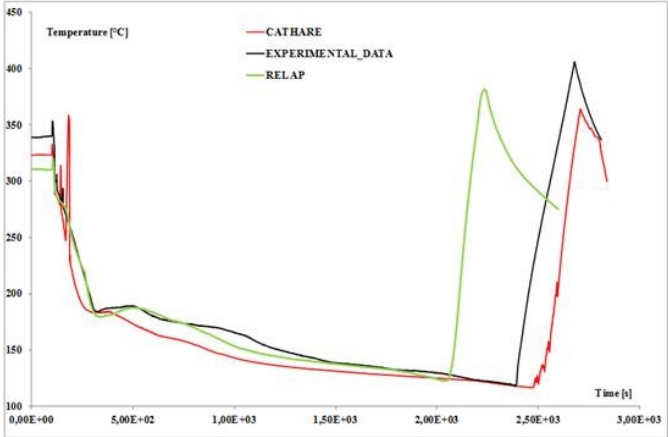


Figure 2.9: Rod clad temperature in the middle height

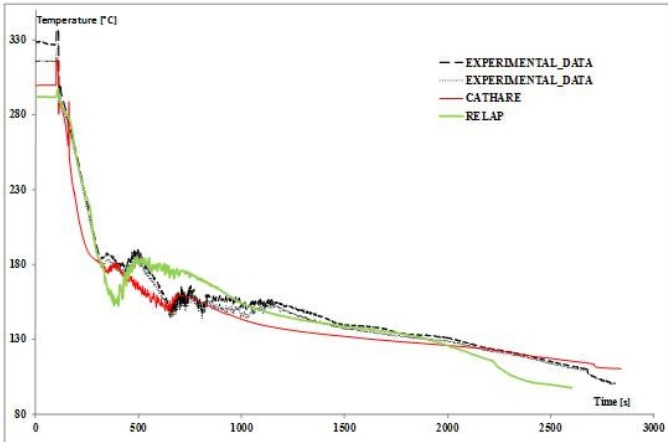


Figure 2.10: Rod clad temperature in the lower height

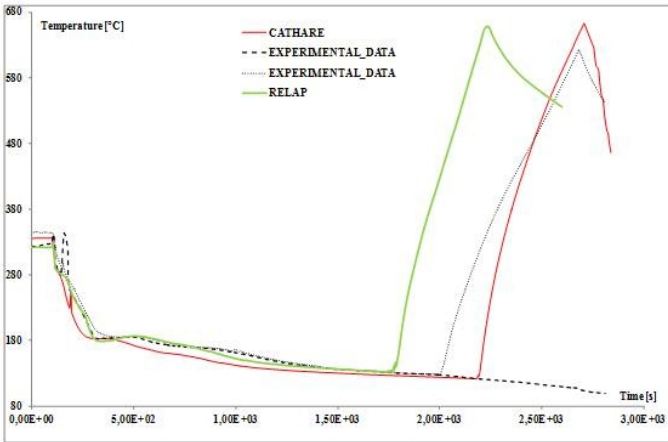




Figure 2.11: Rod clad temperature in the upper height

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	26	121

confirm that the intervention of the accumulator is effective in quenching the first heat-up of the heated rods.

The second part of the transient, that represents the continue loss of mass from the break until the dry-out of the core, is calculated in good agreement with the experiment. The occurrence of the second heat-up at about 2000 s is well predicted by both codes at the different heights of the power channel, but with a slight delay for CATHARE and a greater advance for the RELAP simulation, when compared to the experimental data. Once again, mesh refinement in the power channel could be a good candidate to improve the results of the simulation. The correct prediction of the dry-out occurrence, as shown by the CATHARE code, could be a confirmation of the correct estimation of the primary inventory and consequently of the mass flow rate due to the break, but unfortunately experimental values are not available.

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	27	121

Validation of the CATHARE and NEPTUNE models for the PERSEO facility

In the framework of the NURISP EU project ENEA is developing and validating methods to couple thermal-hydraulic systems and CFD codes. The objective of such an activity is to simulate experimental or accidental transients, with the proper level of detail required to obtain a reliable prediction of the time sequence of the events, including their uncertainties. The reproduction of the PERSEO (in-Pool Energy Removal System for Emergency Operation) experiment performed at SIET laboratories (Piacenza, Italy) was considered a good test for this kind of activity. The reasons for this statement are: *a*) the PERSEO facility is a full scale simulator for Decay Heat Removal (DHR) systems with in-pool heat exchangers, thus the size and flow characteristics are similar to those encountered in Light Water Reactor (LWR) plants; *b*) the CATHARE simulation of the experimental transient has demonstrated that, in order for the computed results to be in agreement with the experimental data, the nodalization of the steam suppression pool should be properly tuned on the experiment, due to the high level of inhomogeneities of the flow in such a component. In a preceding report [3] the results obtained by coupling the system calculation with CATHARE and a 2D CFD simulation of the steam suppression pool (OP) have been discussed. In the present document the results obtained by performing 3D simulations of the OP are reported.

The present chapter is organized as follows: first the PERSEO facility and Test 9 are reviewed; then the CATHARE nodalization and the obtained results are presented; finally the two-phase flow obtained in the OP by considering the discretization of the real 3D geometry and of two simplified geometries will be discussed.

3.2 The PERSEO facility test

3.2.1 The PERSEO facility

The PERSEO facility was designed to test heat removal system of PWR reactors when energy removal systems using in-pool heat exchangers were proposed to be

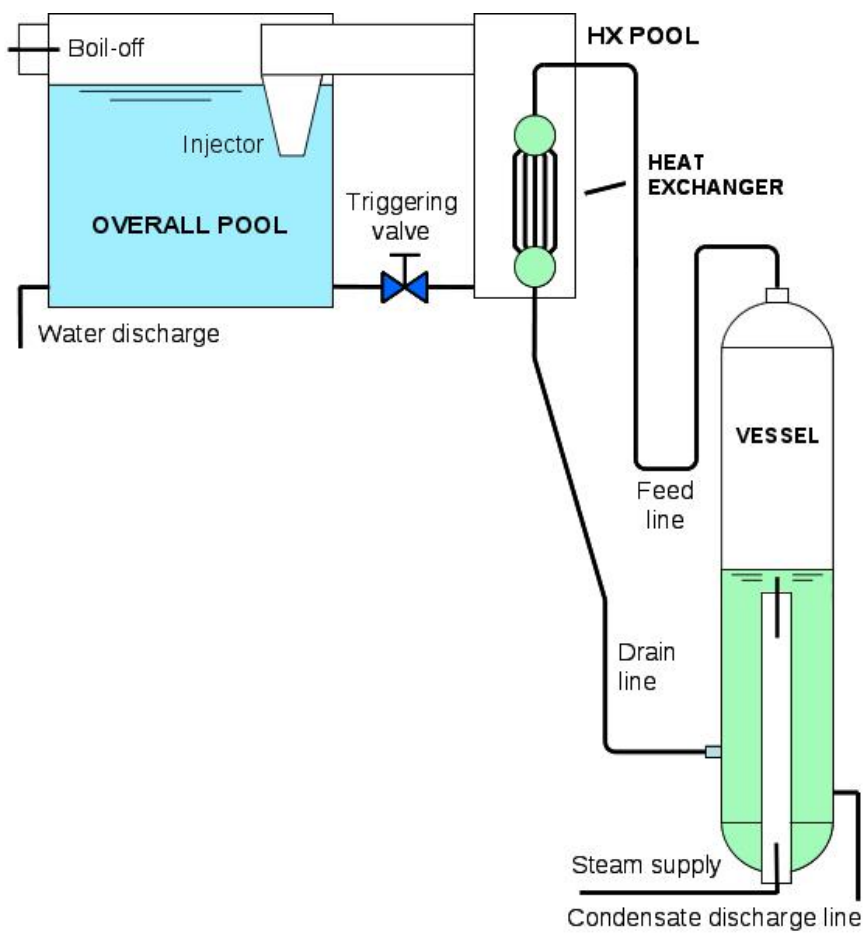


Figure 3.12: The PERSEO (in-Pool Energy Removal System for Emergency Operation) facility

installed in both the GE-SBWR and the Westinghouse AP-600. In particular two heat removal systems were considered for these reactors: the Isolation Condenser (IC) and the Passive Residual Heat Removal (PRHR). In both heat removal systems the heat transfer is started by opening a valve installed on the primary loop. This concept was modified many times. A first proposal, called the Thermal (TV) Valve concept, was studied by CEA and ENEA [50, 57]. In this study the primary loop valve, which was at high pressure and temperature, was moved to the pool loop. Later on in the design the valve was relocated on the steam loop, but at the top of a bell covering the pool with the immersed heat exchanger. In emergency conditions the valve should start the steam discharge formed under the bell and the heat transfer from the primary to the pool loop. The main problem of this configuration was the large valve needed to avoid flow instabilities. Again the idea to move the valve from the primary loop to the pool loop was proposed by ENEA and SIET [48] as an evolution of the TV concept. This configuration has the triggering valve on the secondary loop, in the water line connecting the two pools. A new experimental facility was designed and built at the SIET laboratories by modifying the existing PANTHERS IC-PCC facility (Performance Analysis and Testing of Heat Removal System Isolation Condenser – Passive Containment Condenser). The scheme of the PERSEO (in-Pool Energy Removal System for Emergency Operation) facility is depicted in Figure 3.12. The PERSEO system mainly consists of a primary loop and a secondary one. The primary loop contains the pressure vessel and the heat exchanger which are connected by the steam feed line and the condensate drain line. The secondary loop consists of two pools connected at the top by a steam duct ending with an injector discharging into the overall pool (OP) and at the bottom by a water line with the triggering valve. The Heat Exchanger (HX) is contained in the HX pool. The overall pool contains the water that can be used to cool the primary loop. A test campaign has been conducted on the PERSEO facility in order to verify

Primary side pressure	MPa	4
Primary side temperature	K	523.15
Primary steam MFR	kg/s	8
HX extracted power	MW	14
HX pool side pressure	MPa	0.12
HX pool steam MFR	kg/s	6.5

Table 3.10: The main PERSEO parameters at full power operation for Test 9

the correctness of the proposal and in particular the effectiveness of heat removal. During the operations, the pressure vessel is maintained at saturation conditions for a pressure of about 7 MPa, which is typical of the BWR primary loop or the PWR secondary loop with the steam generators. The vapor pressure is maintained by supplying properly de-superheated steam generated from a nearby power station and is kept constant by controlling the steam supply valve. At the same time the

water level in the vessel is maintained at a specified value by discharging water through the condensate discharge line. At the beginning of the test the HX, the feed line and the drain line are all full of saturated steam. The HX pool can be either full of air or steam, depending on the test, while the OP is full of cold water at the nominal level and the triggering valve is closed. Once the system has reached the test initial conditions the triggering valve is opened and the HX pool is flooded by cold water which can condensate the steam inside the HX tubes. With power transfer from the primary to the pool loop the water starts boiling, and the steam produced in the HX pool is driven towards the OP through the steam duct. At the beginning of the test, the injector is below the water level and it contributes to the mixing of the OP water, thus avoiding temperature stratification. The water inside the OP heats up until the boiling point is reached. The steam produced in the OP flows outside the system at atmospheric pressure through the boil-off outlet. When the water level gets below the injector no condensation takes place any longer in the OP and the steam flows directly outside through the boil-off pipe. The water inventory in the OP decreases according to the heat transfer rate in the HX pool. During the system operation, the natural circulation which becomes stable in both the primary and secondary loops determines the power removed by the system. In some tests, water is discharged directly from the OP bottom to accelerate the transient phase characterized by a decreasing water level.

3.2.2 PERSEO experimental Test 9

Event	Time (s)
Beginning of the test	0
Triggering valve opening	142 (in 21 s)
OP water discharge opening	2790
Onset of OP water boiling	3200
OP water discharge closure	4840
Triggering valve closure	4887 (in 93 s)
End of test	7708

Table 3.11: Chronology of main events characterizing Test 9

Many different tests have been performed during the experimental campaign on the PERSEO facility [53]. Among all of them, test 9 shows the different phases of a long accidental transient. Therefore we have selected this test to investigate the performance of the model during the different phases. Test 9 starts with the HX pool completely full of water. Then, after reaching the boiling conditions, the pool level decreases. This test was aimed to the investigation of the system behavior during long-term accidental transients with a reduced primary pressure (~ 4 bar), when

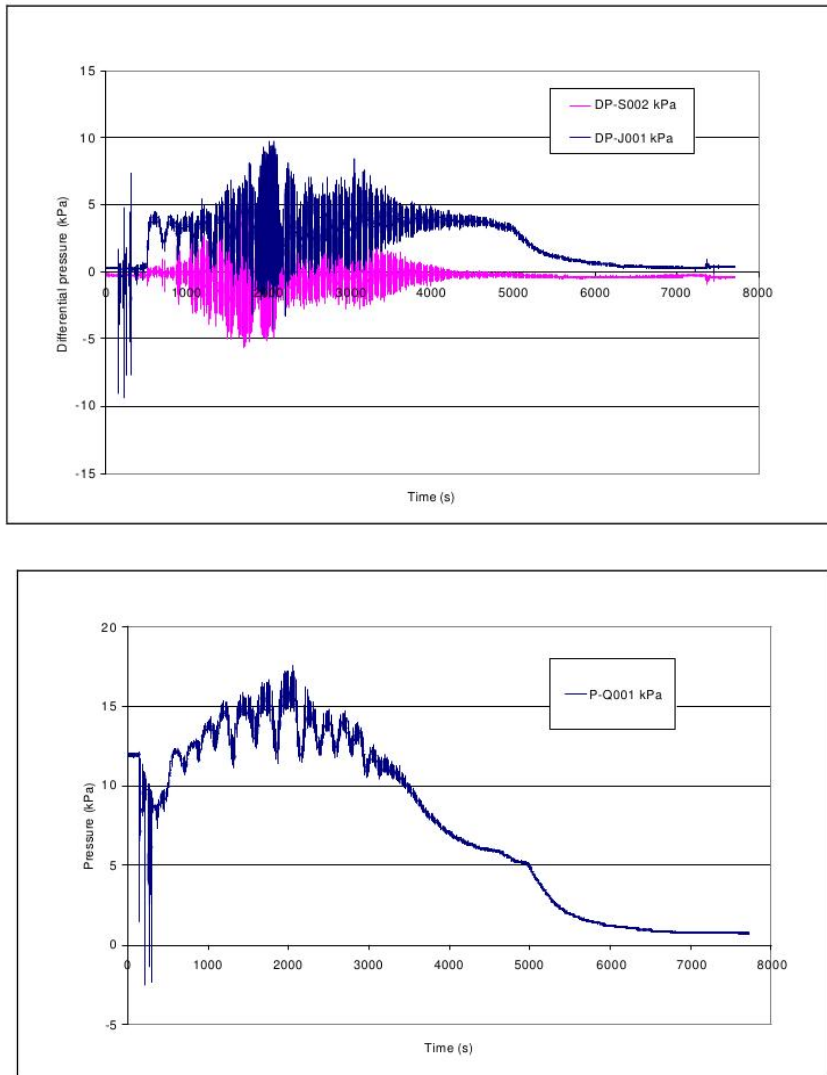


Figure 3.13: Test 9. Overall pool water level behavior (top) and HX pool relative pressure (bottom)

the HX pool is initially full of water and thermal regimes are reached in both pools. Furthermore, this test studies the effectiveness of the injector to mix the OP water as well as the power and flow regime variations after the OP water level decreases below the injector nozzle. The main PERSEO parameters at full power operation for Test 9 during the transient phase are listed in Table 3.10. The chronology of the main events characterizing the test is shown in Table 3.11.

The overall behavior of the system is illustrated in Figures 3.13 and 3.14. After opening the triggering valve at $t = 163$ s, the water level in the HX pool, initially rather low, quickly increases close to the top of the tube bundle. The progressive decrease of the OP water level is started by the water discharge from the bottom

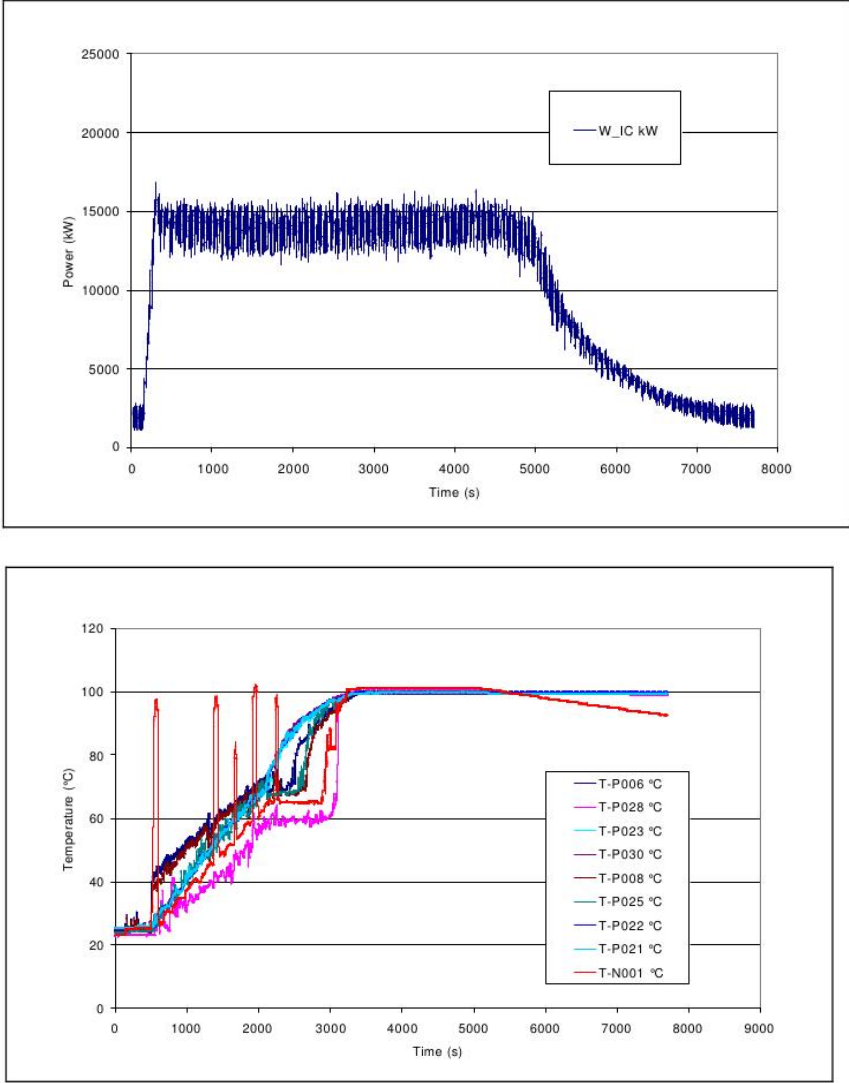


Figure 3.14: Test 9. HX exchanged power (top) and overall pool temperatures (bottom)

at $t = 2790$ s, approximately 400 s before the onset of boiling. The water discharge is stopped at $t = 4840$ s, that is 140 s before the complete closure of the triggering valve. As a consequence, the OP water level drop is terminated in conjunction with the beginning of the decrease of the extracted power which tends to zero at the end of the test.

The extracted power is 14 MW and the onset of boiling in the OP takes place around time $t = 3200$ s. As expected the temperature stratification phenomenon in the OP disappears as soon as the boiling condition is reached in the pool.

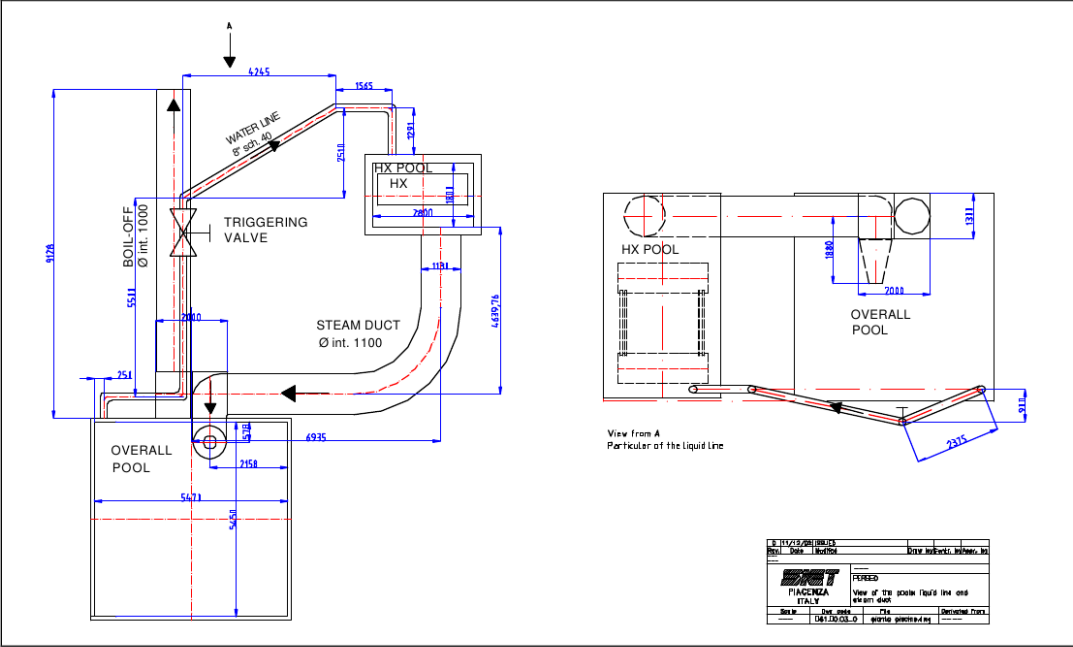


Figure 3.15: PERSEO facility steam duct and water line between the two pools

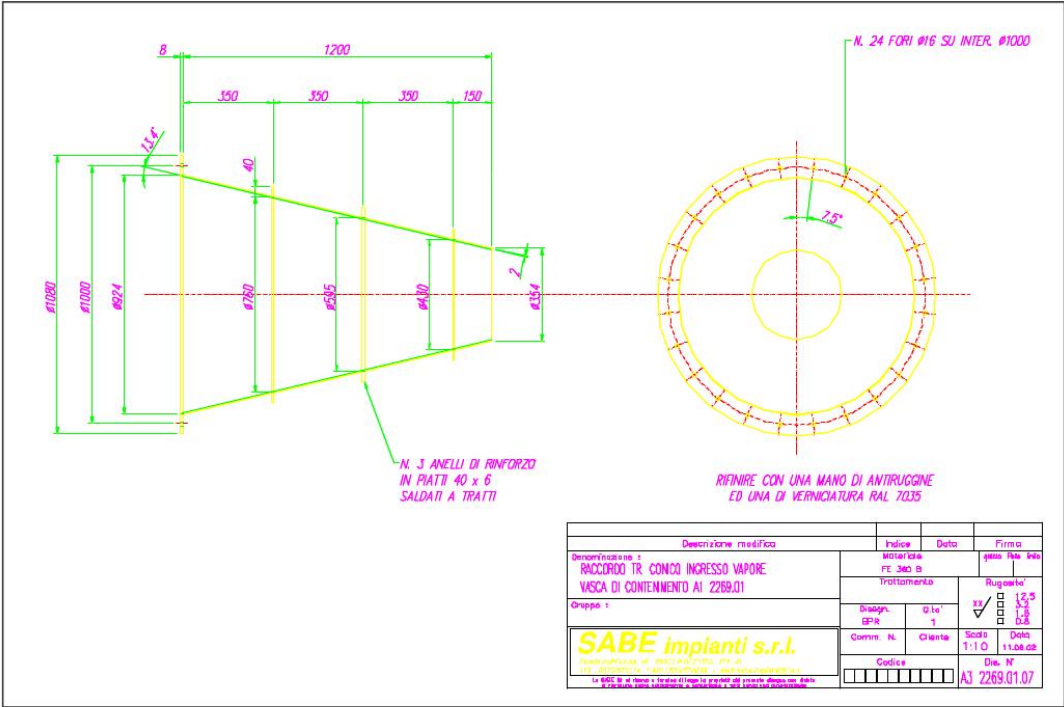


Figure 3.16: Injector or ending part of the steam duct in the overall pool

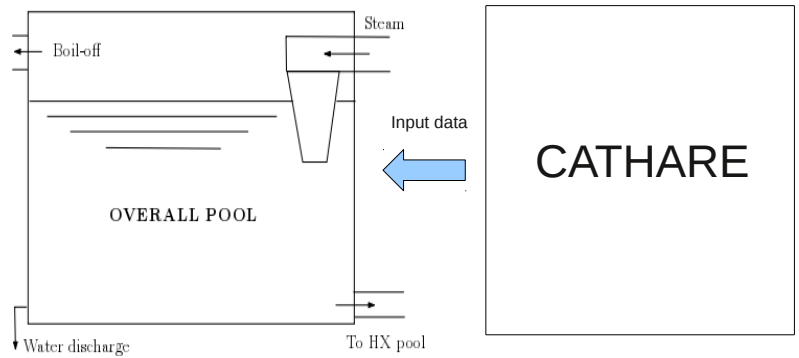


Figure 3.17: Coupling of CATHARE and NEPTUNE codes

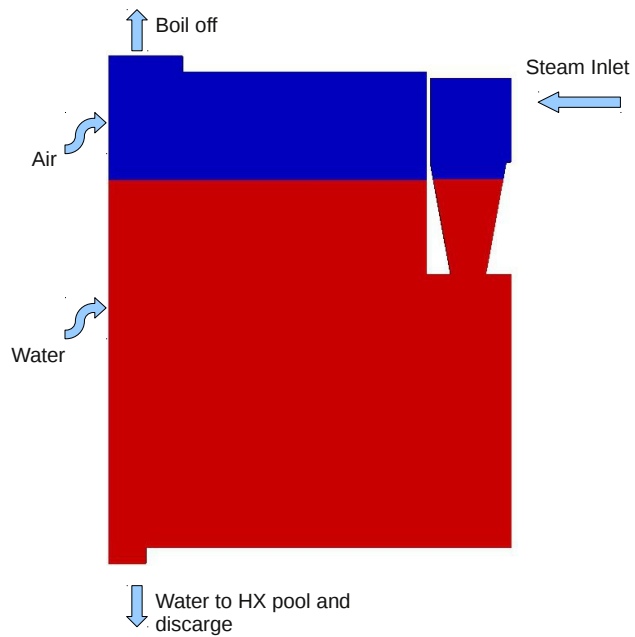



Figure 3.18: Boundary conditions for the component made up by the overall pool and the injector

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	35	121

3.3 CATHARE-NEPTUNE coupled simulation

In this section are presented the results obtained from the analysis of the PERSEO facility using the three-dimensional CFD (Computational Fluid Dynamics) code NEPTUNE. The complexity of the geometry of the PERSEO facility does not allow the 3D simulation of the entire system. For this reason this simulation has been restricted to the most interesting component given by the overall pool and the injector. A detailed sketch of this component is shown in Figures 3.15-3.16 with dimensions and geometric details, while for the CFD simulation appropriate boundary conditions (velocity and temperature fields) have to be given. One of the objectives of the present activity is to couple a system code of the whole system to a CFD code used to investigate a selected component. For this reason the CATHARE code has been coupled to the NEPTUNE simulation through the boundary conditions.

A "weak" coupling has actually been adopted. A "strong" coupling implies that CATHARE and NEPTUNE are solved in separate domains and the separate solutions are iterated until convergence is reached in the whole domain. In the procedure here considered the evolution of the entire system is obtained first with CATHARE, then the flow field in the overall pool and in the injector are solved by the NEPTUNE code, using the CATHARE results to set proper boundary conditions.

The weak coupling between CATHARE and NEPTUNE is sketched in Figure 3.17. The water discharge (WD) is a line that allows water removal in order to accelerate the decrease of the water level in the OP. The water-to-HX-pool line (WL) is the feed line from the OP pool to the HX pool. The steam line allows the water vapor to flow from the HX pool to the injector, while both steam and air flow out the system through the boil-off section. The boundary conditions which are required for this weak coupling between the OP-injector system and the rest of the PERSEO facility are the mass fluxes through the water discharge (WD) and the water-to-HX-pool (WL) lines; atmospheric pressure is imposed at the boil-off section (BO) for the outgoing gas flow while at the inlet section of the injector both temperatures and mass fluxes (or steam velocity) are required. The boundary velocities for the NEPTUNE code can be computed from the mass fluxes predicted by the CATHARE code. Since the state variables in CATHARE are 1D along the components axis, uniform fields across each boundary section have been considered.

3.3.1 CATHARE solution of the PERSEO facility

Test 9 on the PERSEO facility has been simulated by using the CATHARE code and two different nodalizations. Details on the CATHARE solution and discretization can be found in [49]. In this section only the main results of this activity are reviewed, to explain the motivations for the code coupling which has been considered for this type of simulations.

The first nodalization which has been considered is shown in Figure 3.19. The 0D two-node module (volume) is used to represent the primary vessel, the HX collectors,

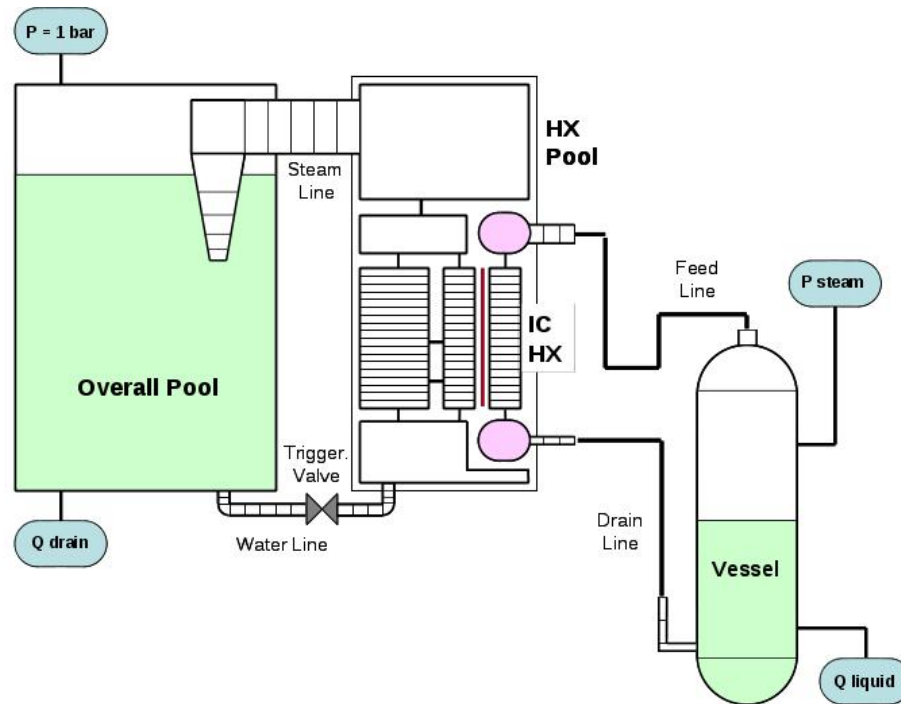


Figure 3.19: CATHARE discretization of the PERSEO facility

the HX pool lower and upper plenum and the OP (complete mixing). The volumes are interconnected by axial elements (pipe) representing the HX tube bundle, the water and steam lines. Axial elements with cross-flow junctions are used to better simulate 2D recirculation within the HX pool and the HX variations of water level during the transient phase. Empirical correlations are used in the system code in order to reproduce the experimental data. In particular the EPICE correlation for boiling heat transfer in the HX pool and the SUPERCLAUDIA correlation for direct contact condensation in the OP are implemented in the code [51]. A pressure boundary condition at the HX pool top was kept active for about 100 s during start-up in order to avoid large numerical instabilities. These instabilities are probably generated by the natural circulation within the HX pool and are the cause of a very fast and strong condensation of the steam in the upper plenum when cold water is injected at the pool bottom. A good evaluation of the pressure drop through the steam injector is needed to compute accurately the HX pool relative pressure and thus the overall system behavior. Therefore, friction losses through the conic-shaped injector are calibrated on differential pressure measurements. To this aim, appropriate singular pressure drop coefficients are taken into account in the injector to reproduce the test measurements.

A second nodalization has been introduced to get a better agreement with the experimental data on the water level in both the OP and HX pools. This nodalization is presented in Figure 3.20 where one can see how the OP pool has been modeled by using three volumes. The reason for this nodalization is related to the

consideration that the void distribution is not expected to be homogeneous inside the steam suppression pool. The highest void fraction is in fact expected to be near the injector exit.

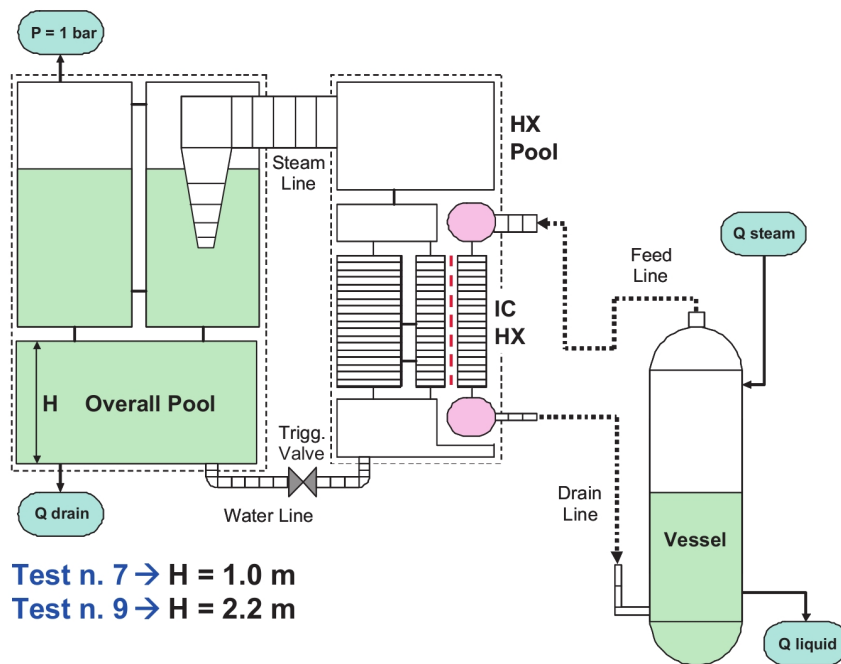


Figure 3.20: CATHARE discretization of the PERSEO facility with three volumes for the OP

In Figure 3.21 the evolution for test 9 of the collapsed water level in the HX pool is shown for the two different nodalizations. It is clearly seen that with the three volume nodalization there is a better agreement between the experimental and the computed values. It should be stressed that the volume partition is subjected to some discretionality. Here three 0D volumes has been used to obtain a good fit of the experimental data. Should the experimental data be not available, the CATHARE user should find a reliable partition for the physical problem under investigation. The possibility to model the OP through a CFD code coupled with the system code should remove this source of uncertainty.

3.3.2 Boundary conditions for NEPTUNE

As previously stated, the boundary conditions for the NEPTUNE code are obtained from the CATHARE solution. Since the state variables in CATHARE are 1D, uniform fields across the different boundary sections have been considered. In Figure 3.22 the mass flow rate through the main duct connecting the HX to the OP pool is presented. On the right of Figure 3.23 is shown the CATHARE solution for the

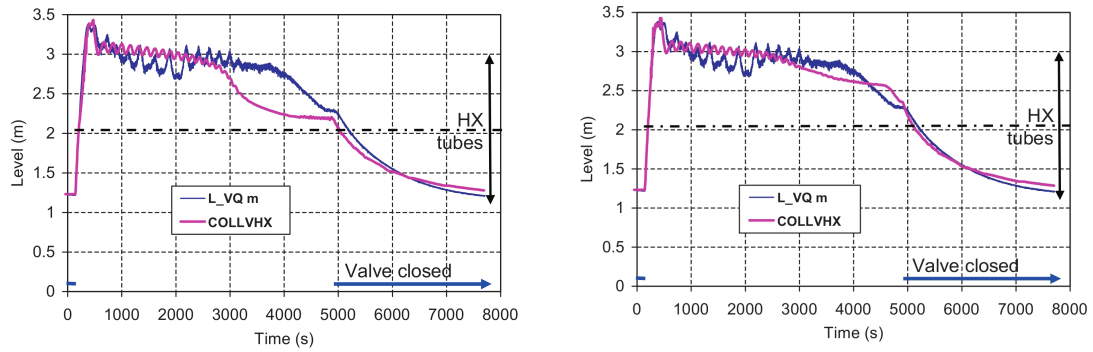


Figure 3.21: HX pool water level behavior with one (left) and three (right) volumes nodalization for the overall pool (CATHARE solution in pink)

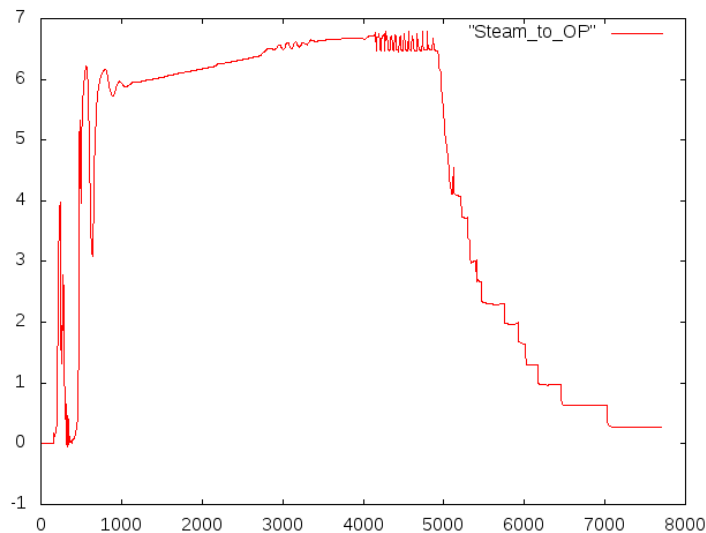


Figure 3.22: Steam mass flow rate [kg/s] to the OP pool as a function of time [s]

mass flow rate through the water discharge (WD) line. This line allows a faster water removal to accelerate the decrease of the water level in the OP pool. On the left of the same figure is reported the mass flow rate of water from the overall pool to the water-to-HX-pool (WL) line.

3.4 NEPTUNE 3D simulation set up on CRESCO

3.4.1 Code setup

The NEPTUNE project provides a two-phase thermal-hydraulics code for two- and three-dimensional computations of the main components of nuclear reactors: cores, steam generators, condensers, heat exchangers. NEPTUNE supports from one to

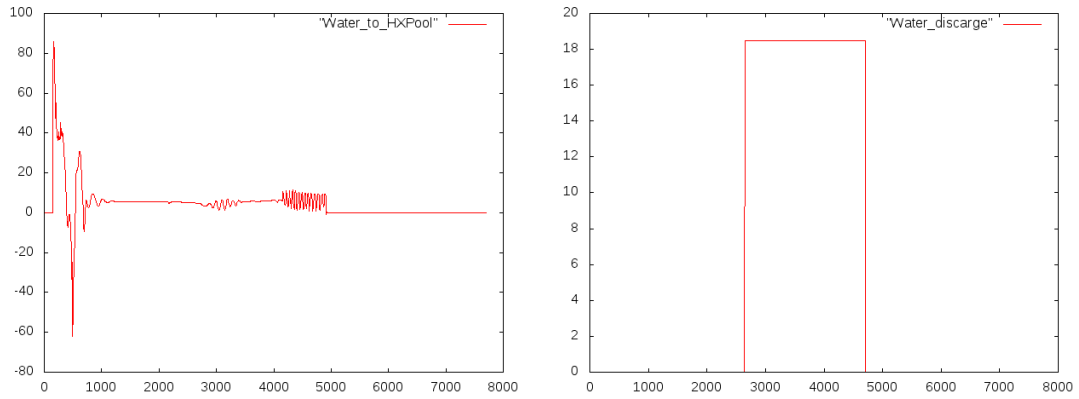


Figure 3.23: Water mass flow rate [kg/s] to the HX pool (left) and to the water discharge line (right) as a function of time [s]

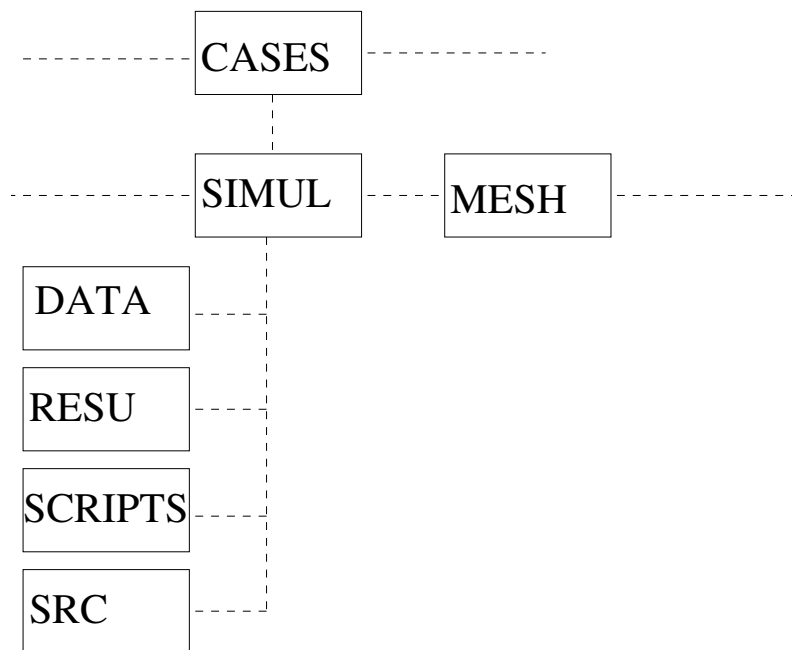



Figure 3.24: Directory of the NEPTUNE code

twenty fluid fields (or phases) and includes thermodynamic laws for water/steam mixtures. It is based on advanced physical models (two-fluid equations combined with interfacial area transport and two-phase turbulence) and a discretization with cell-centered finite volumes. This method can be used on meshes with any type of cells and nonconforming connections. Actually, during this work it turned out that the two-phase flow solver is only applicable to hexahedral meshes, a fact that strongly limits the applicability of the code to complex geometries. The developing team reported that the problem is likely to be overcome with the release of the next version of the code.

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	40	121

The NEPTUNE code is located on CRESCO-ENEA GRID in the directory

```
/afs/enea.it/project/fissicu/soft/Neptune3/Neptune
```

and it requires a specific structure for the configuration and input files. In particular, we can create the main directory **CASES** and put all NEPTUNE simulations inside. If the simulation directory is denoted as **SIMUL** we can create a folder tree for this study with the available script

```
$ buildcase_nept -study SIMUL
```

This generates the correct directory structure. Inside the main directory **CASES** there is a **MESH** directory, where all the meshes are stored, and the study **SIMUL** will have its own directory

```
CASES
+-- CASE1
+-- SIMUL + working directory
...
+-- MESH
```

As shown in Figure 3.24, the working directory **SIMUL** contains the four sub-directories

- **DATA**, where the XML configuration file is stored;
- **RESU**, that is used for the outputs;
- **SCRIPTS**, that hosts the execution scripts;
- **SRC**, in which we can put some additional source file.

Once the **PATH** is set the NEPTUNE GUI starts with the command

```
$ neptune
```

At the moment all the libraries required for the GUI are not available in the CRESCO architecture. Therefore the NEPTUNE GUI has to be run first on a simple workstation to create the file **param** with the code settings. This file is then copied on CRESCO in the appropriate position and the NEPTUNE code can be run in batch mode. In order to execute the application in a shell the proper environment is set by using the script **neptune_env** that is located in the **bin** directory. The recommended way to run NEPTUNE from the command line is to use the **runcase_cresco** script. This is a modified version of the original **runcase** script distributed with the code, that allows to use the script with the LSF parallel job queue manager installed on CRESCO. Templates of both scripts are available inside the directory

```
/afs/enea.it/project/fissicu/soft/Neptune3/
Neptune/package_neptcfid-1.0.8/NEPTUNE_CFD/neptcfid-1.0.8-r844/bin/
```

When the command `buildcase_nept` is executed, both scripts are copied in the `SCRIPTS` directory with the updated paths. If the user wants to customize the path to use a temporary work directory, he/she needs to change the `$NEPT_TMP_PREFIX` variable inside the script. The `runcase` script can be used to run the code in interactive mode for short calculations. There is an interactive node with 16 cores on the CRESCO system that can be used to test the calculation setup before submitting the job to LSF. Once the user is inside the CRESCO system through a front-end node, the machine can be found at the address

```
cresco1x006.portici.enea.it
```

One simply has to enter the `SCRIPTS` directory inside the case under analysis and type

```
$ ./runcase
```

In order to run the code in massive parallel mode and for longer simulation times the `runcase_cresco` script has to be used together with the `bsub` command from LSF. A typical example of job submission is given by

```
$ bsub -n 128 -J JOB_NAME -q cresco_16h24 -o OUT
-e ERR ./runcase_cresco
```


where the option `-J` sets the job name, `-n` the number of processors, `-o` the filename to which the stdout will be redirected, `-e` the error output and `-q` the queue to which the job will be submitted.

3.4.2 Initial conditions, boundary conditions and physical properties files

In the following sections are presented the results obtained by simulating the OP and the injector system on three hexahedral meshes, with different levels of approximation of the geometry. In the overall pool and the injector system three fluids are present: water, steam and air. A few of the physical properties of these fluids are listed in Table 3.12. The label “Cathare table” refers to an internal database

fluid	T (K)	$\rho(Kg/m^3)$	$\mu(Pa s)$	$C_p(W/m^3K)$
steam	297.75	Cathare table	Cathare table	Cathare table
water	420	Cathare table	Cathare table	Cathare table
air	297.75	1.1855	1.71×10^{-5}	1005

Table 3.12: Fluid and flow properties selected for PERSEO test 9

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	42	121

that correlates the pressure and temperature of the water-steam mixture to other physical properties. The reference (and initial) temperatures for all the phases are also shown in the table, with the initial pressure usually set to $p = 1$ atm. The turbulence model in water and steam is the $R_{ij} - \epsilon$ model recommended by the user guide, while the mixing length model is selected for air. The *Separated Phases* drag model and the *Large interface* model have been used for steam/water and air/water interactions, respectively. Simple source terms for enthalpy and momentum between air and steam had to be included in order to prevent these quantities to diverge during the calculation in the regions where each phase was only present with small traces. Actually, the numerical method allows a phase to appear below a given tolerance even where they are not physically expected to be. The conserved quantities of these "spurious phases" must be kept under control through proper source terms in order to prevent the calculation from diverging. The enthalpy source term in the steam enthalpy equation due to direct contact with the air is thus given by

$$Q_{23} = 10^6 \alpha_2 \alpha_3 (T_3 - T_2) \quad (3.1)$$

and has been coded in the user-defined subroutine `usth12.F`. Similarly, the source term in the steam momentum equation for the k -th direction due to direct contact interaction with the air is given by

$$M_{23}^k = \alpha_2 \rho_2 F d_{23} \frac{\alpha_3}{\rho_2} (U_3^k - U_2^k) \quad (3.2)$$

with

$$F d_{23} = 10^5 (1 + V_R^2) \left[\frac{kg}{m^3 s} \right] \quad (3.3)$$

where V_R is the module of the relative velocity between phases. This source term has been coded in the `ustsns.F` subroutine.

Five different boundary conditions have been imposed in the simulations: inlet, outlet, walls, water-to-HX-pool and water discharge lines. At the inlet and the two water lines the mass flow rates, that have been computed in the CATHARE simulations as functions of time, are imposed as integral values on the given boundary surface. These functions are coded in the `usclim.F` subroutine. The saturation enthalpy of the steam at the inlet and the water temperature at the entrance of the two water lines are given as well.

Friction boundary conditions have been imposed at the walls and standard wall functions are considered. Null enthalpy gradient condition is also used.

Finally, atmospheric pressure is considered at the boil-off outlet.

Many options can be activated in NEPTUNE. In this simulation the *Special Modules* option is used to enable special features of the two-phase flow. The option *water/steam module* is activated to use the Cathare tables for water/steam systems, as recommended by the code tutorial. In the *scalar* section of the NEPTUNE code the total enthalpy of each fluid is set, from which it is possible to compute the

corresponding temperature by using the appropriate thermodynamical law. Other physical laws and properties may be added in the file `usphysv.F`. Details to obtain air temperature from its enthalpy can be found in [2].

An iterative procedure will iterate over the fields $\alpha - p - H$ of the Navier-Stokes equations at each time step until convergence is reached. A tolerance of 10^{-6} on the α values has been imposed. Three iterations at each time step have been chosen in order to improve the stability of the calculation.

3.4.3 Experimental data

The aim of the simulation is to reproduce the temperature field of the PERSEO test. In Figure 3.25 is shown the position in the overall pool of the probes where the temperature is measured. The exact location of each probe is given in Table 3.13 and the experimental measurements are reported in Figure 3.26. In particular, the analysis will be restricted to a comparison of the computed temperatures with that of the probes denoted by TP6, TP8, TP23 and TP25.

Position	X	Y	Z	Probe
overall pool under the Injector	500	3400	2195	TP006
overall pool under the Injector	500	3400	1795	TP007
overall pool under the Injector	500	3400	1195	TP008
overall pool central area	1910	3745	2695	TP021
overall pool central area	1910	3745	2195	TP023
overall pool central area	1910	3745	1195	TP025
overall pool central area	1910	3745	50	TP028
overall pool central area	3540	3745	2695	TP030
overall pool central area	1910	3745	2495	TP022

Table 3.13: Exact location (in *mm*) of the probes for temperature measurements with respect to the reference frame indicated in Figure 3.25

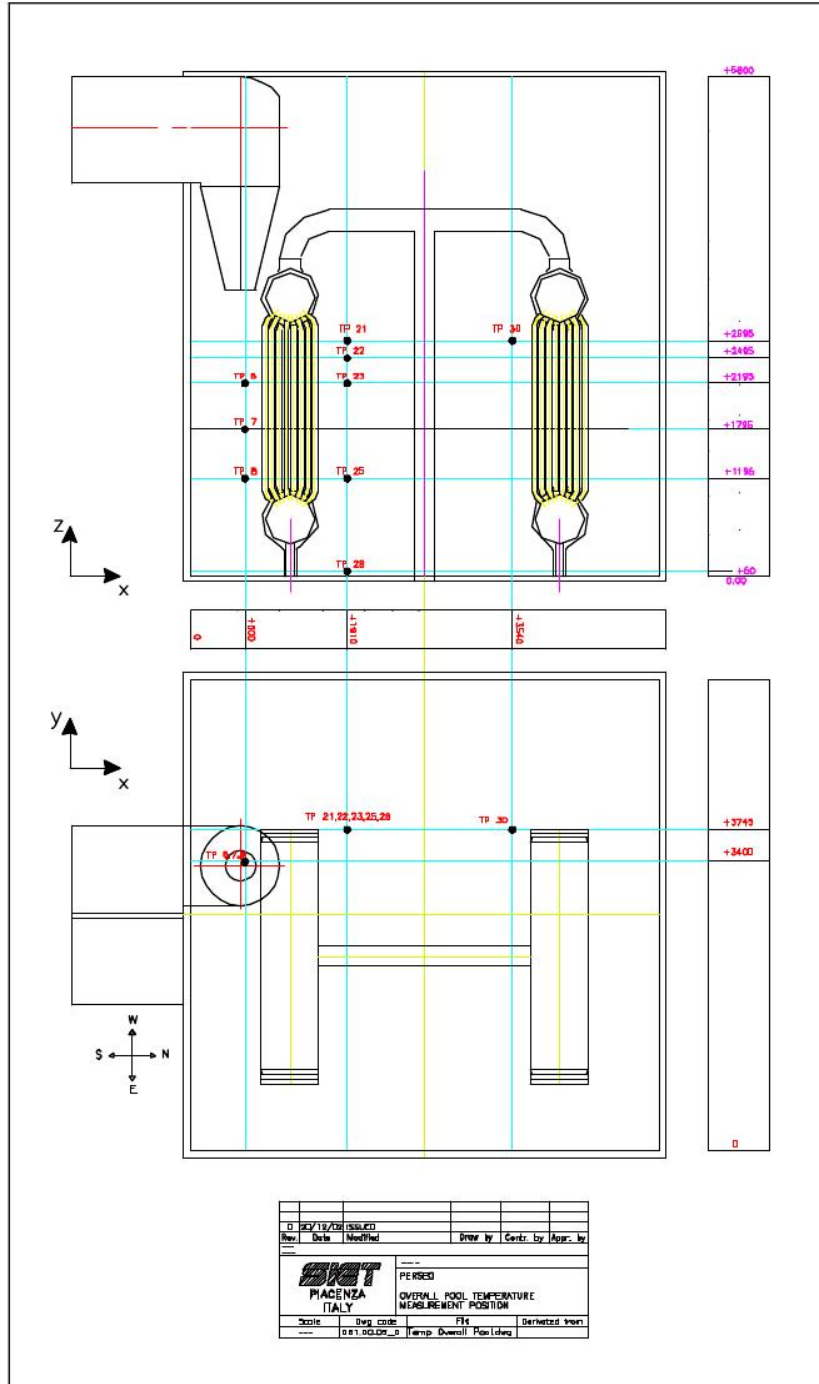


Figure 3.25: Overall pool temperature measurement positions

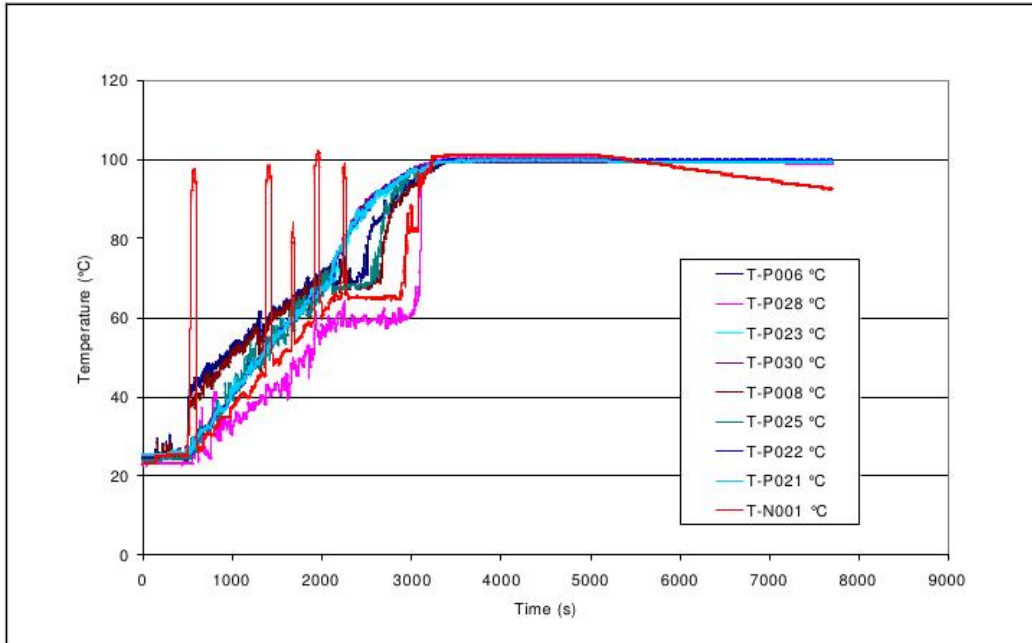


Figure 3.26: Local temperature recorded by the different probes in the overall pool

n	Description	Time (s)
1	Beginning of the test	0 -500
2	pool heating	500-3200
3	pool boiling	3200-4200
4	Low water level	4200-7700

Table 3.14: Main events for the temperature evolution

3.4.4 Results

By observing the temperature evolution for test 9 as reported in Figure 3.27, it is possible to divide the experiment into four parts:

1. Beginning;
2. Temperature stratification (pool heating);
3. Steam injection (pool boiling);
4. Injection above water level.

The time interval of each part is summarized in Table 3.14.

In the next sections the beginning of the experiment will be analyzed on three different meshes. The first mesh has been constructed on the real geometry of

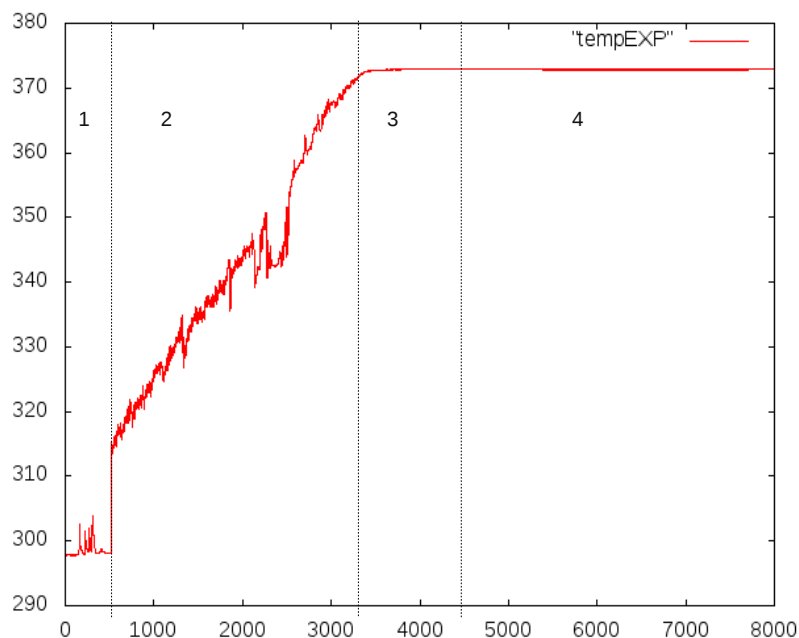


Figure 3.27: Temperature evolution as recorded by probe *TP6* (Temperature in [K] and time in [s])

the overall pool and of the injector. The second mesh has been developed from a modified geometry, characterized by a straight vertical injector entering the pool from the top of the domain rather than from the side. The injector nozzle is at its real position, close to the pool wall. The third mesh has a straight vertical injector as well, which enters the pool from the center of the upper face, rather than close to the lateral wall.

Results obtained on the real geometry

In this section the results obtained with the simulation on the first mesh are reported. The domain includes the overall pool OP and the injector. In Figure 3.28 the external part of the mesh and a detail of the injector and boil-off zone are shown. The grid is composed by 85041 hexahedral cells. It can be seen how the most refined zones are close the injector nozzle and below the injector bend. The dimensions of the small cells below the injector bend are determined by the high curvature of the injector wall and by the proximity to the pool wall. In these cells the local Courant number has its highest values. The time step is then determined by the field values in these cells, while the Courant number in the rest of the computational domain is much lower. This results in a very low efficiency of the calculation. In addition, very small time steps may induce instabilities on the calculation, because of the time derivative terms in the pressure equation.

It should be noted that the grid has been constructed by taking advantage of the

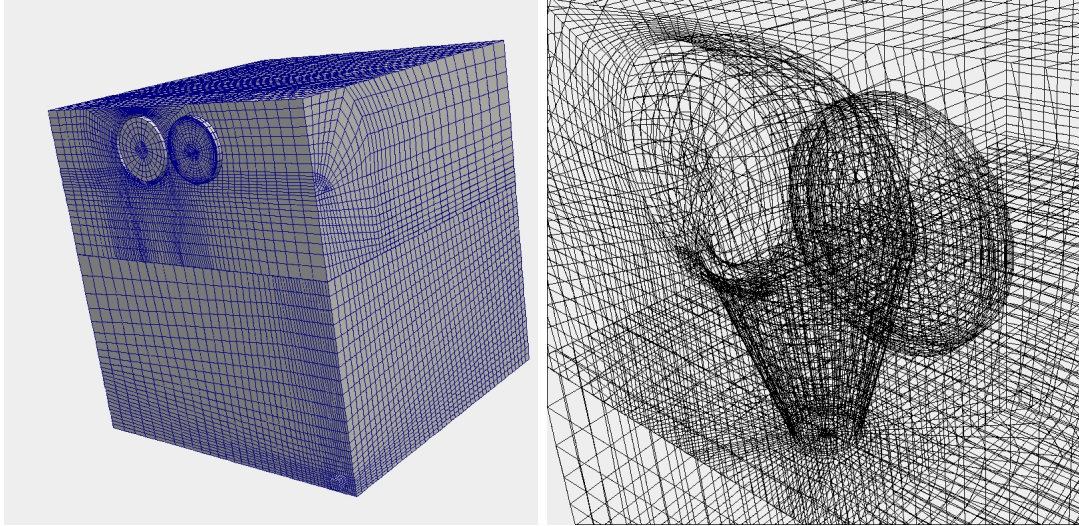


Figure 3.28: PERSEO 3D hexahedral grid (left) and a detail of the injector and boil-off region (right) for the real geometry

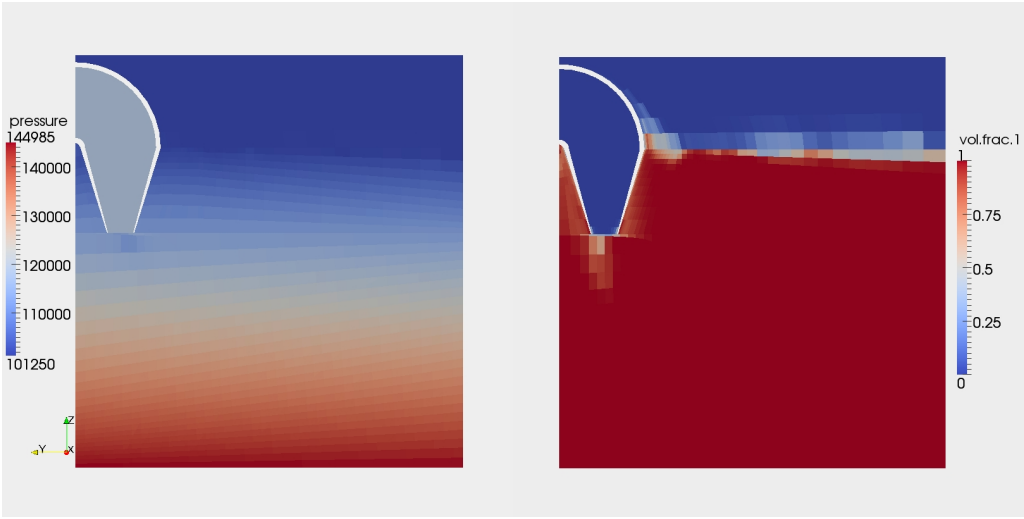


Figure 3.29: Pressure [Pa] (left) and water volume fraction (right) on the vertical section through the injector axis, at $x=3.12$ m and $t_1 = 9.08398$ s

NEPTUNE capability to deal with non-conforming grids. As a matter of fact, the first mesh is given by the union of different parts: the upper half of the pool, the lower half of the pool, the injector, the boil-off and the water line sections. All the junctions are non conforming. It must be emphasized that the code proved to be very sensible to sharp changes in the cell dimensions, even across the grid junction. Therefore, a great attention must be given to this aspect while constructing new grids.

Since the steam mass flow rate is approximately zero up to the time $t_0 = 153$ s,

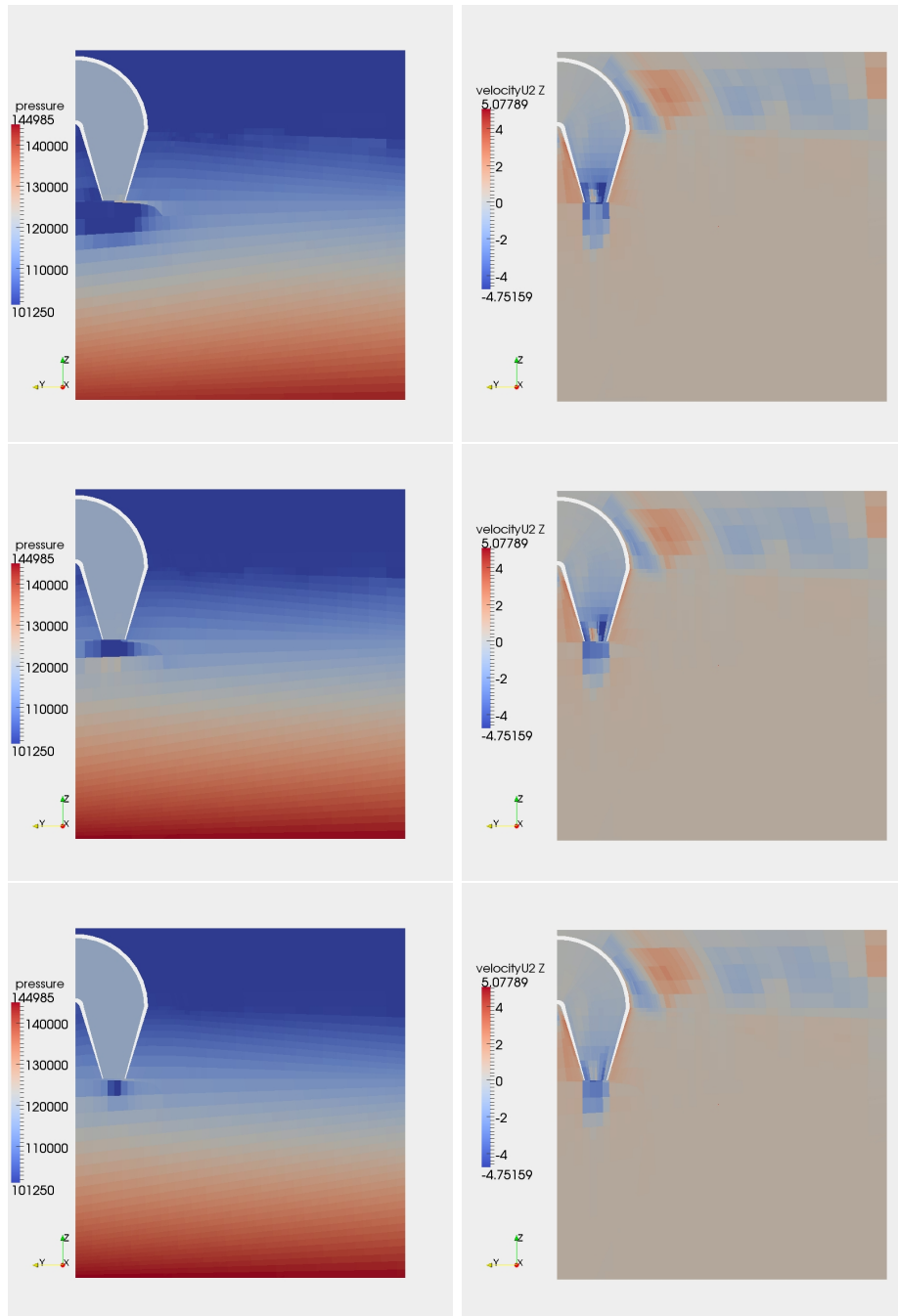


Figure 3.30: Pressure [Pa] (left) and vertical component of the steam velocity [m/s] (right) in the same section of Figure 3.29 at times $t_1 = 9.09419, 9.09833, 9.10886$ s (top to bottom)

the simulation is shifted in time by the quantity t_0 so that the real time is $t = t_1 + t_0$, where t_1 is the simulation time. In Figure 3.29 the pressure and water volume fraction fields are shown on a vertical section passing through the injector

axis at time $t_1 = 9.08398$ s. It can be observed that the steam condenses right at the injector nozzle and that the cell deformation near the free surface causes some distortion in the surface reconstruction itself. In Figure 3.30 a sequence of pressure and steam vertical velocity distributions are presented in the time interval 9.09419 s $< t_1 < 9.10886$ s. At this early stage of the experiment the steam mass flow rate is very low ($Q_{steam} = 0.157$ kg/s) and its velocity does not exceeds 5 m/s. This is the reason for the prompt steam condensation right at the injector nozzle. As it can be seen from the pressure field, the steam condensation causes a pressure drop near the nozzle which then extends to the region between the injector and the pool wall. These pressure fluctuations are also caused by the fact that the volume occupied by the steam inside the injector is too small to compensate the volume lost during the condensation and in a low-mach number formulation of the problem, the boundary conditions do not react to such a loss in volume.

This pressure fluctuations propagate inside the injector until the simulation finally does not converge. However, many sources of instability overlap in this simulation. For this reason it has been decided to generate new grids starting from a simplified geometry, in order to isolate and correct the most important factors that lead the simulation to divergence.

Results obtained on the geometry with a straight injector and the nozzle at the real position

In order to reduce the possible sources of numerical instabilities, a new geometry has been generated by introducing a straight injector entering the pool from the top of the domain. The injector nozzle is in the same position and with the same area.

The grid is still composed by different blocks: lower half of the pool, upper half of

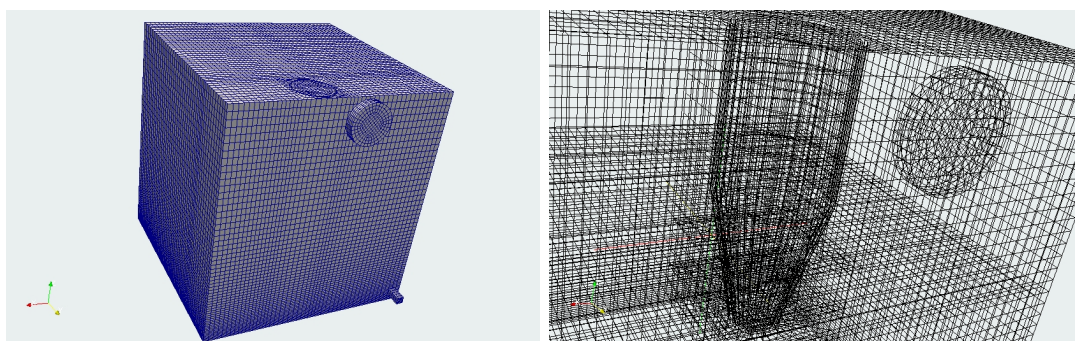


Figure 3.31: PERSEO 3D hexahedral grid (left) and a detail of the injector and boil-off region (right) for the geometry with a straight injector and the nozzle in the right position

the pool, block around the injector, boil-off, injector, (WD) and (WL) water lines. With respect to the first mesh, a block of cells in the part of the pool around the injector as been added. This block has conforming junctions with the remaining

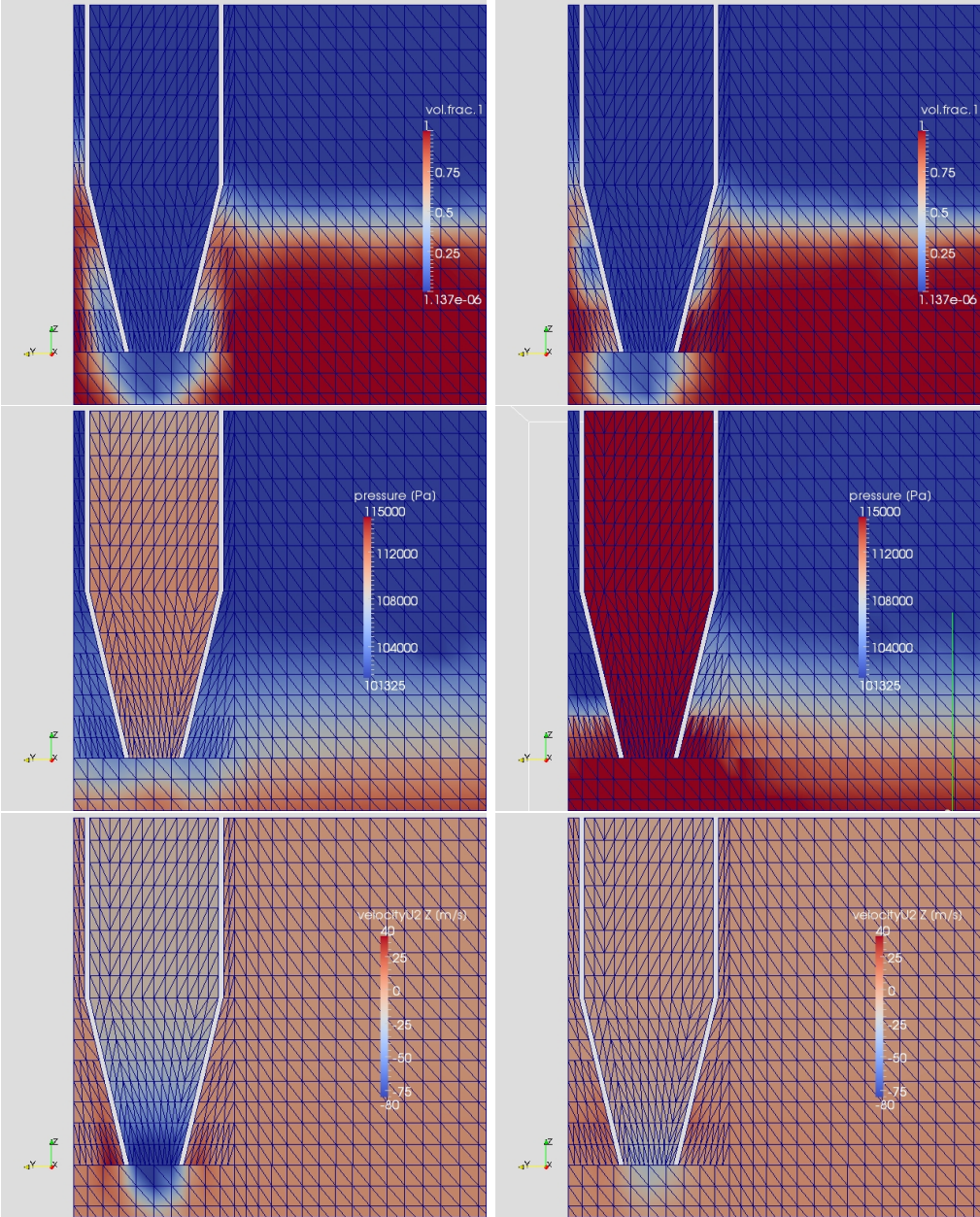


Figure 3.32: Water volume fraction (top), pressure (middle) and vertical component of steam velocity (bottom), on the vertical section through the injector axis at $x=3.12$ m and times $t_1 = 50.6804$ s (left) and $t_1 = 51.8309$ s (right)

part of the upper pool, even though the grid size in the normal direction of the connecting surfaces undergoes a steep variation, especially in the upper part of the injector (again the short distance from the pool wall requires small cells) and near the injector nozzle.

The junctions between the grid in the lower half of the pool and the other blocks

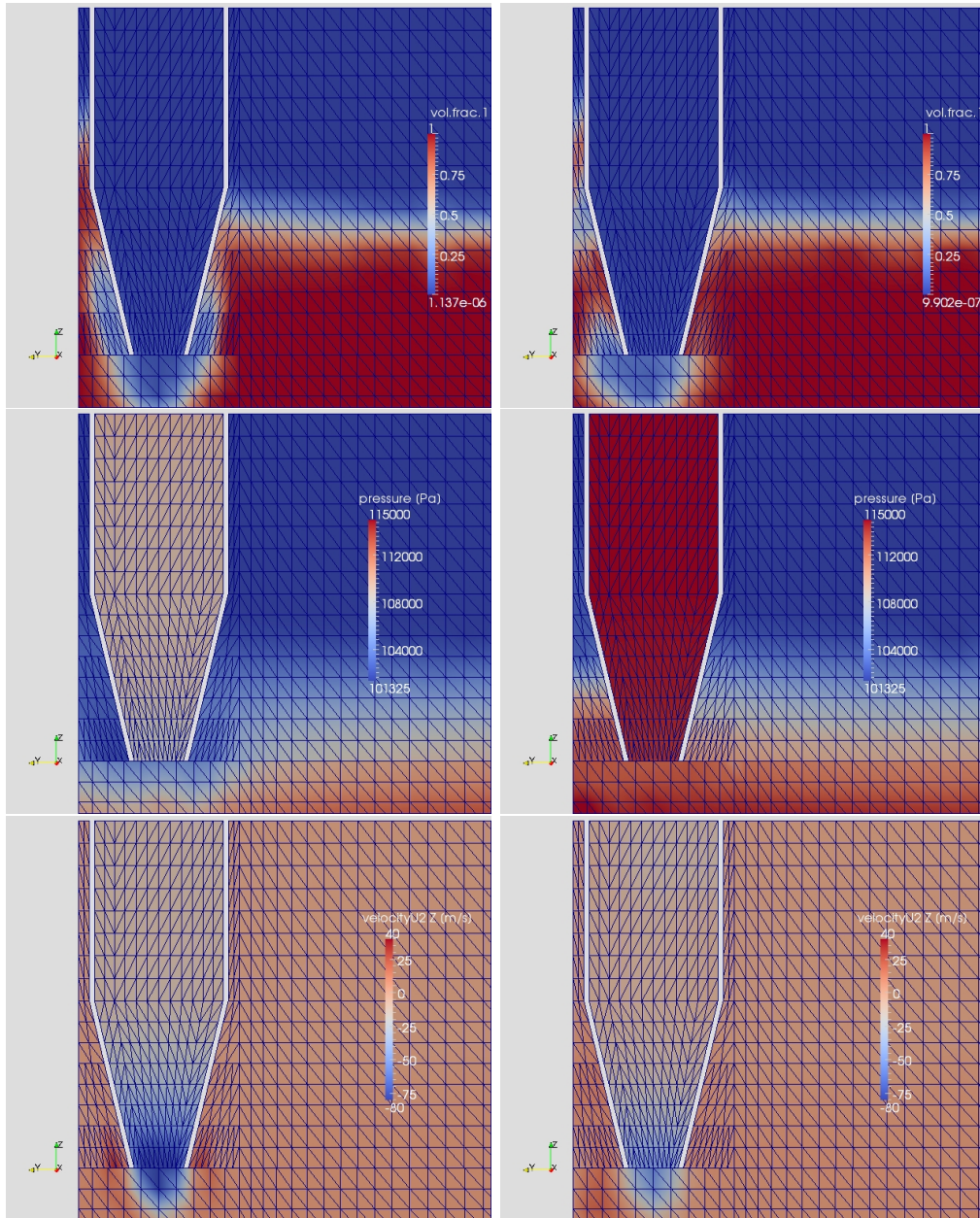



Figure 3.33: Water volume fraction (top), pressure (middle) and vertical component of steam velocity (bottom), on the vertical section through the injector axis at $x=3.12$ m and times $t_1 = 53.7734$ s (left) and $t_1 = 55.9895$ s (right)

are non conforming but, the grid size in the direction normal to the common surfaces are kept constant on both sides. The mesh has now 129338 hexahedral cells.

With the second mesh the simulation starts at time $t_0 = 451$ s, that is before the increase of the steam mass flow at the inlet takes place. The simulation appears to be more stable than with the first mesh, but a few changes were required in the user

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	52	121

defined functions in order to run the simulation up to $t_1 = 60$ s. More particularly, in the drag model `usdrag.F` the water characteristic diameter was set to $d_1 = 5 \cdot 10^{-5}$ when the water volume fraction $\alpha_1 < 2 \cdot 10^{-5}$, in order to prevent the water velocity to diverge in the region of the domain where water is not actually present. However, the pressure fluctuations due to the prompt steam condensation were still present inside the injector, and the routine `usclim.F` was modified in order to reduce this effect. From the available experimental data it is found that the pressure inside the HX pool is almost constant with a value close to $P = 113000$ Pa. Thus a correction for the steam mass flow rate Q_{steam} has been introduced in such a way that Q_{steam} is set to zero when the pressure at the inlet section exceeds $P^{up} = 115560$ Pa, on the other hand it is increased when it drops below $P^{low} = 109250$ Pa. The correction for this second case is

$$Q_{steam} = \sqrt{[Q_{steam}^0]^2 + Q_{adj}^2} \quad (3.4)$$

$$Q_{adj} = 2(P^{low} - P^{in})^* \rho A_{in}^2 \quad (3.5)$$

$$(P^{low} - P^{in})^* = \min(\text{pos}(P^{low} - P^{in}), 1) \quad (3.6)$$

where P^{in} and A_{in} are the pressure at the inlet section and its area, respectively. With this correction the simulation was run up to $t_1 = 60$ s, even if it was not possible to have a perfect control on the mass injected in the system, therefore an intermediate stage was considered.

In Figures 3.32-3.33 the evolution of water volume fraction, pressure and steam vertical velocity is presented on a section passing through the injector, at times in the interval $50.6804\text{s} < t_1 < 55.9895\text{s}$. A part of the mesh is also shown. It can be seen the oscillatory behavior of the flow close to the injector, with the pressure decreasing when the highest velocities are reached at the injector nozzle and the water is pushed away from that position. Then the velocity decreases, the water comes back closer to the nozzle and the pressure inside the injector increases again. It should be observed that the pressure field presents spurious oscillations at time $t_1 = 51.8309$ which are probably caused by the jump in the grid size in that region along the two horizontal directions. The observer should also be aware that the software PARAVIEW, which is used for data visualization, reconstructs the field on triangles when performing a cut. The wiggles along the lines that cut the squared cells with triangles may be in part due to the software itself.

With the correction in the mass flow rate the simulation run up to time $t_1 = 60$ s, but finally diverged. However, this simulation pointed out that one of the main reasons for the numerical instability of the run was the excessive stiffness of the boundary conditions together with a too small portion of the domain occupied by the steam. Another critical aspect was identified in the junction of the grid blocks, where the grid size undergoes a steep variation.

Results obtained on the geometry with a straight injector in the center of the domain and an additional volume simulating a portion of the HX pool

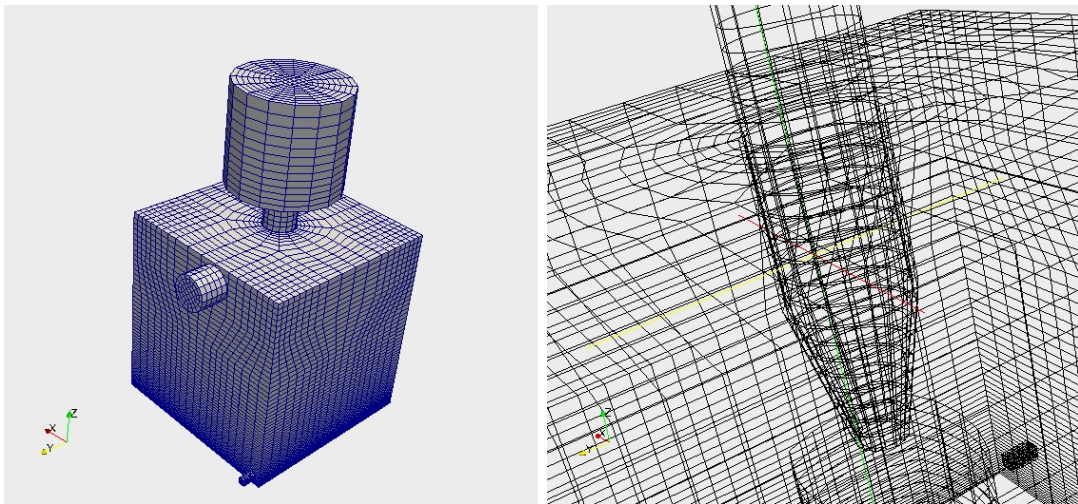


Figure 3.34: PERSEO 3D hexahedral grid (left) and a detail of the injector region (right) for the geometry with a straight injector and a modified position of the injector nozzle

In order to further reduce the sources of numerical instabilities a third geometry has been considered. In Figure 3.34 the discretized geometry is shown together with a local zoom of the injector region. Furthermore, to prevent the pressure oscillations inside the injector, a larger cylindrical vessel has been added upstream. The volume of such a vessel is approximately that occupied by the steam in the HX pool. The inlet section, where the boundary conditions from the CATHARE simulation are imposed, is now the upper surface of this volume. With this modification the pressure fluctuations inside the injector, which are due to the prompt condensation of the steam, are greatly reduced.

With the repositioning of the injector in the center of the domain, the change in the geometry is not negligible and this has an impact on the comparability between numerical and experimental data, but it allows to create a single mesh inside the pool, without non-conforming junctions or sudden changes in the grid size. Furthermore, it removes the presence of small size cells, and the smaller grid dimension is now the injector wall thickness. The grid has now 32250 hexahedral cells and with these changes larger time steps are now possible, removing in this way a possible source of instability.

The simulation has been performed with a variable time step, which is determined by several stability coefficients (CFL, CFL_α , Fourier). It should be remarked that the stability limit of these coefficients was not constant during the simulation. At the very beginning, when the steam mass flow rate at the inlet $Q_{steam} < 3$ kg/s,

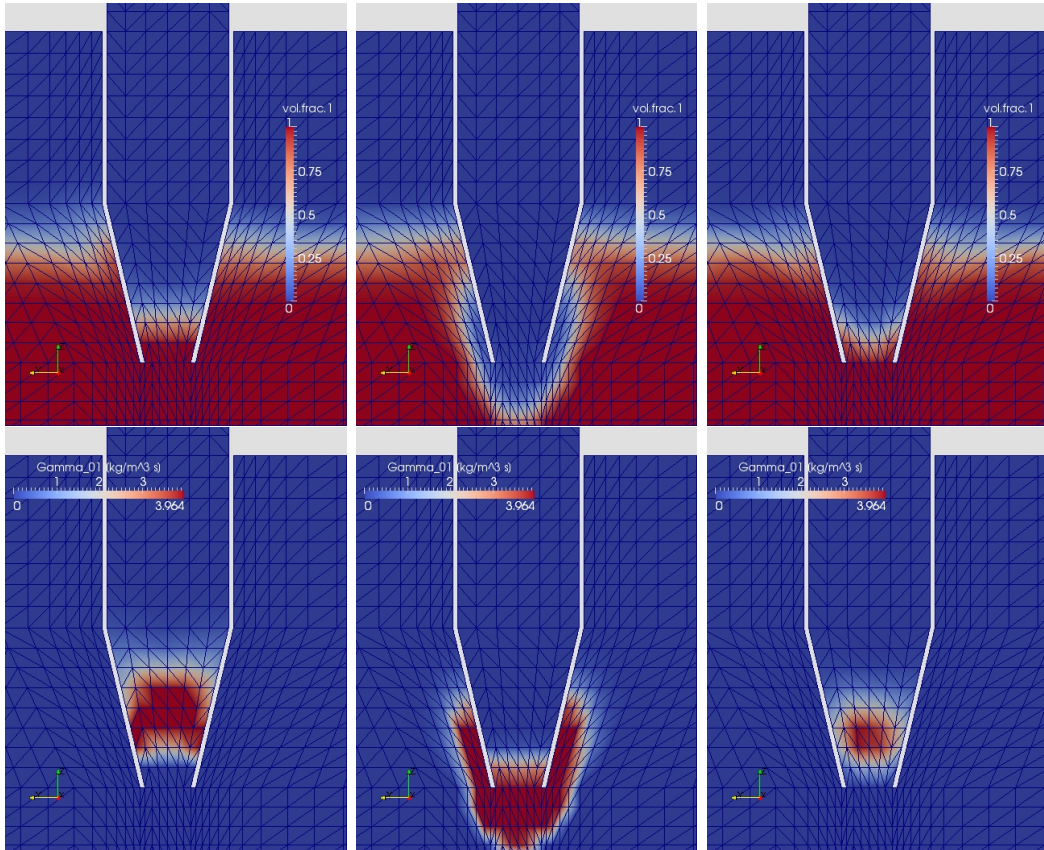


Figure 3.35: Water volume fraction (top) and condensation rate (bottom) for the geometry with a straight injector and a modified position of the injector nozzle, at times $t_1 = 12.88, 14.15, 15.18$ s (left to right)

the system undergoes very large fluctuations: the steam condensation takes place inside the injector until the pressure in the injection system is high enough to push the water out of the injector. This fluctuating behavior can be seen in Figure 3.35 where the water volume fraction and the steam condensation rate are both shown in the time interval $12.88s < t_1 < 15.18s$. Under these severe conditions the stability limit on the CFL_α condition had to be kept in the range (0.15-0.2). This is probably due to the fact that the phase front moves suddenly and at a high velocity and the time step must be tailored on this key feature of the flow. As the mass flow rate at the inlet is increased, the steam velocity is higher and the water front never comes back inside the injector, as shown in Figure 3.36. Steam condensation mostly takes place at the exit of the injection system or outside of it.

At the same time, the high steam velocity at the exit and the small grid dimension below the injector wall require a very small time step. If the stability limit is left at $CFL_\alpha \sim 0.2$, the time step is excessively small and the calculation diverges in the α -P-H loop. For this reason the stability limits have been set to $CFL = CFL_\alpha \sim 0.5$ and $Fourier = 1.0$.

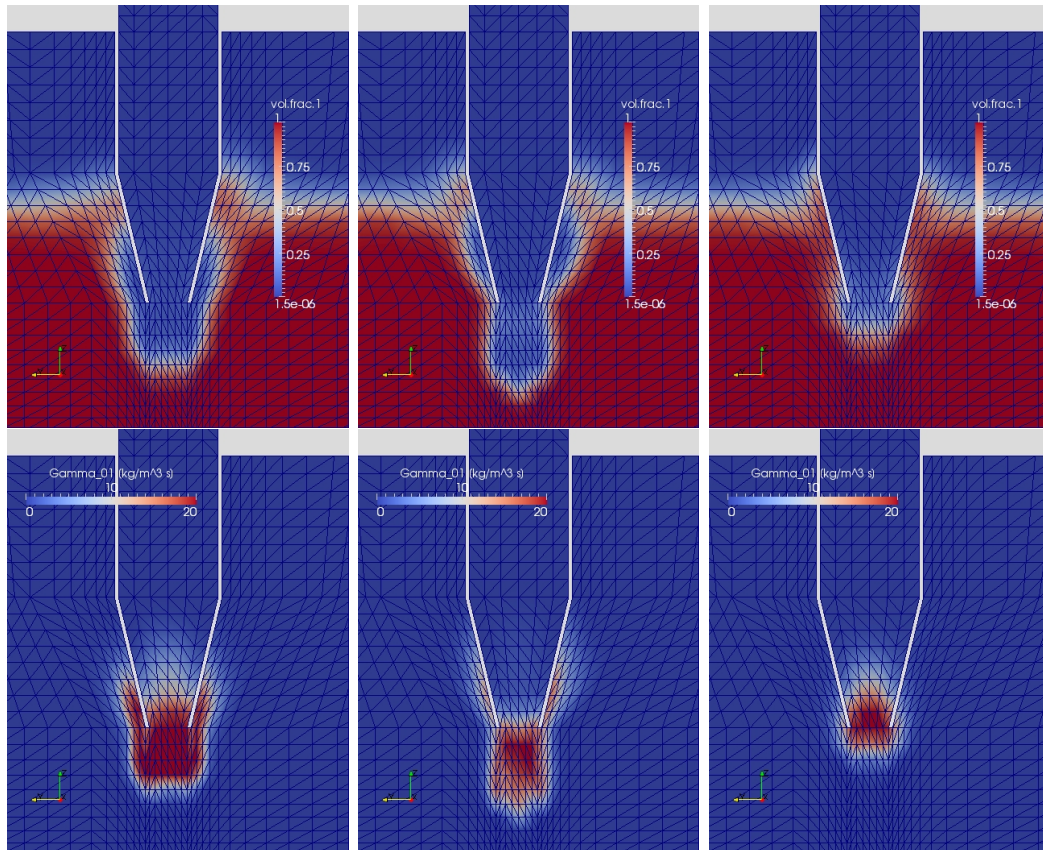


Figure 3.36: Water volume fraction (top) and condensation rate (bottom) for the geometry with a straight injector and a modified position of the injector nozzle, at times $t_1 = 200.6, 202.97, 205.52$ s (left to right)

The velocity field for $t_1 = 200.6$ s is shown in Figure 3.37. From the vertical component of the steam velocity it can be seen that the steam flowing out of the injector can reach the velocity $U_{2z} = -188$ m/s, while the maximum velocity of the flow going up outside the injector is $U_{2z} = 74.55$ m/s. As a consequence the radial component of the velocity reaches its highest value right below the end of the injector wall.

The water temperature field evolution in the time interval $200.6 \text{ s} < t_1 < 243.36$ s is presented in Figure 3.38 together with the corresponding water volume fraction. It can be seen how the high-temperature region below the injector does not extend far below the water/steam front. On the contrary, a possible temperature stratification due to the presence of the wall can be observed near the pool lateral boundary. The reason may become clear by looking at Figure 3.41. There is a net transfer of vertical momentum to the water in the upward direction due to buoyancy effects because of the heated water and to friction on the steam upflow near the external wall of the injector. A circulation is therefore setup inside the pool, with the fluid moving upwards in the central part and downwards close to the lateral walls.

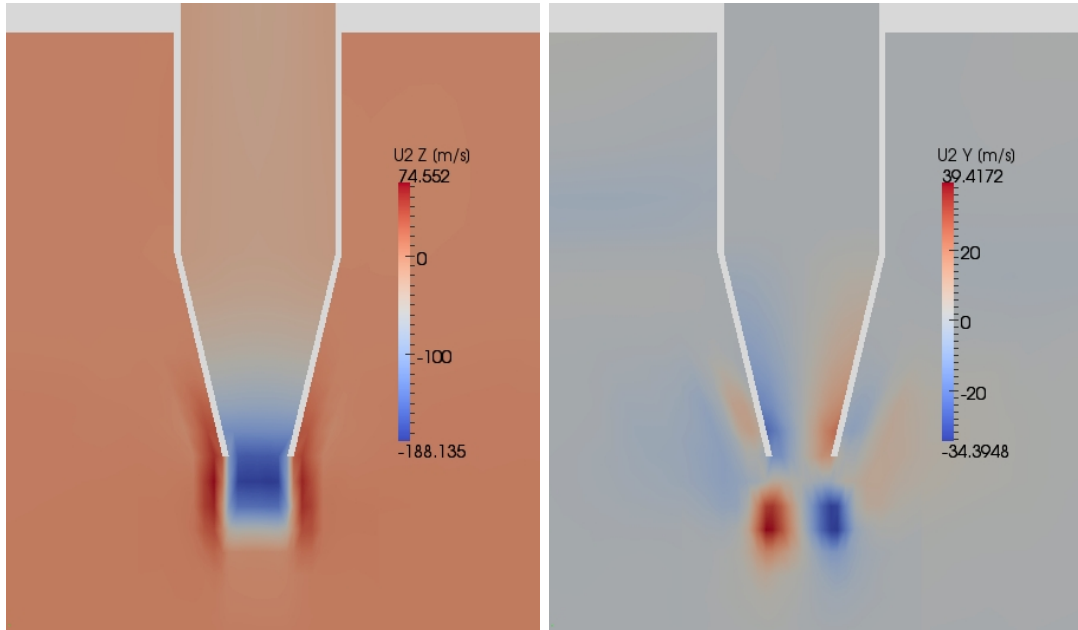


Figure 3.37: Vertical (left) and radial (right) steam velocity components on the vertical section through the injector axis at $x = 2.75$ m and time $t_1 = 200.6$ s, for the geometry with a straight injector and a modified position of the injector nozzle

However, in the real configuration the injector is positioned close to the wall and buoyancy and wall effects will then overlap. Because of this consideration we should reconsider the experimental data and verify if the assumption that the temperature stratification, at least in this first part of the transient, is mainly due to a wall effect is correct. With reference to Figure 3.39 we can consider two couples of probes. The first couple, TP006 and TP008, are positioned below the injector while the second one, TP023 and TP025, are located towards the pool center. The two probes TP006 and TP023 positioned at 2.195 m from the pool bottom, the other two at 1.195 m. There is clearly a temperature difference between the two probes located below, which is not present in the records of the two probes positioned at the pool center, at least in the first part of the transient. It should also be noted that the probes below the injector undergo a very fast temperature rise at the beginning of the transient, while the temperature rise is much more gradual for the probes in the pool center. A consistent temperature stratification seems to be present for $t > 2000$ s, when both the probes at the higher level register a temperature higher than that of the lower probes.

In Figure 3.40 is shown the comparison between numerical and experimental data. The experimental data refer to probes TP006, TP023 and TP025. The numerical data are taken at the same position of TP006 (TP6_num) and below the injector at the same vertical level as TP006 and TP023 (TP23_N_num). The exact position for the latter numerical probe is given, in the simulation frame of reference, by the following coordinates $x = 2.815$ m, $y = 2.7355$ m, $z = 2.195$ m. It can be

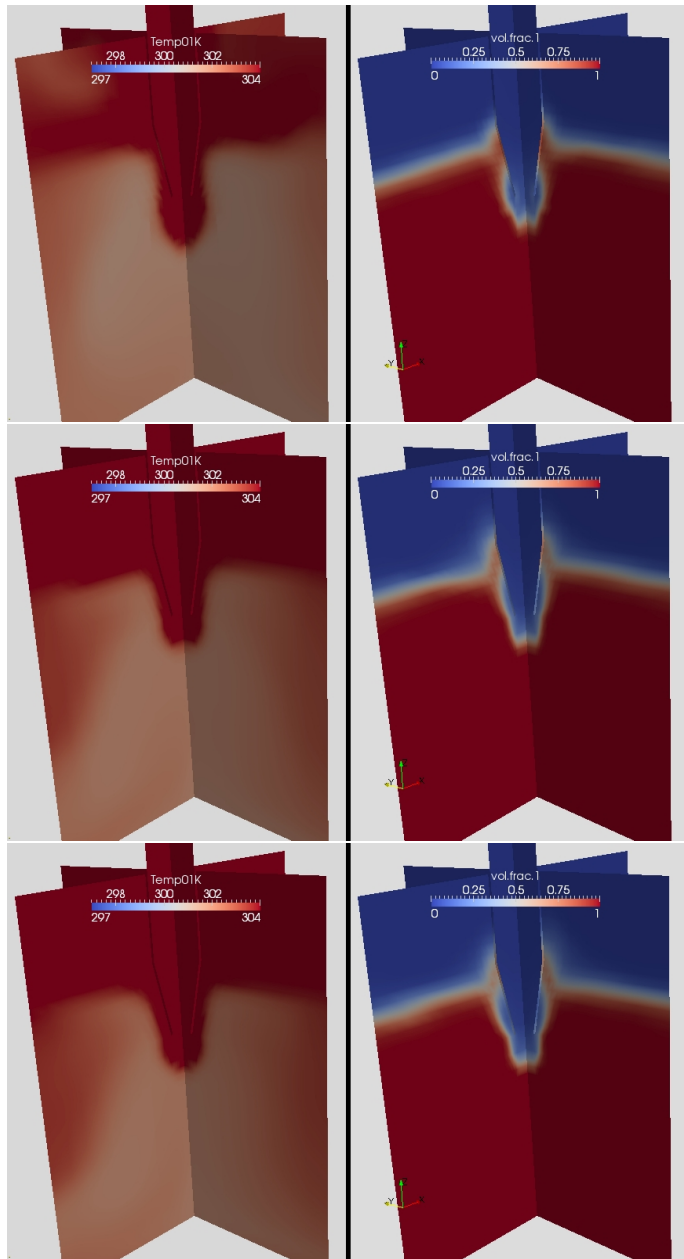


Figure 3.38: Water temperature (left) and volume fraction (right) on the vertical sections through the injector axis at $x = 2.75$ m and $y = 2.75$ m and times in the range $200.6 \text{ s} < t_1 < 243.36 \text{ s}$, for the geometry with a straight injector and a modified position of the injector nozzle

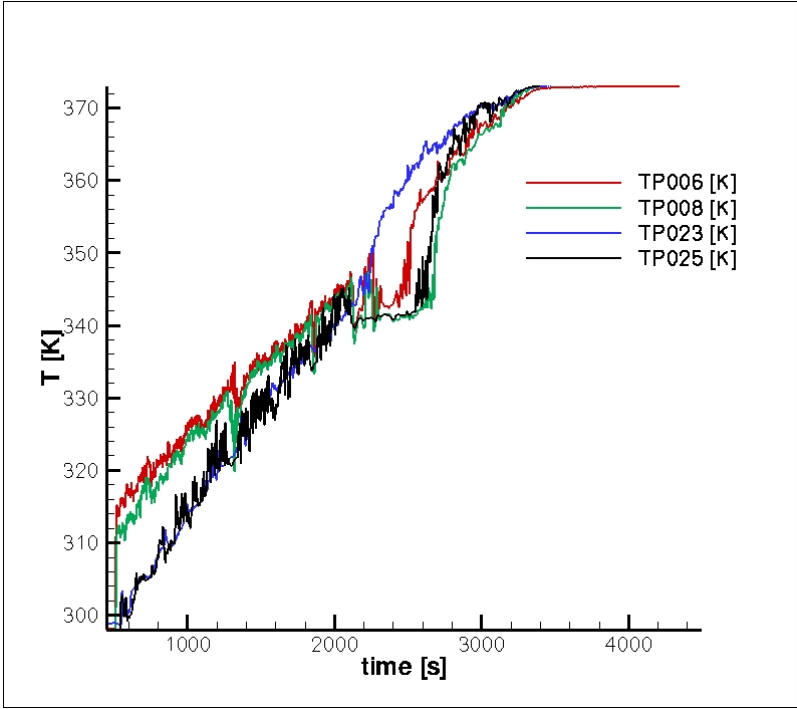


Figure 3.39: Experimental water temperature evolution for different probes

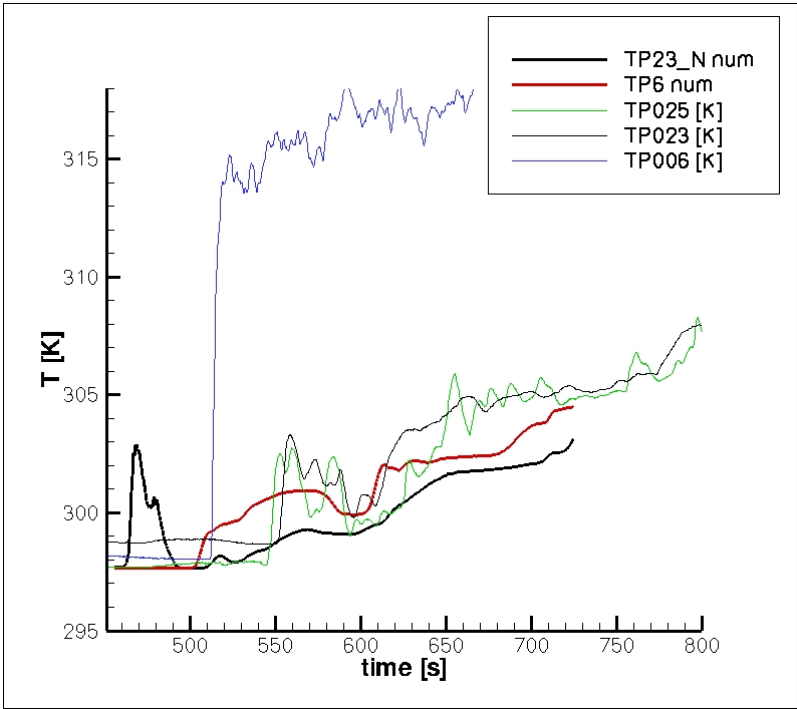


Figure 3.40: Comparison between experimental and numerical water temperature data for different probes

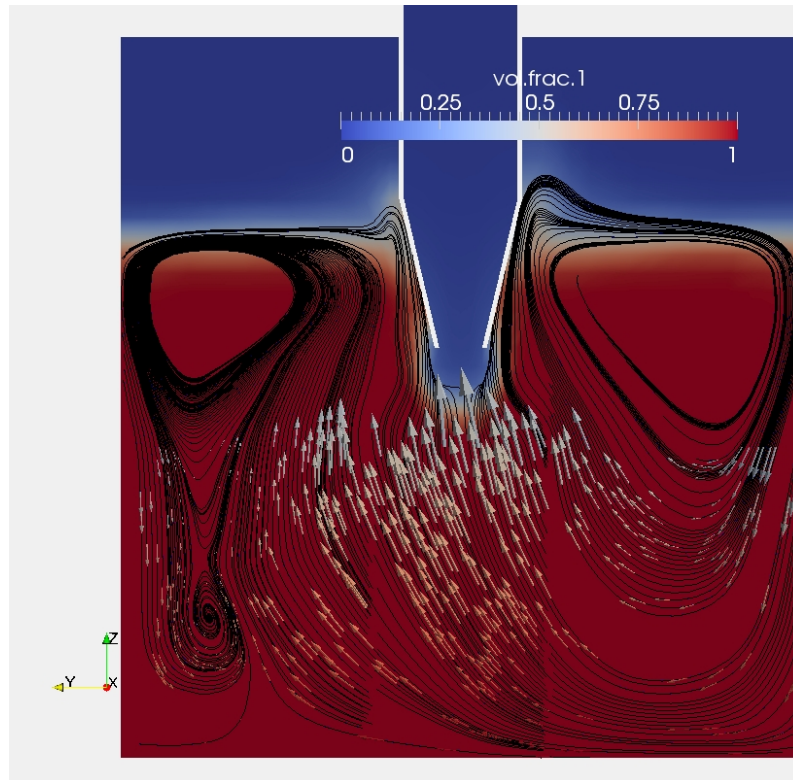


Figure 3.41: Water streamlines and velocity vectors (lower part only) on plane $x = 2.75$ m for $t_1 = 213.28$ s. Contour for water volume fraction.

seen how the numerical data follow quite closely the experimental data taken from the probes at the pool center (TP023 and TP025). The fact that the experimental data for TP006 show a completely different behavior, with a sudden temperature rise up to $T \simeq 315$ K seems to confirm the coupled effect of the injection and wall effects. The comparison between the two numerical series, with TP6_num always above TP23_N_num, shows the presence of a wall effect. It should be considered that in the real case the injector is close to the lateral wall, the hot water below the injector moves immediately downwards, before it can exchange heat with the surrounding colder water. This may explain the step rise in temperature evolution at TP006 which can be seen in the experimental data.

All the above conclusions should be verified in a future work, where the real geometry will be simulated. In addition a grid convergence analysis should also be made. This has not been done in the present work due to time constraint. The scalability problems that will be discussed in section 3.4.5 made it impossible to perform such an analysis in a reasonable time.

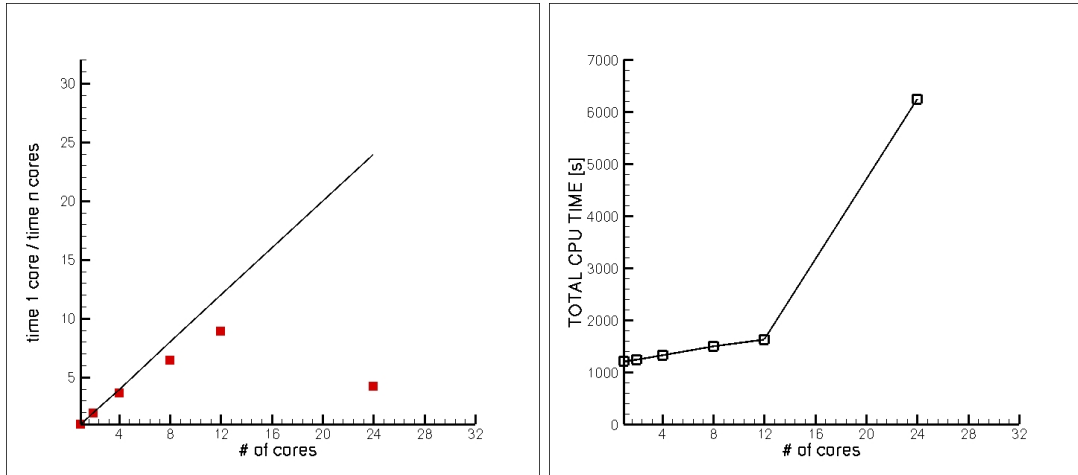



Figure 3.42: Parallel performance of the NEPTUNE code on the ENEA/CRESCO system for the PERSEO geometry with the modified position of the injector

3.4.5 Computational costs

An analysis of the parallel performance of the implementation of the NEPTUNE_CFD code on the ENEA/CRESCO_Casaccia system has been done in order to test the implementation itself. In Figure 3.42 the results of such an analysis are shown. They refer to the simulation performed on the geometry with the modified position of the injector and 100 time steps were taken into account. In this system each node contains two Six-Core Opteron 2427 processors, therefore a total of 12 cores per node are available.

A scalability plot is presented on the left of Figure 3.42. It is clear that the parallel performance of the code is reasonable until the number of cores is less than 12, that is as long as the calculation is done on a single node. It should be recalled that for the third case the mesh contains 32250 computational cells. The performance then dramatically decrease when 24 cores are used, showing an increase of the time required to perform the simulation with respect to the case with either 8 or 12 cores. The lost in efficiency is even more clear on the right of the figure where the total CPU time is shown. While only a slight increase in the total cost of the simulation is present in the range from 1 to 12 cores, a step increase can be observed when two nodes are involved. After some investigations it turned out that the LAM/MPI libraries, which were used to compile the code, can not support the Infiniband Network system of the CRESCO architecture.

In order to obtain a full scalable implementation of the NEPTUNE_CFD code on the ENEA/CRESCO system it is therefore necessary to compile the code with the OpenMPI libraries. For this reason it has been agreed with the CRESCO administration that, should a new release of the NEPTUNE_CFD code become available to ENEA, the developers will try to install it directly on the system, instead of compiling it on different architectures.

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	61	121


Validation of the TRIO_U code with heterogeneous nucleate boiling from a single site

4.1 Introduction

4.1.1 Experimental data

This section is devoted to a simple validation of the TRIO_U code by analyzing the growth and detachment of bubbles from a single nucleation site. This section can be considered as a follow-up step to the work presented in [3]. This approach is the basic step in the simulation of nucleate boiling and can be considered as the starting point in the study of complex heat flux mechanisms such as the Departure from Nucleate Boiling (DNB). Most of the published work in this area are empirical correlations based on experimental data and physical models which rely on simplified balances among the different forces acting on the bubble. Different numerical approaches, which are based on the Lattice Boltzmann Method (LBM) and finite-difference discretization of the macroscopic conservation equations, have been proposed more recently in the literature. However, most of the published work considers a two-dimensional geometry. In this work we present three-dimensional simulations of the nucleate boiling process performed with the TRIO_U code. We follow in time the growth of the bubble up to its departure and measure its diameter and the frequency of the process. In particular we analyze the dependence of the departure diameter on a number of physical quantities which appear in several correlations, such as gravity, surface tension and contact angle. The preliminary results we have obtained are in agreement with the experimental correlations and the two-dimensional LBM numerical results.

The heat transfer process in convective nucleate boiling is used in many industrial applications where a large amount of heat needs to be removed from a relatively small area. For a given amount of heat flux and flow rate, the heat transfer coefficients in nucleate boiling are much higher than in single-phase forced convection. Therefore the former needs much lower bulk velocities and hence lower pressure drops to remove and transport the same amount of heat.


	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	62	121

Nucleate boiling is a very complex phenomenon even when it is limited to the boiling cycle on a single nucleation site with well-separated events. In fact even in this simple case a detailed understanding is required of the activation process of nucleation sites, of the growth of the bubble, characterized by the presence of several interacting volume and surface forces during the liquid phase change, and of the detachment of the bubble, which involves large interfacial distortion and topology adjustment as well. A detailed analysis of local processes is an important issue at high heat fluxes, where the departure from nucleate boiling (DNB) is the principal physical mechanism to determine the maximum heat flux, or critical heat flux (CHF), that can be obtained by nucleate boiling in safe conditions.

In this work we investigate the bubble diameter at departure and the frequency of the bubble cycle in isolated bubble regimes. The frequency of the bubble cycle is defined as the inverse of the sum of the waiting period, necessary to rebuild a superheated layer of liquid after the bubble release, and the growth period of the bubble. The bubble diameter at departure can be estimated with a number of correlations that usually consider a balance among the forces that tend to hold the bubble to the wall and those that tend to detach the bubble from the solid surface. In isolated bubble regimes a number of mechanism-based correlations have been developed. In these models the transient heat conduction in the liquid close to the heater surface is an important mechanism for the heat transfer process. The bubble acts like a pump that removes superheated liquid near the solid surface replacing the bubble volume with cold liquid after its departure [35].

Several empirical correlations, based on experimental data and physical analogies, have been proposed in the past. These correlations have often been used in engineering applications even if usually they have a limited range of applicability especially in the investigation of new geometrical configurations [32, 33, 34]. More recently the direct numerical simulation (DNS) has become an important tool to investigate the bubble growth and departure in the partial nucleate boiling regime. A few models divide the geometrical domain into micro- and macro-regions. The micro-region is the thin liquid layer that forms between the solid surface and the evolving liquid-vapor interface. The macro region is the rest of the domain. Different approximations of the conservation equations and transport mechanisms are used in these two regions [36].

The interface is usually represented by an implicit formulation or front capturing technique. The level set approach is very popular in this field, where the interface is described by the zero level set of a smooth distance function ϕ , which is advected by the flow [37]. Another approach considers the color function C which is the discrete version of the discontinuous characteristic function χ , equal to one in the reference phase and zero in the secondary phase. Numerically speaking, the color function is the fraction of each computational cell which is occupied by the reference phase [38]. The lattice Boltzmann method (LBM) has also been used by many authors to study two-phase flows at micro-scale level both in isothermal conditions or with heat transfer [39, 40].

 Ricerca Sistema Elettrico	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	63	121

Most of the papers that have been published are two-dimensional studies. In this chapter we consider the three-dimensional TRIO_U code, developed by CEA in Grenoble. The code solves the single-fluid formulation of the Navier-Stokes equations. The Navier-Stokes system is then coupled with an advection-diffusion energy equation for the liquid phase while the vapor is considered at saturation temperature. A front-tracking technique is used to represent the interface, which is defined by a Lagrangian mesh made up by triangles. In this study we measure the bubble diameter at departure and compare it with correlation data and results from other numerical approaches. Preliminary results of the frequency of the bubble cycle are also presented.

In the last decades many empirical and mechanistic correlations have been developed for pool nucleate boiling and have been recently reviewed by Dhir [41]. In the following we mention a few of them which are relevant to this study.

The first correlation for pool nucleate boiling was derived by Rohsenow [32] who reduced the problem to a single-phase forced convection problem, where the heat is removed from the wall by the liquid motion caused by the bubble detachment and rise. The data are correlated as

$$\frac{Re_b Pr_l}{Nu_b} = C_s Re_b^m Pr_l^n, \quad (4.7)$$

where m and n are empirical exponents and the constant C_s is related to the fluid and solid system. The characteristic length in the Reynolds number Re_b and in the Nusselt number Nu_b is the bubble diameter at detachment. The velocity in Re_b is the surface vapor velocity computed with a simple energy balance.


The correlation proposed by Cooper [34] for a flat plate is based on an extensive study of many experimental data

$$\frac{q^{1/3}}{\Delta T_w} = 55 P_r^{(0.12-0.4343 \ln(R_w))} (-0.4343 \ln(P_r))^{-0.55} M^{-0.5}. \quad (4.8)$$

In this equation R_w is the surface roughness in microns.

While all of the correlations indicate that the wall heat flux varies with the third power of ΔT_w , the dependence on other physical parameters, such as gravity, contact angle, and surface roughness, is quite different among the different correlations, if not present at all.

In the isolated bubble regime the heat flux for partial nucleate boiling can be written as the sum of three terms, which identify the contributions from the transient conduction around nucleation sites, the micro-layer evaporation under the bubbles during their growth, and the natural convection on inactive areas [42]. Furthermore, a power law of the nucleating cavity diameter has been proposed for the density of nucleation sites on the heater surface [43]. Here we are mainly concerned with the proposed correlations for the bubble diameter at departure and the bubble release frequency.

 Ricerca Sistema Elettrico	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	64	121

Bubble diameter D_b at departure. Among the many correlations presented in the literature we consider

$$D_b = 0.0208 \phi \sqrt{\frac{\sigma}{g(\rho_l - \rho_v)}}, \quad (4.9)$$

proposed by Fritz [44]. This correlation was derived from a balance between buoyancy, which tends to lift the bubble away from the solid surface, and surface tension, which sticks the bubble on the wall. Deviations from this expression are reported at high pressure. In general it is found that the different correlations are not consistent with each other with respect to the independent variables. For example, in some experiments it is found that the bubble departure diameter increases with wall superheat, in others it decreases. The difficulty to derive a generalized correlation is also related to the fact that the evolution of the temperature and velocity fields, which vary both in time and space, have an influence on the growth rate and the forces acting on the bubble.

Bubble release frequency f . This frequency is defined as the sum of the waiting and growth periods. The waiting time depends on the temperature distribution in the solid and in the liquid near the nucleation site. The growth time depends on the evaporation rate in the thin layer between the wall and the bubble and around the bubble as well, until the bubble reaches the departure diameter. Here we consider Zuber's relation [45]

$$f D_b = 0.59 \left(\frac{\sigma g(\rho_l - \rho_v)}{\rho_l^2} \right)^{1/4}. \quad (4.10)$$

Many correlations contain the product $f D_b$, hence the problems in the estimate of the departure diameter D_b combine with the difficulties in the computation of the waiting and growth periods.


4.1.2 The governing equations

In the following simulations it is assumed that the density of the fluid is not dependent on the pressure and only slightly on the temperature. The assumption of constant physical property is assumed for the dynamic viscosity, the thermal conductivity, the specific heat capacity and the surface tension coefficient. Buoyancy effects are taken into account only in the gravitational force term. This assumption is known in the literature as the Boussinesq's approximation. The gravitational force is written for a non isothermal fluid as

$$\mathbf{F}_m = \rho_0 \beta_T \mathbf{g} (T - T_0). \quad (4.11)$$

The incompressibility assumption, which is based on small temperature variations, implies that the static pressure has no influence on the flow field. Thus a reduced pressure is defined which includes the static pressure as well

$$P_0 = \frac{P}{\rho_0} - gz, \quad (4.12)$$

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	65	121

and is used together with the velocity \mathbf{u} as one of the independent variables in the Navier-Stokes equations. By using these simplifications and by assuming that the flow is laminar, the following conservation equations are solved by the TRIO_U code

$$\begin{aligned} \nabla \cdot \mathbf{u} &= 0, \\ \frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u}\mathbf{u}) &= -\nabla P_0 + \nabla \cdot \boldsymbol{\tau}^0 + \mathbf{F}_m^0 + \mathbf{S}_m^0, \\ \frac{\partial T}{\partial t} + \nabla \cdot (\mathbf{u}T) &= -\nabla q^0 + S_{th}^0. \end{aligned} \quad (4.13)$$

This system of partial differential equations is given by the mass and momentum conservation equations and the conservation of internal energy, respectively. In the system (4.13) we define the viscous stress tensor, the buoyancy force and the momentum source term as

$$\boldsymbol{\tau}^0 = \frac{\mu}{\rho_0} (\nabla \mathbf{u} + \nabla \mathbf{u}^T), \quad (4.14)$$

$$\mathbf{F}_m^0 = \beta_T \mathbf{g} (T - T_0), \quad (4.15)$$

$$\mathbf{S}_m^0 = \mathbf{S}_m / \rho_0. \quad (4.16)$$

We also consider the Fourier's law and the energy source term in the following form

$$q^0 = -\lambda \nabla T / (\rho_0 c_p), \quad (4.17)$$

$$S_{th}^0 = S_{th} / (\rho c_p). \quad (4.18)$$

We point out that in a two-phase flow the momentum source term \mathbf{S}_m^0 contains the capillary force located at the interface between the two fluids.

The numerical model used in TRIO_U to discretize the set of equations (4.13) is the finite volume method. This set of conservation equations is written in a conservative form and is integrated over a control volume Ω bounded by the surface Γ . By applying Gauss' theorem we obtain the following system of equations which is valid for laminar flow

$$\begin{aligned} \int_{\Gamma} \mathbf{u} \cdot \mathbf{n} \, d\Gamma &= 0, \\ \int_{\Omega} \frac{\partial \mathbf{u}}{\partial t} \, d\Omega + \int_{\Gamma} \mathbf{u}\mathbf{u} \cdot \mathbf{n} \, d\Gamma &= \int_{\Gamma} \nu \nabla \mathbf{u} \cdot \mathbf{n} \, d\Gamma + \int_{\Gamma} P_0 \mathbf{n} \, d\Gamma + \int_{\Omega} (\mathbf{F}_m^0 + \mathbf{S}_m^0) \, d\Omega, \\ \int_{\Omega} \frac{\partial T}{\partial t} \, d\Omega + \int_{\Gamma} \mathbf{u}T \cdot \mathbf{n} \, d\Gamma &= \int_{\Gamma} \alpha \nabla T \cdot \mathbf{n} \, d\Gamma + \int_{\Omega} S_{th}^0 \, d\Omega. \end{aligned} \quad (4.19)$$

Here \mathbf{n} denotes the normal unity vector of the Γ surface which is oriented outwards from the reference phase. The finite volume technique implemented in TRIO_U needs a structured mesh of the computational domain and staggered grids for the velocity and pressure fields. In particular, scalar fields, such as pressure and temperature, are defined in the center of the cell while the velocity normal components

are located in the midpoint of a cell face. In the integral formulation (4.19) the fluxes across each face of a control volume are calculated in a different way according to the type and order of the discretization scheme. We have used a conservative second-order central scheme for the diffusive term and a QUICK scheme for the nonlinear advection term.

The Navier-Stokes equations are integrated in time by the SOLA algorithm that separates the computation of pressure from that of velocity. It can be viewed as a fractional-step algorithm. Starting with a velocity field \mathbf{u} that satisfies the divergence constraint, this algorithm computes the pressure field P , by solving a Poisson's equation, and the associated time derivative $\partial\mathbf{u}/\partial t$ in such a way that the divergence-free condition of \mathbf{u} holds at the next time step. In the explicit version of the SOLA algorithm, the pressure is taken implicit in the momentum equation and the velocity is taken implicit in the continuity equation. All the other terms are explicit. Therefore we have

$$\begin{aligned}
\nabla \cdot \mathbf{u}^{n+1} &= 0, \\
\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \nabla \cdot (\mathbf{u}^n \mathbf{u}^n) &= -\nabla P_0^{n+1} + \nabla \cdot \boldsymbol{\tau}^{0,n} + \mathbf{F}_m^{0,n} + \mathbf{S}_m^{0,n}, \\
\frac{T^{n+1} - T^n}{\Delta t} + \nabla \cdot (\mathbf{u}^n T^n) &= -\nabla q^{0,n} + S_{th}^{0,n}.
\end{aligned} \tag{4.20}$$

In these simulations a front-tracking method is used to track the fluid interface. The front tracking method considers the fluid interface as being the boundary of a moving volume over a fixed domain. The boundary between the fluids can be represented by a set of connected marker points advected by the velocity field. When the interface is tracked by marker points two different grids are usually employed: a fixed grid, over which the governing equations are solved, and the lower-dimensionality grid that tracks the interface during its motion. In Figure 4.43 we show the unstructured mesh representing the interface which is made up by triangles.

The Lagrangian mesh is advanced by a velocity field that is the solution of the Navier-Stokes equations on the fixed grid. As the front moves, the grid values of the fluid properties are updated. The speed of the displacement of the nodes of the interfaces is equal to the velocity at the center of the Eulerian element in which each node is located at the beginning of the time step. The movement of the mesh defining the interface is taken into account by using the characteristic method where the new position is computed by integrating the velocity field along the path. When a collision between two different interfaces occurs or when the interface pinches and finally breaks, it is necessary to build a new mesh. This new mesh defines the inner region and the outer region with respect to the interface by using a level set approach. For this purpose, the Juric level-set reconstruction algorithm has been implemented in the TRIO.U code to allow topology changes in an automatic fashion, without the need to prescribe the physical break-up/coalescence mechanism of the interface [46]. The breaking of the mesh is based only upon geometrical considerations related to the relative distance between the two bubbles. In particular we have chosen the

indicator function as a source field to compute the level set function needed by the Juric algorithm.

4.2 Numerical results

4.2.1 Set up of the TRIO_U code

This section is devoted to extend the validation of the TRIO_U code initiated in [3]. In this section we propose to study the three-dimensional bubble cycle for heterogeneous nucleate boiling from a single site in pool boiling configuration. The two-phase model of the TRIO_U code is based on the Navier-Stokes equations coupled with the phase advection equation. Since the phases are assumed to be immiscible the knowledge of the spatial distribution of each phase is related to the tracking of the interface. The location of the interface is defined by a set of markers that are advected by the velocity field [20, 21, 22]. An example of the unstructured Lagrangian grid that defines the fluid interface is shown in Figure 4.43.

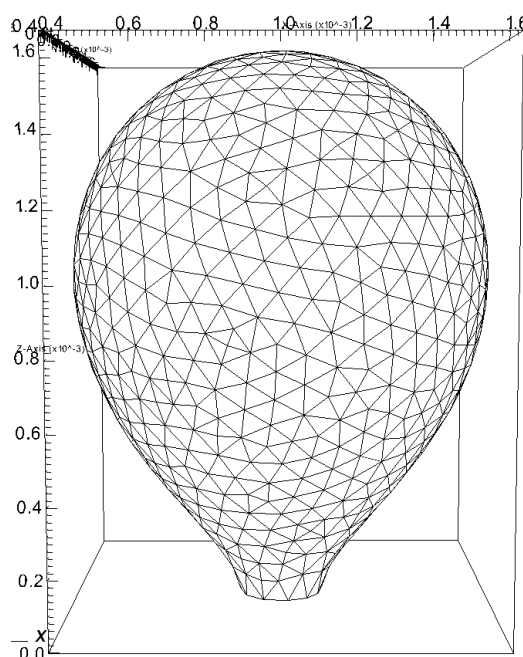


Figure 4.43: Unstructured Lagrangian grid that defines the fluid interface

In order to start the TRIO_U code on CRESCO-ENEA GRID we set the TRIO_U environment, the mesh grid and the input file with the physical and numerical parameters. The TRIO_U platform is located on CRESCO-ENEA GRID in the directory

`/afs/enea.it/project/fissicu/soft/Triou`

Once the bin directory is in the PATH environment variable, all the programs of the platform can be launched from anywhere. The command to start the TRIO_U application is

```
$ triou
```

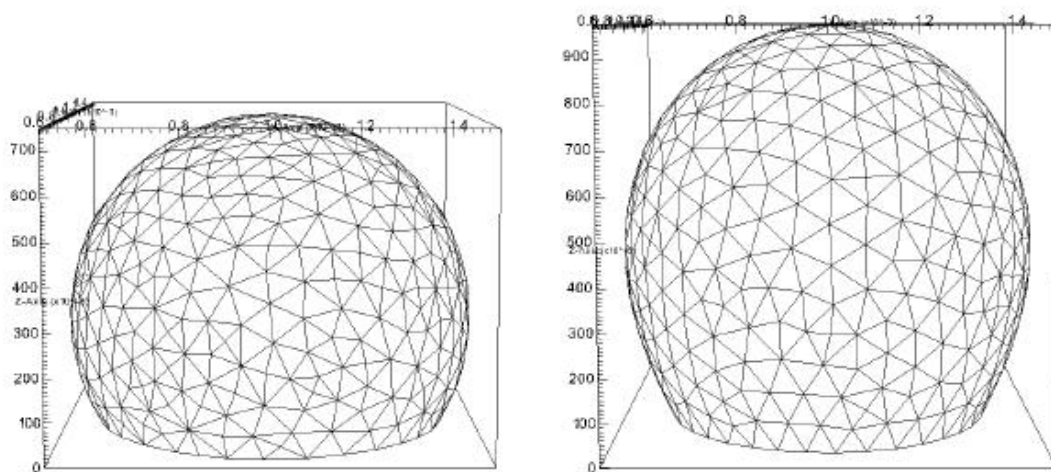


Figure 4.44: Interface evolution: initial bubble growth

In order to start the TRIO_U code a dataset file must be created. This file contains all the parameters values and all the options. The options are introduced in the code through keywords. We are planning to consider a bubble in a simple three-dimensional Cartesian domain with right hexahedral cells. Since this is a very simple domain we generate the mesh with the TRIO_U internal generator. In order to do this we need to include the commands for mesh generation in the data file. These commands are

```
dimension 3
domaine domain
# Mesh description #
Mailler domain
{ pave pave1
{
origine 0. 0. 0.
longueurs 1. 1. 1.
nombre_de_noeuds 51 51 51
}
{
bord paroi    X = 0.      0. <= Y <= 1. 0. <= Z <= 1.
bord haut     Z = 1.     0. <= X <= 1. 0. <= Y <= 1.
```

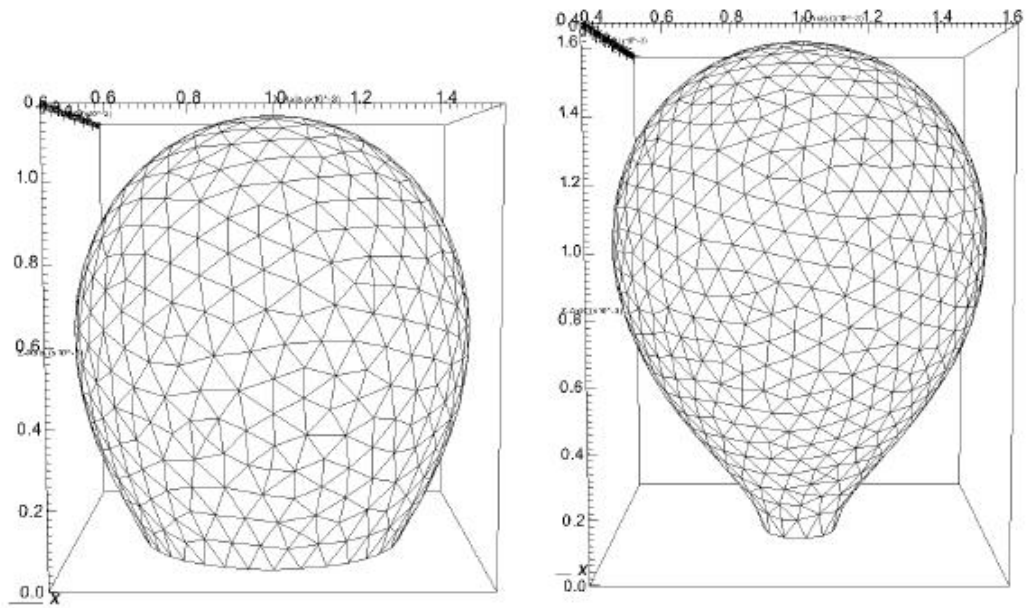


Figure 4.45: Interface evolution: bubble growth

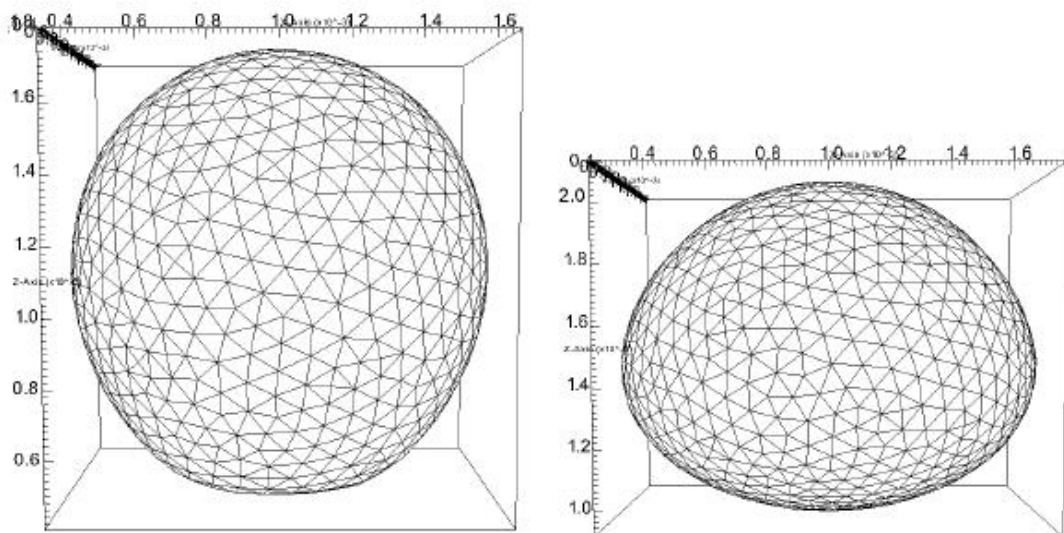


Figure 4.46: Interface evolution: detachment

```

bord bas      Z = 0.      0. <= X <= 1. 0. <= Y <= 1.
bord paroi   X = 1.      0. <= Y <= 1. 0. <= Z <= 1.
bord paroi   Y = 0.      0. <= X <= 1. 0. <= Z <= 1.
bord paroi   Y = 1.      0. <= X <= 1. 0. <= Z <= 1.
}
}
transformer dom x*0.002-0.001 y*0.002-0.001 z*0.002

```

The first line indicates the use of a three-dimensional coordinate system. The domain is described in the `Mailler` block which consists of two blocks enclosed by curly brackets. The first block describes the origin coordinates, the domain dimensions and the number of nodes. The second block defines the geometrical boundary and its labeling. The domain is then transformed with the command `transformer` to fit the dimension of the bubble. The properties of the fluid are assigned inside the same file. We have three blocks: fluid, gas and fluid-gas. These blocks are reported below

```

Fluide_Incompressible eau
Lire eau
{
  mu champ_uniforme      1 70.e-6
  rho champ_uniforme     1 610
  lambda champ_uniforme  1 461.e-3
  cp champ_uniforme      1 8270
}

```

```

Fluide_Incompressible air
Lire air
{
  mu champ_uniforme      1 22.e-6
  rho champ_uniforme     1 100
  lambda champ_uniforme  1 111.e-3
  cp champ_uniforme      1 11000
}

```

```

Fluide_diphasique fluide
Lire fluide
{
  fluide1 eau
  fluide0 air
  sigma constant          0.002
  chaleur_latente constant -50000
}

```


The properties of water, steam (gas) and the water-steam interface can be changed according to the problem. For details one can see [19, 3]. The code follows the time evolution of the velocity and temperature fields by integrating the system of equations (4.19). The interface is described by a deformable mesh which evolves inside the domain as shown in Figures 4.44-4.46.

4.2.2 Bubble release diameter for heterogeneous nucleate boiling from a single site

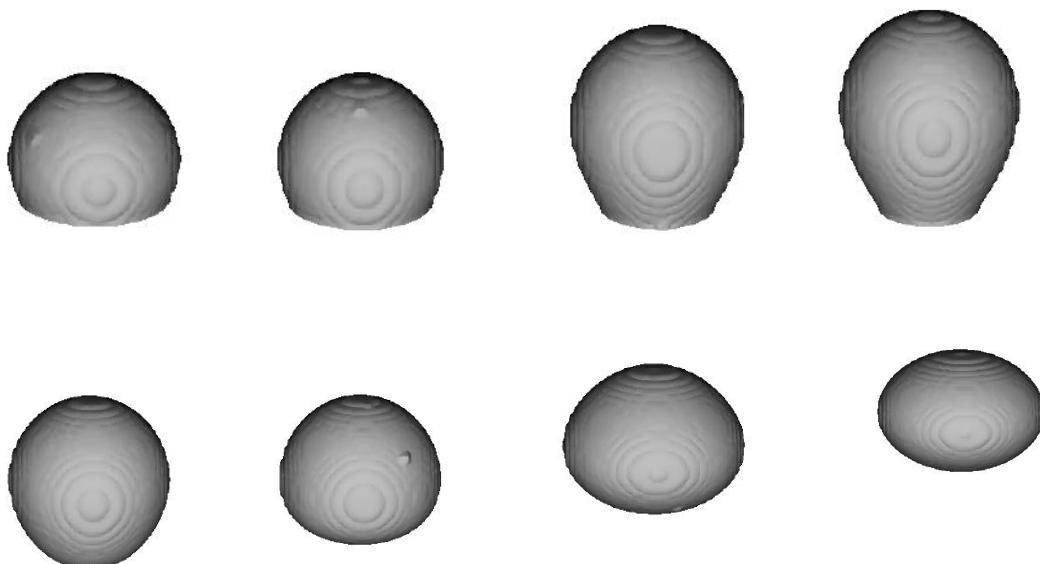



Figure 4.47: Simulation of bubble detachment from a wall

The numerical simulations are carried out in a three-dimensional Cartesian domain discretized by a structured grid with cubic cells. The simulation starts with the initial condition of a simple steady state. The velocity field is set to zero everywhere, while the temperature in the liquid is constant and equal to the saturation temperature of the vapor phase. During the evolution the temperature of the vapor phase is assumed to be constant and equal to the saturation temperature at the bubble pressure while the temperature in the liquid phase is governed by the third equation of system (4.20). At the moment the numerical model is not able to generate a small vapor nucleus on its own. Therefore, in the input file the user should specify an analytical expression for the interface that separates the two phases at the beginning of the simulation. This expression must take into account implicitly the static contact angle θ of the vapor/liquid interface with respect to the wall.

 Ricerca Sistema Elettrico	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	72	121

At the bottom of the domain a uniform heat flux q is prescribed. Due to the presence of this heat flux a thermal boundary layer of superheated liquid develops near the wall and around the bubble and the initial vapor nucleus grows due to the evaporation of the superheated liquid. When the diameter of the bubble reaches the departure diameter the bubble detaches from the wall and moves upwards because of buoyancy forces. We assume that the nucleation sites are distributed regularly in the center of square cells on the heated surface. In this case appropriate symmetry conditions are applied on the lateral faces of our computational domain as boundary conditions for both the velocity and temperature fields. Finally, an outlet boundary condition is chosen for the momentum equation at the top of the domain, where a constant value of the pressure is assumed. A Dirichlet boundary condition is prescribed for the temperature field with the constant value $T = 80^\circ$.

In [3] we compare the results of our simulations with the correlation (4.9) and the results obtained with a two-dimensional LBM [40]. These numerical simulations deal with the numerical investigation of the departure diameter as a function of gravity, with a constant surface tension and a contact angle coefficient. The constant contact angle has been set to less than 90° and the detachment evolution is very smooth. By changing the value of the gravity g we fit the results of our simulations with the expression

$$D_b = a g^b \quad (4.21)$$

and the best fit is given by the pair of values $(a, b) = (9.49, -0.4866)$, while with $b = -0.5$ we find $a = 10.02$. This second expression is in good agreement between the results obtained by the TRIO_U code, with a three-dimensional dynamical simulation, and the correlation based on a static force balance. Similar results are also obtained with the lattice Boltzmann code.

In [3] we also consider the numerical investigation of the departure diameter as a function of the contact angle, with constant surface tension and gravity force. When we consider the dependence of the departure diameter on the contact angle we recover the linear behavior for $\theta \geq 50^\circ$. Below this value the diameter is rather insensitive to the value of the contact angle. This behavior is also found in [40] at small values of θ . As a matter of fact this behavior is also found experimentally, even if the departure from the linear behavior is measured at lower values of the contact angle, $\theta \sim 20^\circ$.

Finally, in [3] we study the dependence of D_b on the surface tension coefficient σ . We now fit our results with the expression

$$D_b = c \sigma^d \quad (4.22)$$

and the best fit is given by the pair of values $(c, d) = (5.175, 0.4325)$. The results are once again rather close to the behavior indicated by the correlation (4.9), but not as good as in the case of gravity. If we set $d = 0.5$ then $c = 6.23$.

4.2.3 Bubble release frequency for heterogeneous nucleate boiling from a single site



Figure 4.48: Evolution of the bubble cycle (1)

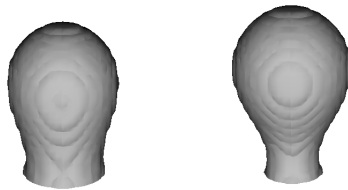


Figure 4.49: Evolution of the bubble cycle (2)

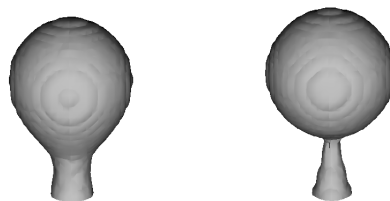


Figure 4.50: Evolution of the bubble cycle (3)

In the simulations described in [3] we study the detachment of a single bubble. Once the condition of full detachment is reached, the interface moves away from the heated lower wall and lifts up the domain. However for the study of the bubble detachment frequency a difficult problem arises for the simulations discussed in [3]. Once the first bubble detaches from the wall, no vapor, hence an interface as well, is left on the wall and a new bubble cycle cannot start again. This difficulty is overcome by placing in the center of domain a bubble of large radius (1 mm) slightly intersecting (0.01 mm) the wall. High contact angles are used to keep the solid surface wet. In these conditions, after the first bubble detaches some vapor is

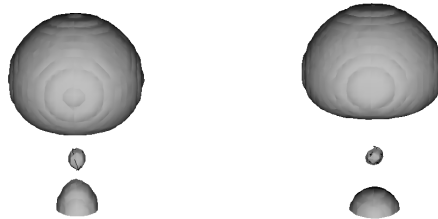


Figure 4.51: Evolution of the bubble cycle (4)



Figure 4.52: Interface on the wall after a bubble cycle

still left on the heated surface and the bubble cycle can start once again. With the aid of the visualization tool VisIt it is possible to view the results produced by the simulations with the TRIO_U code. The evolution of the interface is reported in Figures 4.48-4.51. From Figure 4.52 it is evident that, once the bubble is detached from the wall, a layer of vapor remains on the wall. This layer allows the formation of a new bubble.

As shown in the paper by Hazi and Markus [40], bubble release frequencies are related to the gravity magnitude and the value of the surface tension coefficient by Zuber's relation

$$f^{-1} \sim D_b \left[\frac{g\sigma(\rho_l - \rho_g)}{\rho_l^2} \right]^{-1/4} \quad (4.23)$$

In this section we intend to investigate the evolution of the bubble release frequency by changing the value of three different parameters: the magnitude of the gravity force g , the value of the surface tension σ and the contact angle θ . The results obtained by the simulations conducted with TRIO_U are then compared with the results provided by the paper of Hazi and Markus [40]. For all simulations we set a maximum time of about $t = 0.6$ s and record the following quantities: the time instant at which the first bubble is released, the number of bubbles that are released within the time interval, the amount of time τ between the release of two consecutive bubbles, and the mean interval $\bar{\tau}$ between the release of two consecutive bubbles.

The quantity $\bar{\tau}$ is defined by

$$\bar{\tau} = \sum_{i=1}^{N-1} \tau_i / (N - 1), \quad (4.24)$$

where N is the number of bubbles that are released. In the calculation of $\bar{\tau}$ one must subtract the time interval up to the first detachment of the bubble. The first bubble detachment is shorter because of the initial interface location. The simulations require a few cycles in order to remove the influence of the initial conditions and reach a more or less stationary situation. The mean frequency of the cycle \bar{f} is defined by

$$\bar{f} = \frac{1}{\bar{\tau}}. \quad (4.25)$$

If we combine the two correlations (4.9) and (4.10) we find

$$f \propto \frac{g^{3/4}}{\theta \sigma^{1/4}}. \quad (4.26)$$

In agreement with (4.26), our preliminary results show that the frequency of the cycle increases with gravity and decreases with both the contact angle and the surface tension coefficient.

Bubble release frequency as a function of gravity

Gravity (m/s ²)	Release time (s)
60	0.072
70	0.054
80	0.040
90	0.029
100	0.020

Table 4.15: Release time of the first bubble as a function of the gravity g

In these simulations five different values of the gravity g values are taken. The value of gravity determines the buoyancy forces acting on the bubble. In the TRIO_U simulations we consider values between 60 and 100 m/s^2 , while the other simulation parameters are kept fixed: $\sigma = 0.05 N/m$ and $\theta = 110^\circ$. Starting from an initial configuration with the bubble in the center of the lower wall, the buoyancy forces acting on the bubble tend to detach it after a certain time. The first bubble is released after different times, according to the value assigned to the parameter g . In particular, following Tables 4.15 -4.16 one can observe that the first bubble detaches sooner from the lower wall the greater is the magnitude of gravity. This is valid

Gravity g (m/s^2)	Bubble (#)	Release time (s)	τ (s)
60	Bubble1	0.072	
	Bubble2	0.236	0.164
	Bubble3	0.381	0.145
	Bubble4	0.518	0.137
70	Bubble1	0.054	
	Bubble2	0.205	0.151
	Bubble3	0.340	0.135
	Bubble4	0.463	0.123
	Bubble5	0.583	0.120
80	Bubble1	0.040	
	Bubble2	0.179	0.139
	Bubble3	0.304	0.125
	Bubble4	0.420	0.116
	Bubble5	0.531	0.111
90	Bubble1	0.029	
	Bubble2	0.161	0.132
	Bubble3	0.279	0.118
	Bubble4	0.395	0.116
	Bubble5	0.501	0.106
100	Bubble1	0.020	
	Bubble2	0.146	0.126
	Bubble3	0.261	0.115
	Bubble4	0.367	0.106
	Bubble5	0.465	0.098
	Bubble6	0.560	0.095

Table 4.16: Bubble order, release time and interval between two release events for different g

Bubble (#)	Released time (s)	τ (s)	$\bar{\tau}$ (s)	f (s^{-1})
Bubble1	0.072			
Bubble2	0.236	0.164		
Bubble3	0.381	0.145		
Bubble4	0.518	0.137		
		0.148		6.726

Table 4.17: Bubble release frequency for $g = 60 m/s^2$

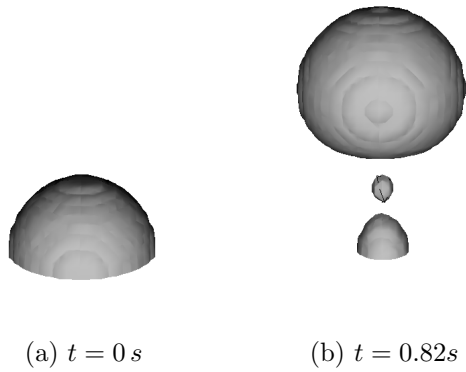


Figure 4.53: Interface shape after the detachment of the first bubble

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	\bar{f} (s^{-1})
Bubble1	0.054			
Bubble2	0.205	0.151		
Bubble3	0.340	0.135		
Bubble4	0.463	0.123		
Bubble5	0.583	0.120		
			0.132	7.561

Table 4.18: Bubble release time for $g = 70\text{ m/s}^2$

Bubble	Release time (s)	τ (s)	$\bar{\tau}$ (s)	\bar{f} (s^{-1})
Bubble1	0.040			
Bubble2	0.179	0.139		
Bubble3	0.304	0.125		
Bubble4	0.420	0.116		
Bubble5	0.531	0.111		
			0.122	8.146

Table 4.19: Bubble release time for $g = 80\text{ m/s}^2$

Bubble	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.029			
Bubble2	0.161	0.132		
Bubble3	0.279	0.118		
Bubble4	0.395	0.116		
Bubble5	0.501	0.106		
			0.118	8.474

Table 4.20: Bubble release time for $g = 90 \text{ m/s}^2$

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.020			
Bubble2	0.146	0.126		
Bubble3	0.261	0.115		
Bubble4	0.367	0.106		
Bubble5	0.465	0.098		
Bubble6	0.560	0.095		
			0.108	9.259

Table 4.21: Bubble release time for $g = 100 \text{ m/s}^2$

both for the release of the first bubble and for the subsequent releases. As we have mentioned earlier, if we analyze the results provided by the simulations, we observe that the bubbles are not released at constant time intervals. In particular, the time interval between the release of two consecutive bubbles tends to decrease. In fact, for all the gravity values considered, a steady bubble release process is never established before 0.6 s . In Table 4.16 a schematic summary of the results of the simulations conducted with TRIO_U is reported. It is easily seen that the release of the first bubble is quicker than all subsequent events. This should not be misleading because, as it was already mentioned, at $t = 0\text{ s}$ the bubble interfaces are larger than those of the following release cycles. (Figure 4.53 is related to a simulation in which $g = 60\text{ m/s}^2$.) This consideration will be also valid when we analyze later on the simulation results obtained for the release frequency as a function of the other two independent parameters, σ and θ . Tables 4.17-4.21 summarize the results including the average interval ($\bar{\tau}$) and the mean release frequency (\bar{f}).

Bubble release frequency as a function of surface tension

Surface tension (N/m)	Release time (first) (s)
0.03	0.025
0.04	0.049
0.05	0.072
0.06	0.093
0.07	0.112

Table 4.22: Bubble release time for different surface tension σ

Table 4.23 summarizes the simulation results. In this subsection the rate of release of a single bubble is studied as a function of the surface tension coefficient σ with constant gravity $g = 60\text{ m/s}^2$ and contact angle $\theta = 110^\circ$. In particular, the surface tension is assumed in the range from 0.03 N/m up to a maximum value of 0.07 N/m .

From the initial configuration, where the bubble is at the center of the domain, the net effect of the forces acting on the bubble itself tends to detach it. However, it is indeed the surface tension that tends to keep the bubble on the solid surface. In fact, as one modifies the value of the surface tension, one can note that the first bubble is generated at different instants of time. In particular, from Table 4.22 it is observed that the release time of the first bubble is longer as the surface tension σ is increased. Tables 4.24-4.28 summarize the release cycle time, the average interval $\bar{\tau}$ and the mean release frequency \bar{f} .

Surface tension (N/m)	Bubble (#)	Release time (s)	τ (s)
0.03	Bubble1	0.025	
	Bubble2	0.149	0.124
	Bubble3	0.260	0.111
	Bubble4	0.361	0.101
	Bubble5	0.461	0.100
	Bubble6	0.556	0.095
0.04	Bubble1	0.049	
	Bubble2	0.191	0.142
	Bubble3	0.324	0.133
	Bubble4	0.444	0.120
	Bubble5	0.558	0.114
0.05	Bubble1	0.072	
	Bubble2	0.236	0.164
	Bubble3	0.381	0.145
	Bubble4	0.518	0.137
0.06	Bubble1	0.093	
	Bubble2	0.278	0.185
	Bubble3	0.445	0.167
0.07	Bubble1	0.112	
	Bubble2	0.326	0.214
	Bubble3	0.445	0.208

Table 4.23: Number of released bubbles, release time and time interval between two events as a function of surface tension σ .

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	\bar{f} (s ⁻¹)
Bubble1	0.025			
Bubble2	0.149	0.124		
Bubble3	0.260	0.111		
Bubble4	0.361	0.101		
Bubble5	0.461	0.100		
Bubble6	0.556	0.095		
			0.106	9.416

Table 4.24: Bubble release time for $\sigma = 0.03$ N/m.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.049			
Bubble2	0.191	0.142		
Bubble3	0.324	0.133		
Bubble4	0.444	0.120		
Bubble5	0.558	0.114		
			0.127	7.858

Table 4.25: Bubble release time for $\sigma = 0.04 N/m$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.072			
Bubble2	0.236	0.164		
Bubble3	0.381	0.145		
Bubble4	0.518	0.137		
			0.148	6.726

Table 4.26: Bubble release time for $\sigma = 0.05 N/m$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.093			
Bubble2	0.278	0.185		
Bubble3	0.445	0.167		
			0.176	5.681

Table 4.27: Bubble release time for $\sigma = 0.06 N/m$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.112			
Bubble2	0.326	0.214		
Bubble3	0.534	0.208		
			0.211	4.739

Table 4.28: Bubble release time for $\sigma = 0.07 N/m$.

Bubble release frequency as a function of the contact angle

Contact angle	Release time (first) (s)
100°	0.050
110°	0.072
115°	0.083
120°	0.094
125°	0.105
130°	0.116

Table 4.29: Bubble release time as a function of θ .

Contact angle	Bubble (#)	Release time (s)	τ (s)
110°	Bubble1	0.072	
	Bubble2	0.236	0.164
	Bubble3	0.381	0.145
	Bubble4	0.518	0.137
115°	Bubble1	0.083	
	Bubble2	0.253	0.170
	Bubble3	0.408	0.155
	Bubble4	0.551	0.143
120°	Bubble1	0.094	
	Bubble2	0.271	0.177
	Bubble3	0.438	0.167
	Bubble4	0.599	0.161
125°	Bubble1	0.105	
	Bubble2	0.303	0.198
	Bubble3	0.496	0.194
130°	Bubble1	0.116	
	Bubble2	0.332	0.220
	Bubble3	0.571	0.239

Table 4.30: Number of released bubbles, release time and interval of time between two bubbles as a function of θ .

As we have said in the previous sections, the bubble release frequency is strongly determined by the value of the contact angle. In particular from the introductory considerations in Section 4.2.2, the greater the contact angle, the higher is the at-

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.072			
Bubble2	0.236	0.164		
Bubble3	0.381	0.145		
Bubble4	0.518	0.137		
			0.148	6.764

Table 4.31: Bubble release time for $\theta = 110^\circ$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.083			
Bubble2	0.253	0.170		
Bubble3	0.408	0.155		
Bubble4	0.551	0.143		
			0.156	6.442

Table 4.32: Bubble release time for $\theta = 115^\circ$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.094			
Bubble2	0.271	0.177		
Bubble3	0.438	0.167		
Bubble4	0.599	0.161		
			0.168	5.949

Table 4.33: Bubble release time for $\theta = 120^\circ$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	f (s ⁻¹)
Bubble1	0.105			
Bubble2	0.303	0.198		
Bubble3	0.496	0.194		
			0.196	5.102

Table 4.34: Bubble release time for $\theta = 125^\circ$.

Bubble (#)	Release time (s)	τ (s)	$\bar{\tau}$ (s)	\bar{f} (s ⁻¹)
Bubble1	0.116			
Bubble2	0.332	0.220		
Bubble3	0.571	0.239		
			0.229	4.373

Table 4.35: Bubble release time for $\theta = 130^\circ$.

traction force between the bubble and the wall. As a result of these interactions, the residence time of the near-wall liquid increases while the release frequency decreases.

In the simulations carried out with TRIOU_U we set the contact angle between 110° and 130° . Depending on the value assigned to the contact angle, the first bubble release time yields different values. Table 4.29 shows that the bubble release time is short for low contact angles, as mentioned in the article of Hazi and Markus [40]. In Table 4.30 we show a schematic summary of the results obtained with TRIO_U. Tables 4.31-4.35 summarize the bubble cycle frequency: the bubble release time, the average interval $\bar{\tau}$ and the mean release frequency \bar{f} .

In conclusion, nucleate boiling is a very complex phenomenon both from the physical and from the numerical point of view. From the physical point of view this phenomenon requires a comprehensive understanding of the activation process of the nucleation sites which is characterized by the presence of several volume and surface forces that interact with each other and that should be modeled with great care. From the numerical point of view it requires an accurate interface description of the detachment of the bubble involving large interfacial distortion with changes of topology as well. Here we have presented some three-dimensional numerical simulations of the nucleate boiling process performed with the TRIO_U code. By following the bubble growth up to its departure from a heated wall we have measured its diameter and the frequency of the release process. In particular, we have analyzed the dependence of the departure diameter and of the cycle frequency on a number of physical quantities which appear in several correlations, such as gravity, surface tension and contact angle. The results are in agreement with the experimental correlations and the two-dimensional Lattice Boltzmann numerical results in [40].

Integration of codes into the SALOME platform

5.1 SALOME GUI integration

5.1.1 Introduction

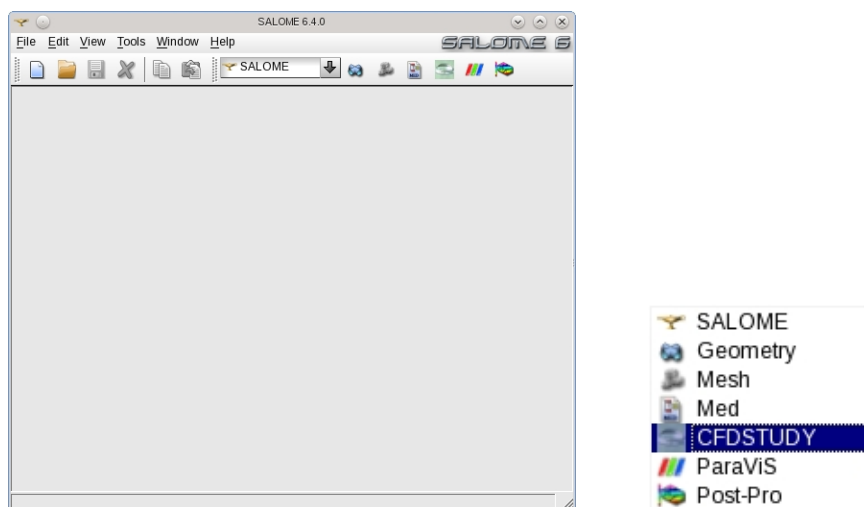


Figure 5.54: SALOME platform and GUI integration

Since GUI programs are developed with the purpose of generating and executing application input files, the first step for code integration is to set standards for the Graphical User Interface (GUI) of all the different applications under a unique framework. As a standard framework we consider the open-source SALOME GUI based on the QT and Python-QT libraries. The SALOME platform is the standard framework chosen by the European NURISP Project. A few applications described in the first chapter have their Graphical User Interface (GUI) based on the QT and Python-QT libraries. Other codes, like NEPTUNE and TRIO_U from EDF and CEA, are developing a GUI starting from the model GUI interface of the open source code SATURNE, which is Python-QT based. Furthermore, the in-house codes can also be adapted to the SATURNE GUI to have similar Python-QT based

GUIs. At the moment, the SATURNE GUI can be considered as the standard model for constructing GUI applications. This open-source GUI is an independent QT/Python-QT application that can be used as a standalone program or embedded into the SALOME platform. Inside the SALOME platform each GUI may improve its performance by using all the other available tools. The SALOME platform comes with a mesh generator and the PARAVIEW visualization application that allow us to complete the pre-processing and the post-processing stages inside the platform itself. The SATURNE GUI standalone version can be integrated into the SALOME GUI and it is available under a submenu of the SALOME main menu bar (CFDSTUDY) as shown in Figure 5.54. The selection of the application starts the loading of the application GUI inside the SALOME framework itself. In Figures 5.55-5.56 a snapshot of the open-source SATURNE GUI as a standalone application and a snapshot of the version integrated in the SALOME platform are shown. All the actions available in the standalone version are then available inside the SALOME platform as well.

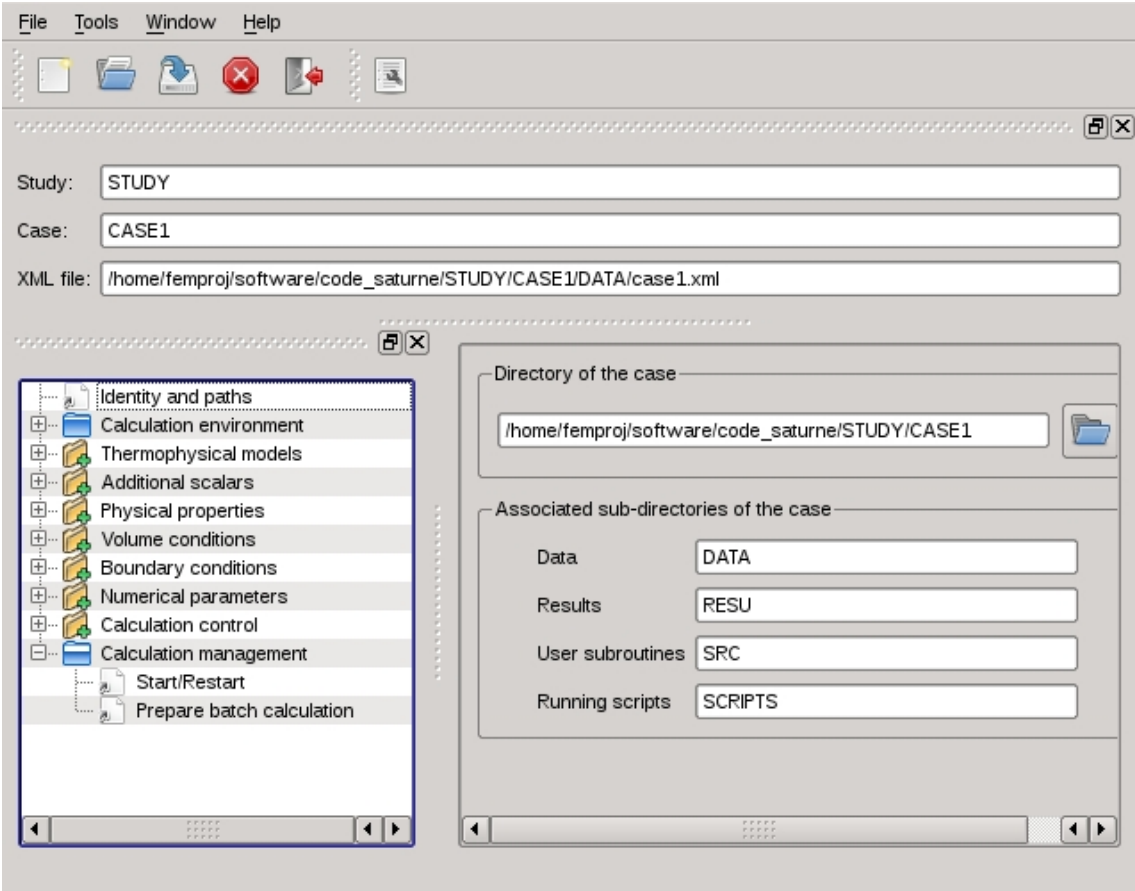


Figure 5.55: SATURNE GUI as a standalone application

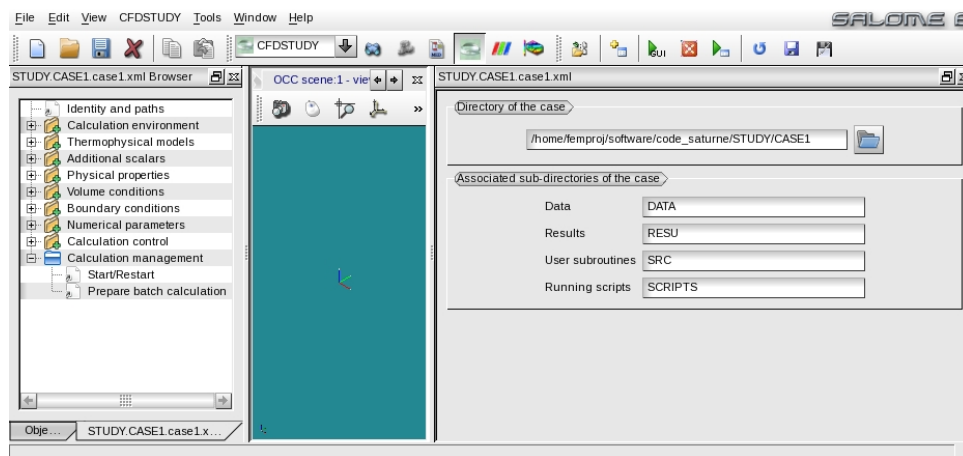


Figure 5.56: SATURNE GUI integrated in SALOME platform

The standard GUI considered here is based on the SATURNE code; it must be started from the application main directory with the command

```
code_application salome
```

where `code_application` denotes the GUI main executable of each application. In case of error the command displays the use as follows

```
Usage: ./code_application <topic>
```

Topics:

```
help
autovnv
compile
config
create
gui
info
run
salome
```

Options:

```
-h, --help show this help message and exit
```

The command first sets the environment variables for the application libraries and then starts the GUI. For the different options the reader can refer to the SATURNE GUI guide in [29]. If the environment variables are already set then from the GUI directory of the application one can simply write

```
python ./bin/code_application salome
```

In this case the application starts correctly due to the previous settings. The GUI for the different applications and for SALOME as well is written in Python and therefore the Python environment must be installed.

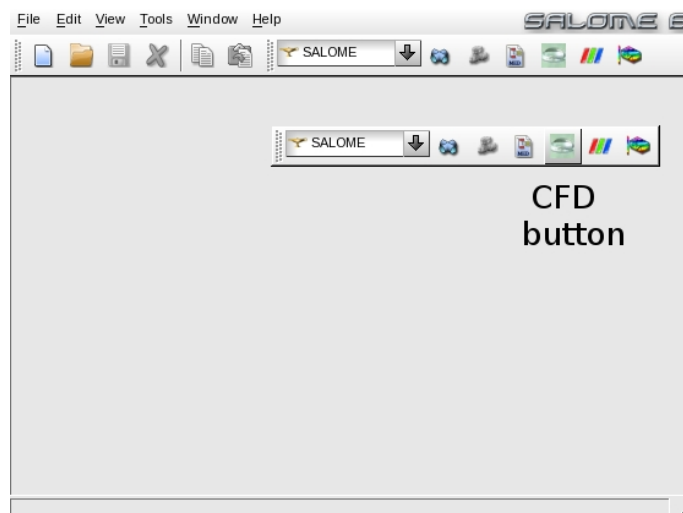


Figure 5.57: Application CFDSTUDY load button in SALOME GUI

In order to load the application interface one must follow some simple steps. After the above command is executed then the SALOME GUI starts and the screen in Figure 5.57 is shown. If the CFDSTUDY load button is selected, the screen dialogs of Figures 5.58 and 5.59 appear and the application is loaded in the SALOME GUI. The application toolbar is loaded on the SALOME toolbar on the top right. In order to set the application directory, one must fill the *Study location* and the *Study name* fields in the dialog window of Figure 5.59. At this point, as shown in Figure 5.60, the application directory tree can be seen in the SALOME directory window. In order to open the application GUI, the GUI configuration file in the DATA directory of the application must be activated (see Figure 5.60). The GUI configuration file in XML format is immediately detected by the SALOME platform and displayed under the DATA directory. If an XML file is not in this directory the application GUI cannot be detected. In this case an empty XML file should be created in order to activate the application GUI. With a mouse right-click on the XML file under the DATA directory menu, the *Open GUI* option can be selected and this selection immediately opens the application GUI window as in Figure 5.56.

The application GUI can start in different forms with different combinations of windows. The usual SALOME main window panel is divided into different zones. As one can see from Figure 5.61 we have the main SALOME window in the center of the screen. In the main window the SALOME platform can load different other programs such as CAD applications and mesh generators. On the left of Figure 5.61 there is the object browser window (application directory window) with the directory tree. In particular one can note that the application directory tree is under

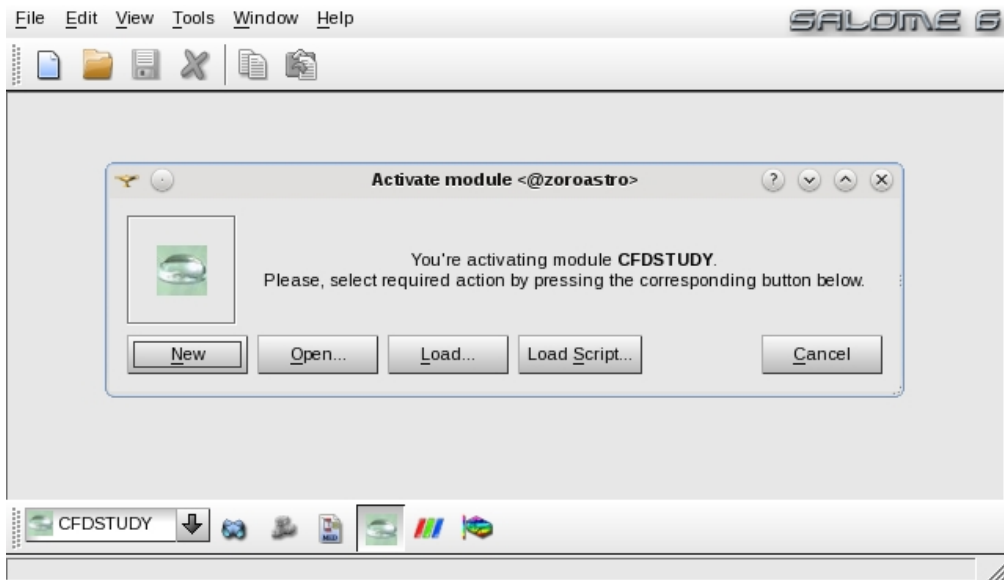


Figure 5.58: Activating a CFDSTUDY GUI

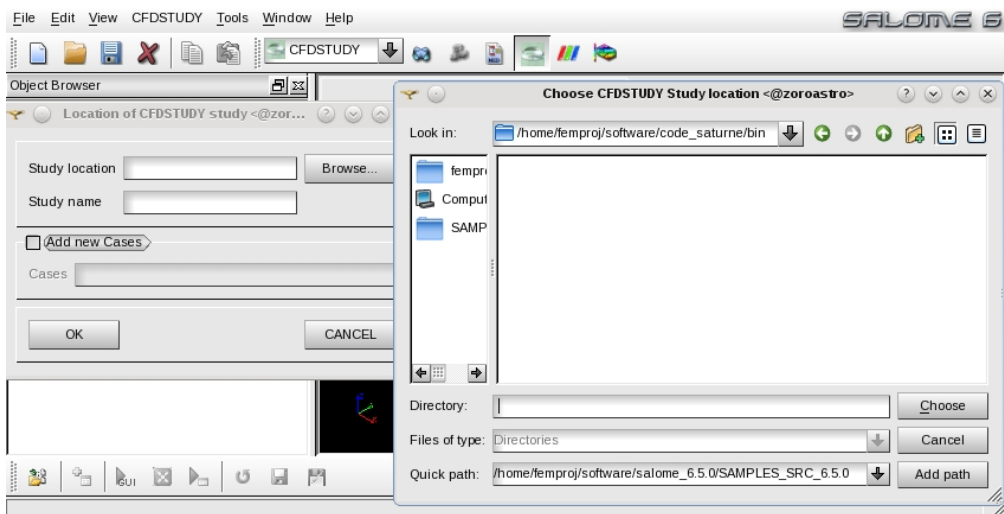


Figure 5.59: Activating the application GUI in SALOME

the CFDSTUDY directory. Also, the MESH directory contains all the meshes for the study and it is located immediately under CFDSTUDY.

The application directory consists of four subdirectories: DATA (configuration directory), RESU (result directory), SCRIPTS (script directory) and SRC (source file directory). The main directory contains only the Makefile which is used for compiling the application code. The application GUI window, which is located on the bottom left side of the SALOME GUI, is shown in Figure 5.61. The tree in the application GUI selects the pages with the options to be written in the configuration file. The RESU directory contains all the input files and all the output files that

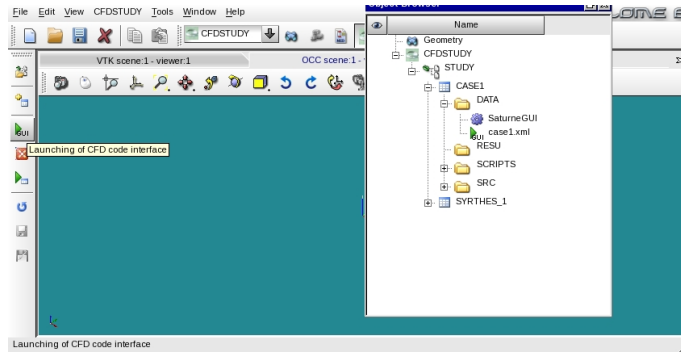


Figure 5.60: Activating the application directory tree in SALOME

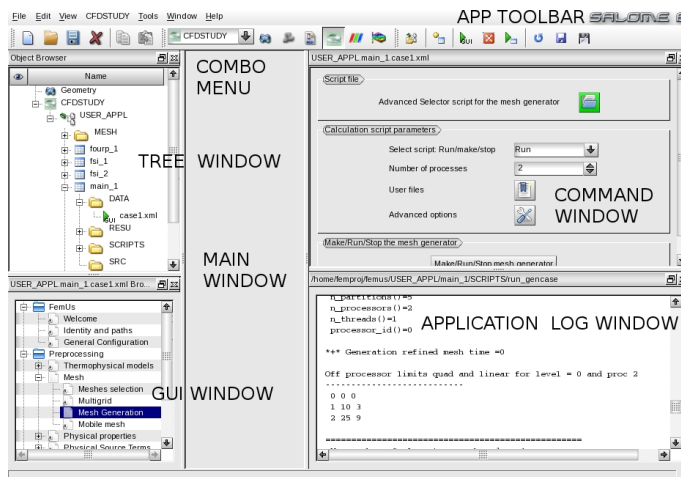


Figure 5.61: Application GUI inside the SALOME framework

can be analyzed through the use of post-processing applications like PARAVIEW. The MED format is used in the SALOME applications. All the files needed for the restart and visualization of the application code are in the RESU directory. The main scripts for the compilation and execution of the application can be found in the SCRIPTS directory. The DATA directory contains the configuration files. A configuration file in XML format must be contained in this directory. On the right side of the SALOME GUI window in Figure 5.61 there are the application command window and the application execution log window. The command window contains the menus and buttons to compile and execute the application. The execution log window, on the bottom right of Figure 5.61, appears when a command in the above window is launched and it reports the log of the commands.

5.1.2 GUI integration with open-source codes: the SATURNE case

The SATURNE GUI directory is composed of the `saturne` and `salome` subdirectories. In the `saturne` directory there is the standalone version of the SATURNE GUI and in the `salome` directory there are the Python classes necessary for the GUI integration in the SALOME framework. These classes are very complex and even a brief description is beyond the scope of this report. Therefore we will describe only what one needs in order to understand how to integrate other codes inside the SALOME framework by using the SATURNE GUI as template.

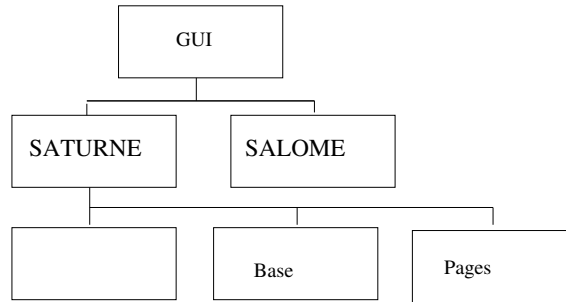


Figure 5.62: SATURNE GUI directory structure

The `saturne` main directory has two subdirectories: `Base` and `Pages`. The main `saturne` directory also contains some configuration files. The `Base` subdirectory consists of the main window widget class and the control over all the GUI pages. The `Pages` directory contains all the pages and the corresponding commands. The most important configuration files are `cs_config.py` and `cs_package.py`. The class inside the `cs_package.py` file contains all the key names used inside the SATURNE GUI together with all the directories where the Python and MPI libraries are located. The type of compiler for FORTRAN, C and C++ can be set here together with the header files of these libraries. In the case of parallel computing, the GUI must dialog with an MPI library as well. The parallel libraries in Table 5.36 can be used. In the

Parallel library
LAM/MPI
MPICH
MPICH2
Open-MPI

Table 5.36: Parallel libraries that can be used with the SATURNE GUI

`cs_config.py` configuration file we must define all the libraries needed for the GUI. Information about the list of libraries that can be linked to the SATURNE GUI must be set. In Table 5.37 there is a brief list of the most important libraries needed for the

Library	version	Info
python	> 2.4	necessary for the SATURNE GUI
pyqt	> 4.3	necessary for the SATURNE GUI
qt	> 4.3	necessary for the SATURNE GUI
libxml2	> 2.6.19	necessary for the SATURNE GUI
swig	> 1.3.30	necessary for the MEI library
Metis	> 4.0	for optimized domain decomposition
CGNS	> 2.5	
MED_fichier	> 2.3	
HDF5	> 1.6.4	necessary for MED_fichier library

Table 5.37: Libraries needed with the SATURNE GUI

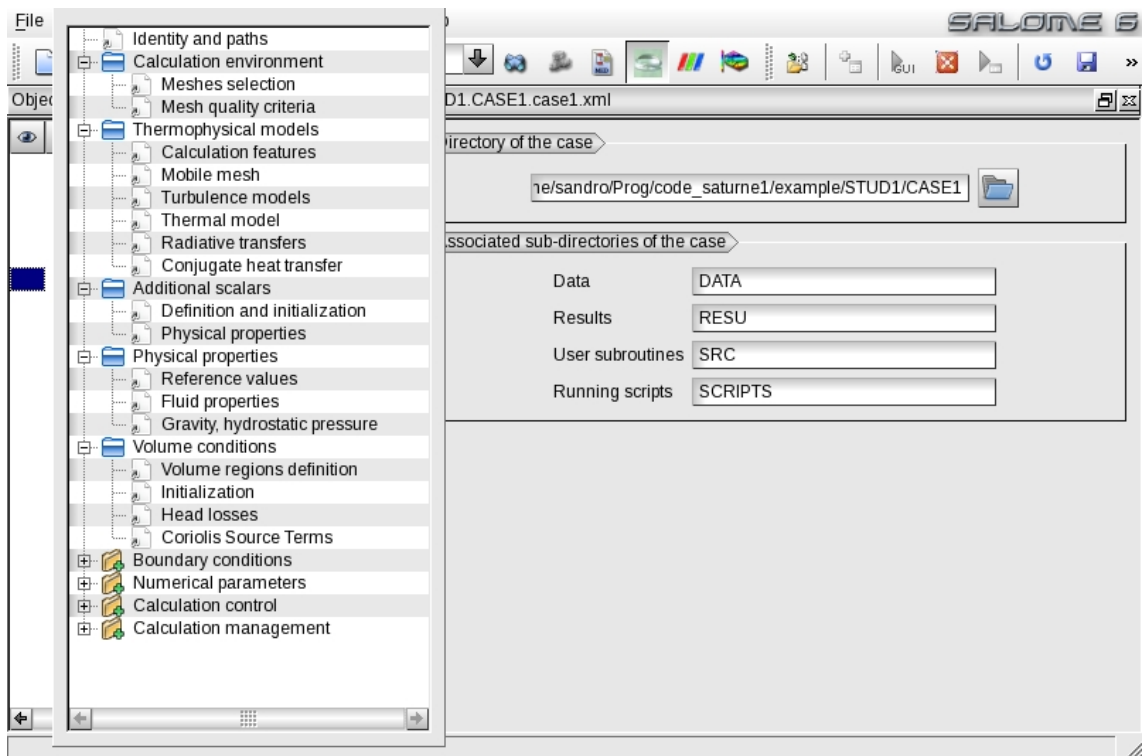



Figure 5.63: The tree of the template GUI inside SALOME framework

SATURNE GUI. The version numbers here reported are those used for developing and testing the version 2.0-beta2 of SATURNE. Other versions (newer or older ones) might still be compatible. The path to the main directory together with the header and library directories of all these libraries must be set in the `cs_package.py` file. If one wants to use the SALOME GUI framework all these libraries should be the same for both SATURNE and for SALOME. Coexisting different versions of the same libraries may not work properly or at all inside the SALOME framework.

In the **Base** directory one can find the pages for the main window and in par-

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	93	121

ticular for the tree menu that appears in the application GUI. The tree menu of SATURNE is shown in Figure 5.63. The tree is generated by the ItemTree class inside the BrowserView.py file. The definition of the tree in Python language is as follows:

```
def _browser(self):
    tree ="""
        Identity and paths
        Calculation environment
        Meshes selection
        Mesh quality criteria
    Storage system description
        Storage system type
        Storage system geometry
        Inlet description
        Outlet description
        Network description
    Thermohydraulic parameters
        Hydraulic load
        Thermal load
    Thermophysical models
        Calculation features
        Mobile mesh
        Turbulence models
        Thermal model
        Gas combustion
        Current species
        Pulverized coal combustion
        Electrical models
        Radiative transfers
        Conjugate heat transfer
        Atmospheric flows
    Additional scalars
        Definition and initialization
        Physical properties
    Physical properties
        Reference values
        Fluid properties
        Gravity, hydrostatic pressure
    Volume conditions
        Volume regions definition
        Initialization
        Head losses
        Coriolis Source Terms
```

```

Particles and droplets tracking
  Global settings
  Statistics
  Output
Boundary conditions
  Definition of boundary regions
  Boundary conditions
  Particles boundary conditions
  Fluid structure interaction
Numerical parameters
  Time step
  Steady flow management
  Equation parameters
  Global parameters
Calculation control
  Time averages
  Output control
  Volume solution control
  Surface solution control
  Profiles
Calculation management
  Start/Restart
  Prepare batch calculation
"""
return tree

```

The selection of a page is set inside the `DisplaySelectedPage` class in the `Toolbox.py` file. Both files are in the Base directory. The GUI menu is defined as a copy of the tree menu defined in the `_browser(self)` object. Each indentation corresponds to a level in tree GUI. For example Thermophysical models is a directory that consists of 11 pages (or subdirectories): Calculation features, Mobile mesh, Turbulence models, Thermal model, Gas combustion, Current species Pulverized coal combustion, Electrical models, Radiative transfers, Conjugate heat transfer and Atmospheric flows. Not all the elements of the tree are visible. The visibility is defined inside the model which is contained in the body of the `BrowserModel` class. In fact, if one sees the corresponding SALOME GUI in Figure 5.63 only the pages Calculation features, Mobile mesh, Turbulence models, Thermal model and Conjugate heat transfer appear under the Thermophysical models menu. The other pages are not visible. The name that appears in the tree is completely arbitrary. A page corresponds to one name. The corresponding table of names and pages is written in the `Toolbox.py` file. For example, we have

```

elif page_name == tr("Calculation features"):
    import Pages.AnalysisFeaturesView as Page

```



```

thisPage = Page.AnalysisFeaturesView(root, case, tree)

elif page_name == tr("Mobile mesh"):
    import Pages.MobileMeshView as Page
    thisPage = Page.MobileMeshView(root, case, tree)

elif page_name == tr("Turbulence models"):
    import Pages.TurbulenceView as Page
    thisPage = Page.TurbulenceView(root, case)

elif page_name == tr("Thermal model"):
    import Pages.ThermalScalarView as Page
    thisPage = Page.ThermalScalarView(root, case, tree) .

```

This implies that the page with label “Calculation features” is linked to the AnalysisFeaturesView file in the Pages directory (Pages.AnalysisFeaturesView). Inside this page the AnalysisFeaturesView class defines all the details of the GUI page. All the pages are defined by three files whose name ends with Form, Model and View, respectively. For example the AnalysisFeatures page is defined by the three files: AnalysisFeaturesForm.py, AnalysisFeaturesModel.py and AnalysisFeaturesView.py. The Form file contains the layout of the Page, as shown in Figure 5.64. It defines the numbers and the locations of the buttons and combo-menu widgets.

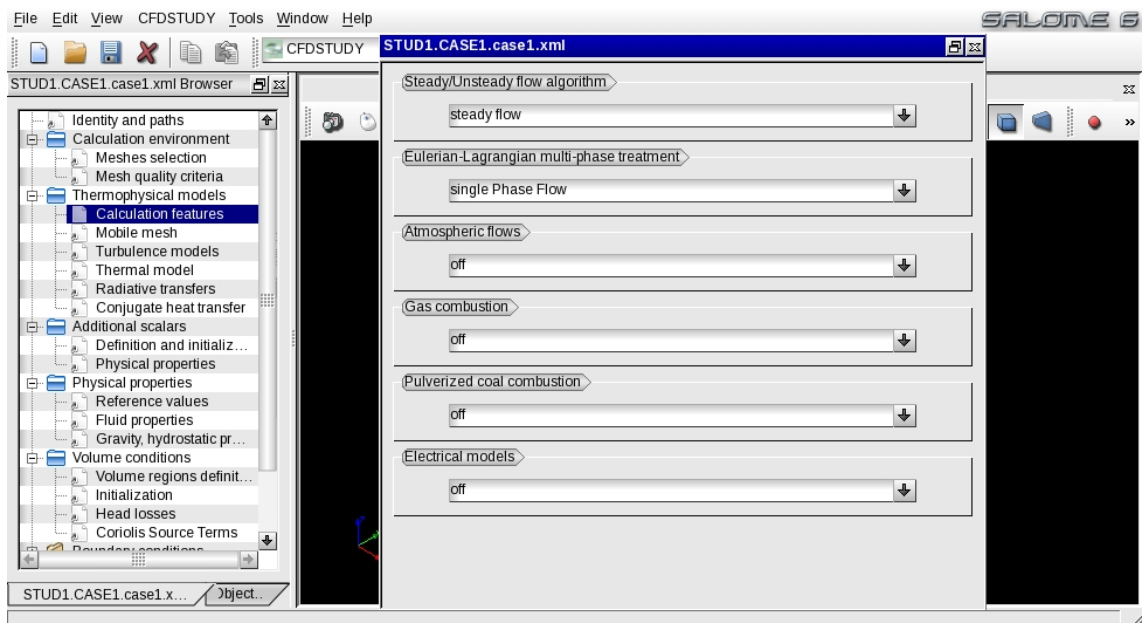


Figure 5.64: The tree of the template SATURNE GUI inside the SALOME framework

The connections and the actions under mouse and keyboard input are defined in

the `View` file. The visibility of the page and the code commands are defined in the `Model` file. The GUI pages can be easily modified with a small knowledge of C++ language, object oriented programming and QT libraries.

Once the standalone GUI is defined, the `salome` directory can be modified. The GUI `salome` directory controls the loading and unloading of the GUI in the SALOME framework. One of the most important issues is that the classes defined in the SALOME directory must load the same libraries with the same version of the classes in the `saturne` directory. The files must be modified in order to satisfy this requirement.

5.1.3 GUI integration with NURISP codes: the NEPTUNE and TRIO_U cases

The NEPTUNE and TRIO_U codes of the NURISP platform have a GUI interface constructed with different approaches. The NEPTUNE GUI is shown in Figure 5.65. The code is not open-source and at the moment a SALOME integration at

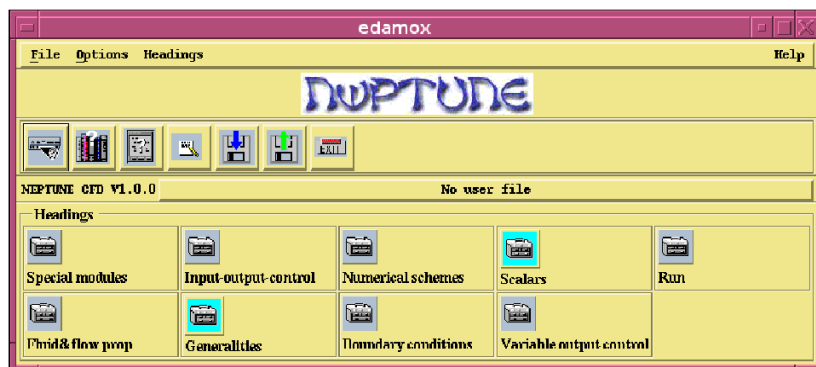


Figure 5.65: Neptune graphical user interface.

the GUI level is not available. CEA and EDF have planned to set up a new GUI in agreement with the SATURNE GUI standard.

The TRIO_U GUI is shown in Figure 5.66. This GUI is based on the Firefox browser and it is not compatible with Python-QT standards. The code is not open-source and at the moment a SALOME integration at the GUI level is not available for this code as well. The integration in the SALOME platform is promoted by the NURISP project but it is necessary to wait some time for the integration development.

With open-source codes the situation is different, and therefore in the next section we briefly propose the integration of an open-source in-house code.

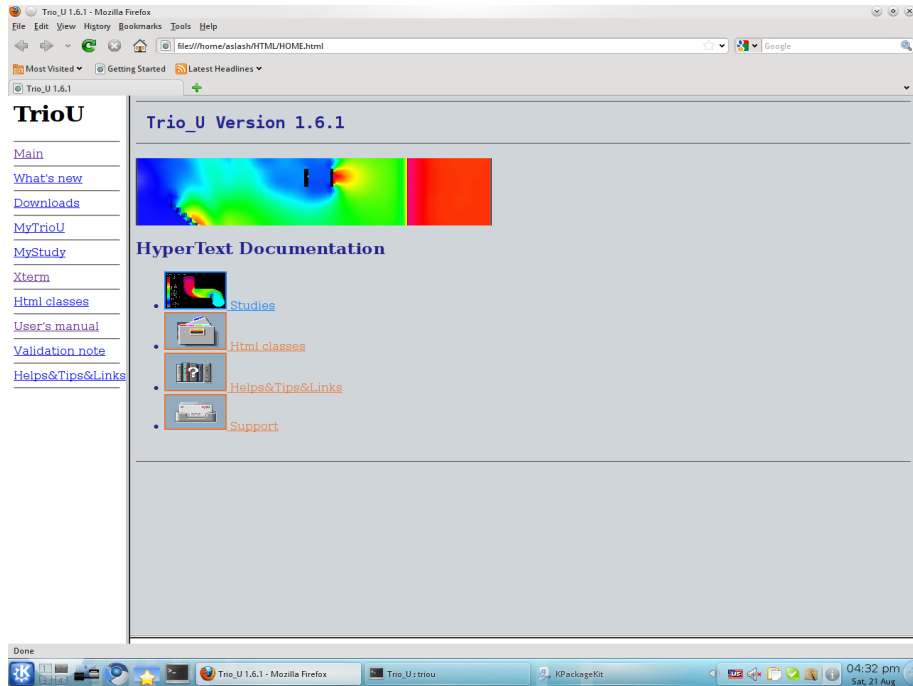


Figure 5.66: TRIO_U graphical user interface

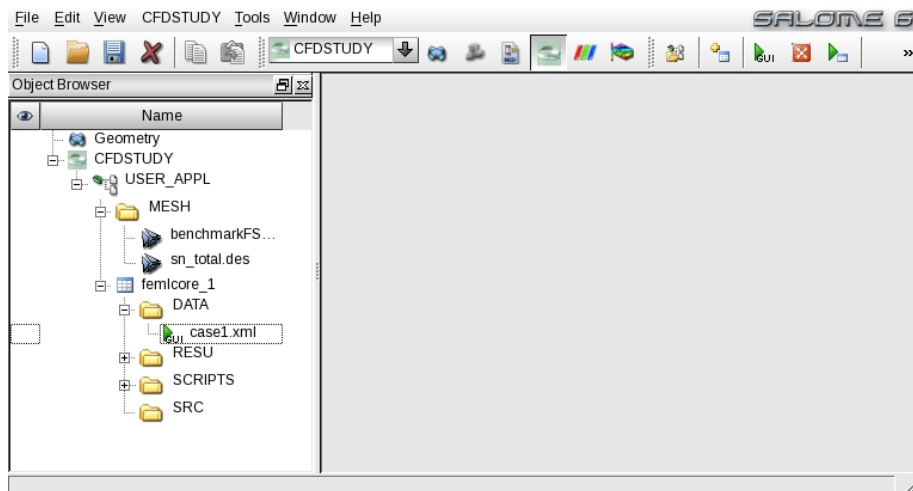



Figure 5.67: FEMuS graphical user interface

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	98	121

5.1.4 GUI integration with in-house codes: the FEMuS case

We have successfully integrated the in-house FEMuS code GUI in the SALOME platform by using the SATURNE template GUI. The FEMuS GUI is shown in Figure 5.67. We briefly report some useful information for this GUI integration. We have modified the `cs_package.py` file in order to define the MPI library and the SALOME main directory location. All the libraries needed for the GUI are set in the `cs_config.py` configuration file. We define all the libraries needed for the GUI as shown in Table 5.37. The GUI tree must be defined to set the FEMuS data input. In the `BrowserView.py` file we see the following tree defined inside the `_browser` object

```
def _browser(self):
    tree = ""
    FEMuS
        Welcome
        Identity and paths
        General Configuration
    Preprocessing
        Thermophysical models
            Equation models
            Coupled Equation model
            Equation advance setting
    Mesh
        Meshes selection
        Multigrid
        Mesh Generation
        Mobile mesh
    Physical properties
        Reference values
        Fluid properties
        Solid properties
    Physical Source Terms
        Source Terms
    Boundary conditions
        Boundary conditions setting in files
        Boundary conditions
        Definition of boundary regions
    Initial conditions
        Initial conditions setting in files
        Volume regions definition
        Initialization
    Computing
        Time step
```

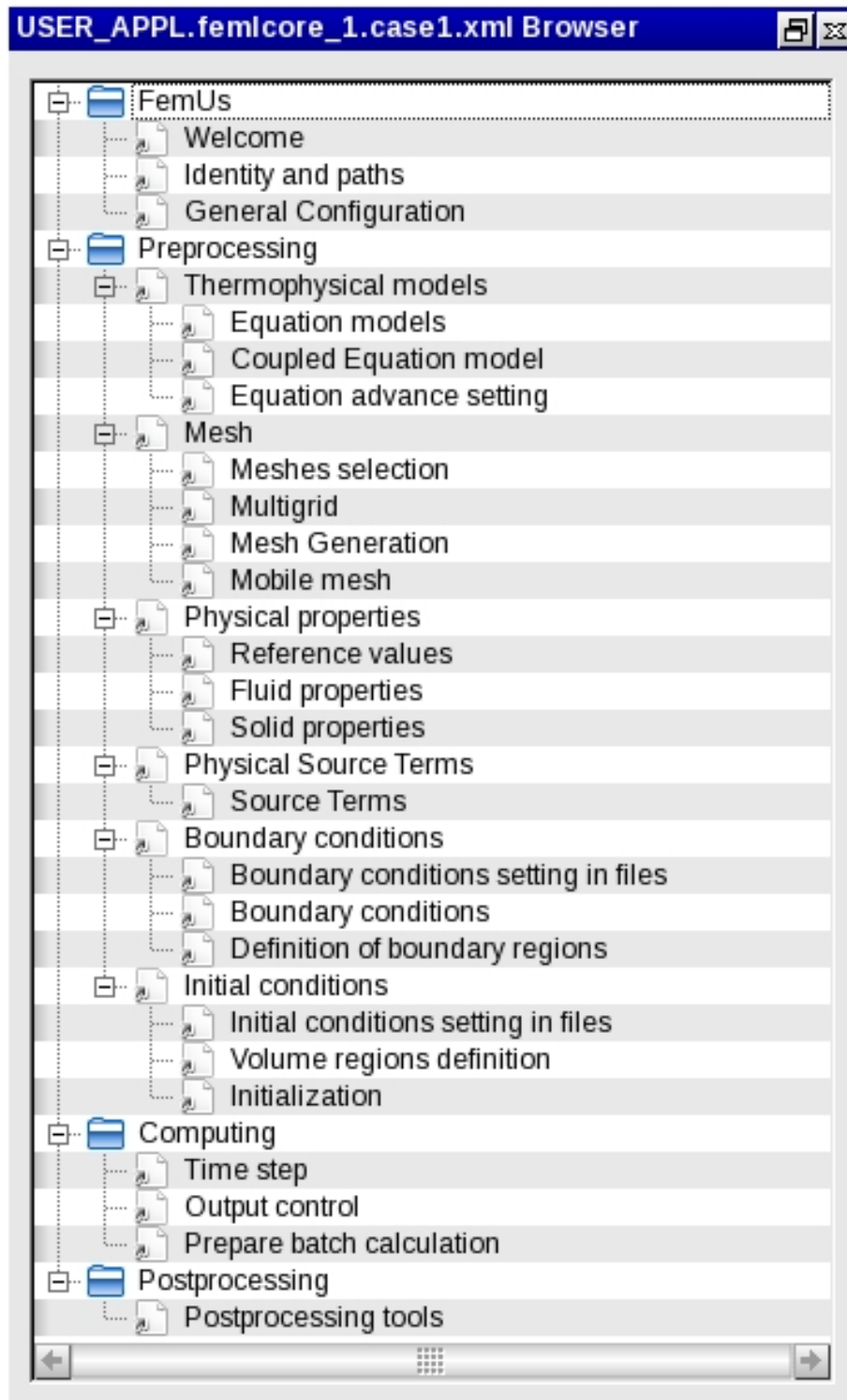


Figure 5.68: FEMuS tree graphical user interface

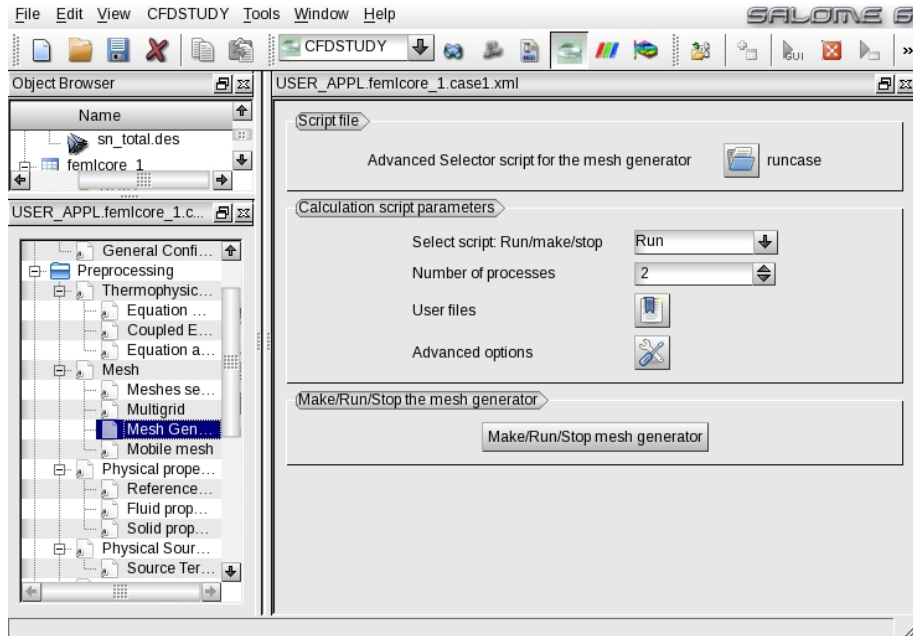


Figure 5.69: FEMuS GUI mesh generation page

```

Output control
Prepare batch calculation
Postprocessing
Postprocessing tools
""
return tree

```

The tree menu is divided into three sections: Pre-processing, Post-processing and Computing. Each section contains the necessary pages which are defined inside the **Pages** directories. In the Pre-processing section, the mesh, the physical properties, the initial and boundary conditions must be set. The time step and the output control can be defined in the Computing branch. In the Post-processing section some important tools are made available for visualization. The above tree, when shown in the SALOME GUI framework, appears as in Figure 5.68. In order to define the pages one can modify the existing ones in the **Form**, **View** and **Model** files as previously discussed. The pages are written in Python and the widgets, which are there present, are generated by using the QT libraries. An example of page is shown in 5.69 where the mesh generator application can be managed by selecting options over a combo menu, a button and a line-edit object.

5.1.5 GUI integration with PARAVIEW

It is possible to integrate a version of the PARAVIEW GUI inside the SALOME framework. The PARAVIEW integrated GUI activation item appears in the menu, as shown in Figure 5.70 on the left. After the selection, the PARAVIEW GUI is

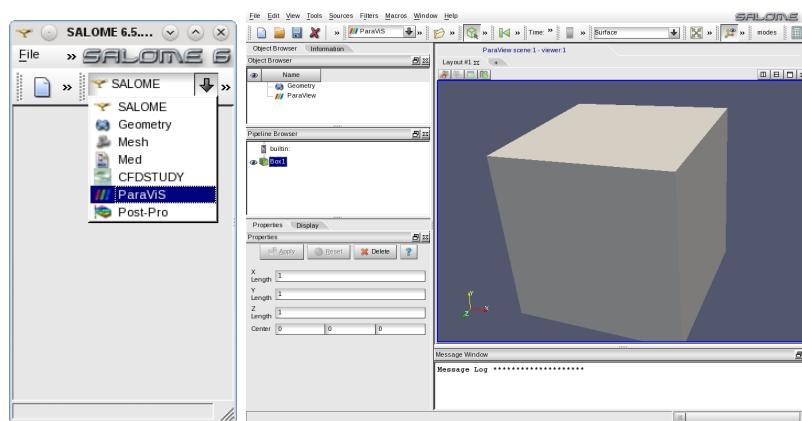


Figure 5.70: Activating the PARAVIEW GUI inside the SALOME GUI

loaded inside the SALOME GUI as one can see in Figure 5.70 on the right.

5.2 Pre-processing and post-processing integration

5.2.1 Integration with open-source codes: the SATURNE case

SALOME-GEOM application

A second step towards code integration is the use of standard input mesh and standard output formats for simulations with the different codes of the platform. In order to reach this objective the mesh generated with the SALOME mesh generator should be used for all codes. Furthermore, all the outputs should be written with a standard format in order to be visualized inside the platform itself with the PARAVIEW application. If the input and output formats are the same then one can use the output of a code as the input for another one and viceversa. For open-source codes like SATURNE the mesh generation and the pre/post-processing operations can be modified in such a way that it is possible to use them completely inside the platform.

Briefly, we explain how this is possible with the open-source SATURNE code that can be considered as a template. First we open the SALOME GUI. Then we design the geometry of the domain with the SALOME CAD application. In order to do this we select the GEOM application from the combo menu at the top. The

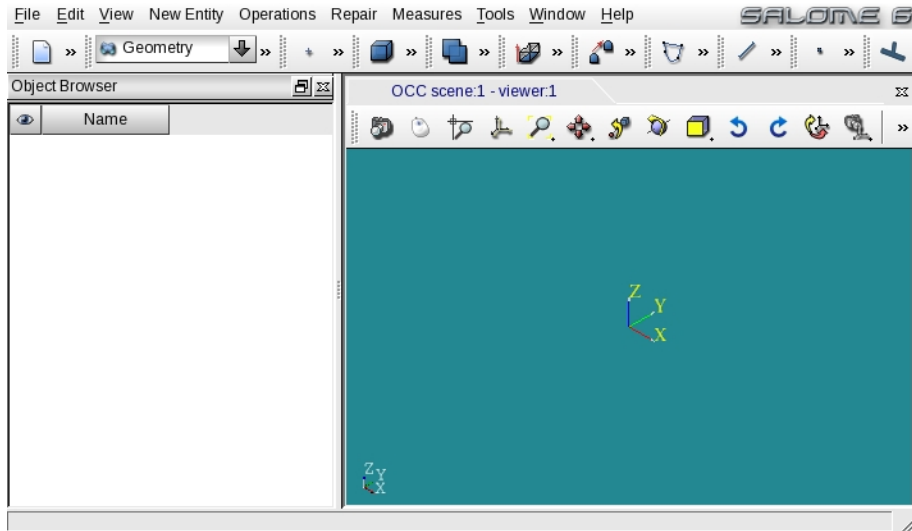


Figure 5.71: SALOME GEOM: GUI view

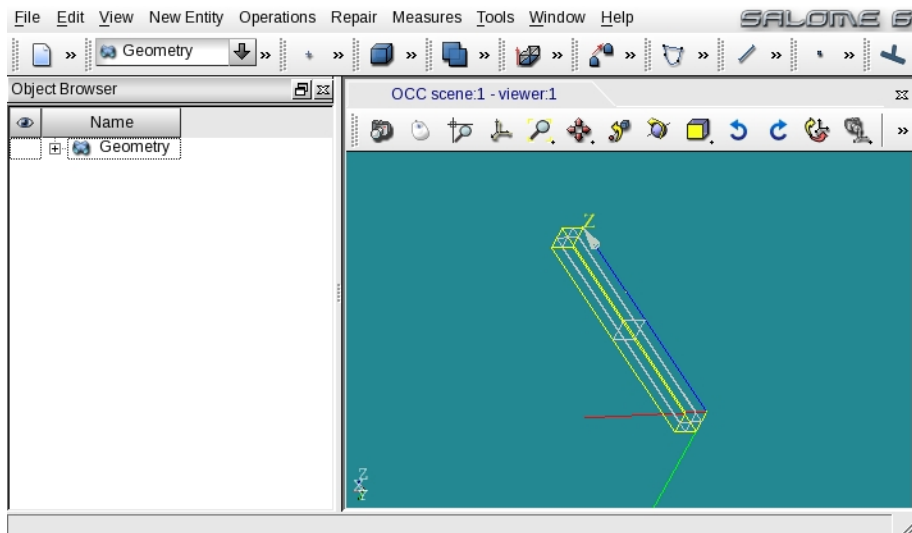


Figure 5.72: SALOME GEOM: box domain (CAD)

GEOM module appears as in Figure 5.71. By using standard commands we can design a box with dimension $[0, .1] \times [0, .1] \times [0, 1]$. For details one can see [9]. The result is shown in Figure 5.72. The code usually solves partial differential equations, therefore boundary conditions are necessary for the existence of solutions. Before leaving the GEOM application one must generate and label the different boundary regions. To define a boundary region one must call the subregion option from the GEOM menu and then select different parts of the boundary as in Figure 5.73. Details can be found in [9].

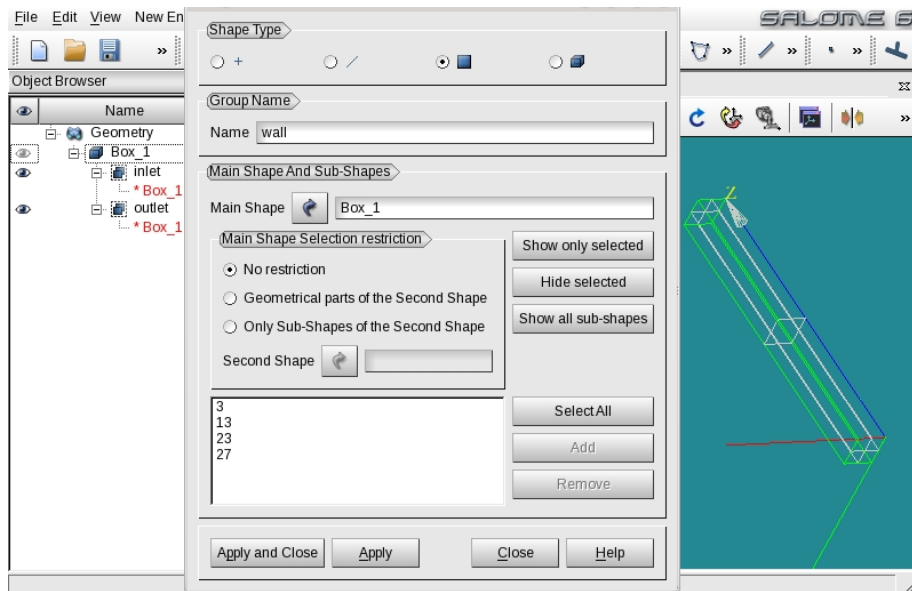


Figure 5.73: SALOME GEOM: definition of the boundary regions: inlet, outlet and wall boundary

SALOME-MESH application

Once we have the geometrical boundary of the domain we must generate a volumetric mesh. From the SALOME combo-menu on the menu bar we select MESH and the MESH application starts as in Figure 5.74. We use the geometry designed with the GEOM application to generate the boundary. We must label the different boundary regions in order to understand where boundary conditions are set. We select each boundary region as seen in Figure 5.75. Each region can be detected automatically inside this page by the SALOME GUI. Therefore, an easy set of the boundary conditions can be done by simply assigning a label to each different region. The details of the generated mesh and the discretization of the lateral wall region are shown in Figures 5.76-5.77. The mesh file must be saved in the MESH directory of the case, as explained in the previous sections. The standard format for the SALOME platform is MED and therefore the name of the file should be for example `mesh.med`.

SALOME-SATURNE application

By using the procedure described in Section 5.1.2 one can open the SATURNE application inside the SALOME GUI as shown in Figure 5.78. After the application is available on the GUI, we can open the mesh and run the mesh check (see [29]). A check file is generated. It is important to see if the boundary regions appear in the file. One should see something like

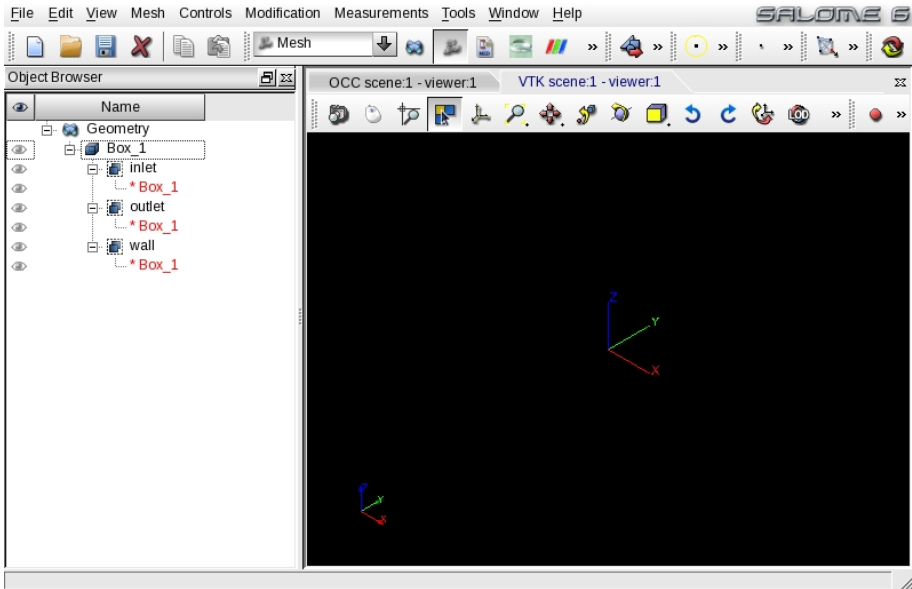


Figure 5.74: SALOME MESH: startup screen

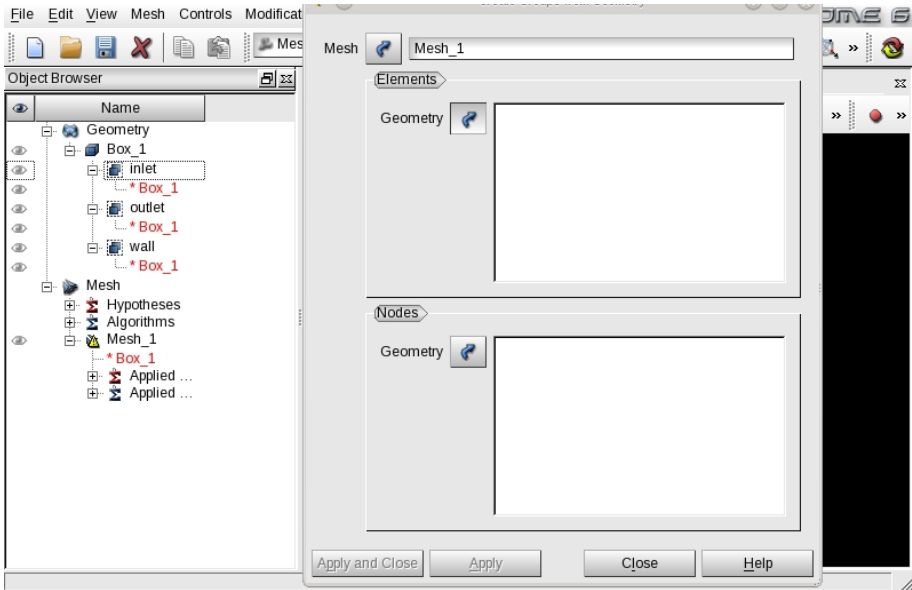


Figure 5.75: SALOME MESH: region selection for boundary conditions

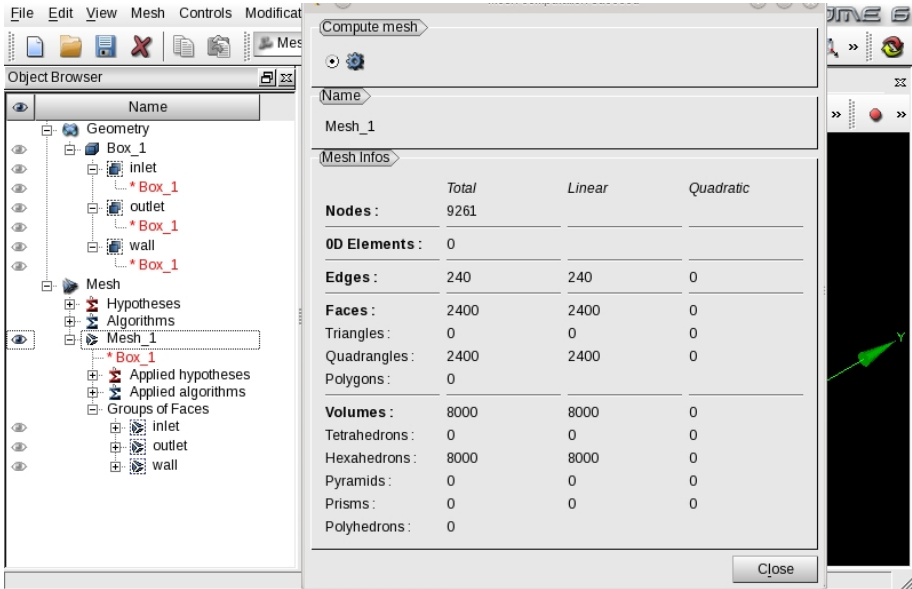


Figure 5.76: SALOME MESH: mesh generation

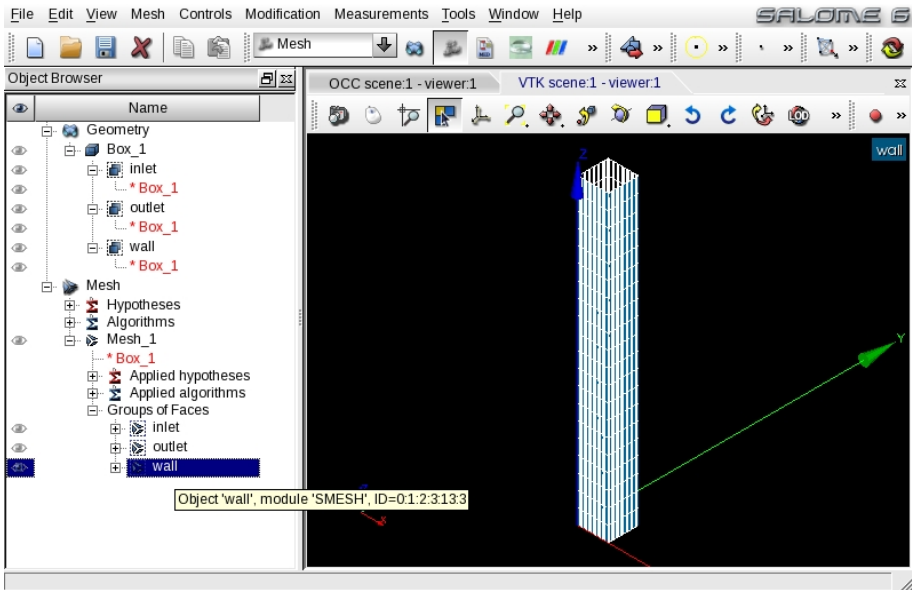


Figure 5.77: SALOME MESH: view of the lateral wall region

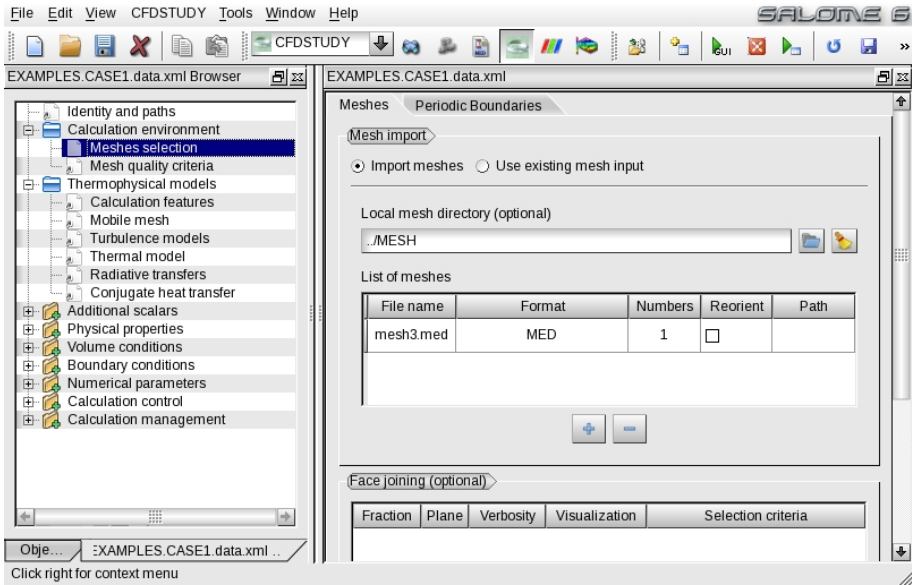


Figure 5.78: SALOME SATURNE: startup screen

Definition of face and cell families

```

Family 1
  Group "inlet"
Number of boundary faces :      400
Family 2
  Group "outlet"
Number of boundary faces :      400
Family 3
  Group "wall"
Number of boundary faces :      1600
Family 3
  Default family
  (no group)
Number of cells           :      8000
Number of internal faces :      22800

```

In this case three boundary regions are detected: **inlet**, **outlet** and **wall**. A file with boundary information is produced with **.log** extension. As shown in Figure 5.79 one must load this file and all the boundary regions should appear inside the GUI. The boundary condition regions are shown in Figure 5.80. For each boundary region one then must select the type of boundary: symmetry, inlet, outlet and

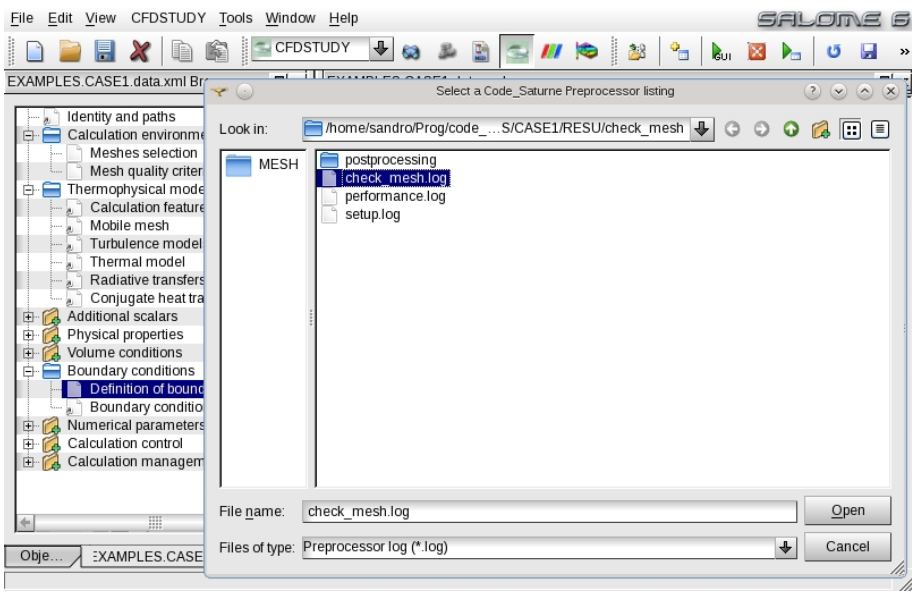


Figure 5.79: SALOME SATURNE: file .log selection for boundary conditions

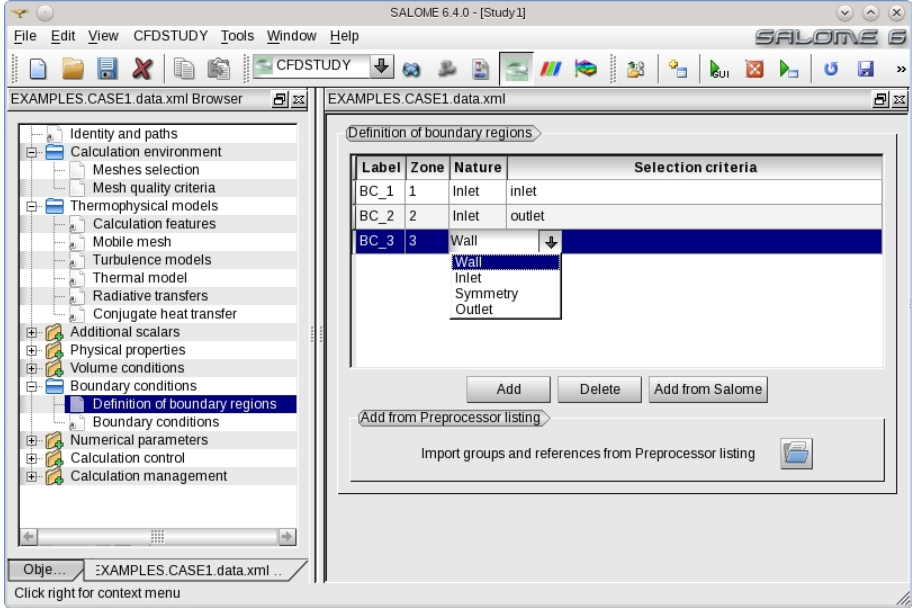


Figure 5.80: SALOME SATURNE: setting the type of boundary conditions

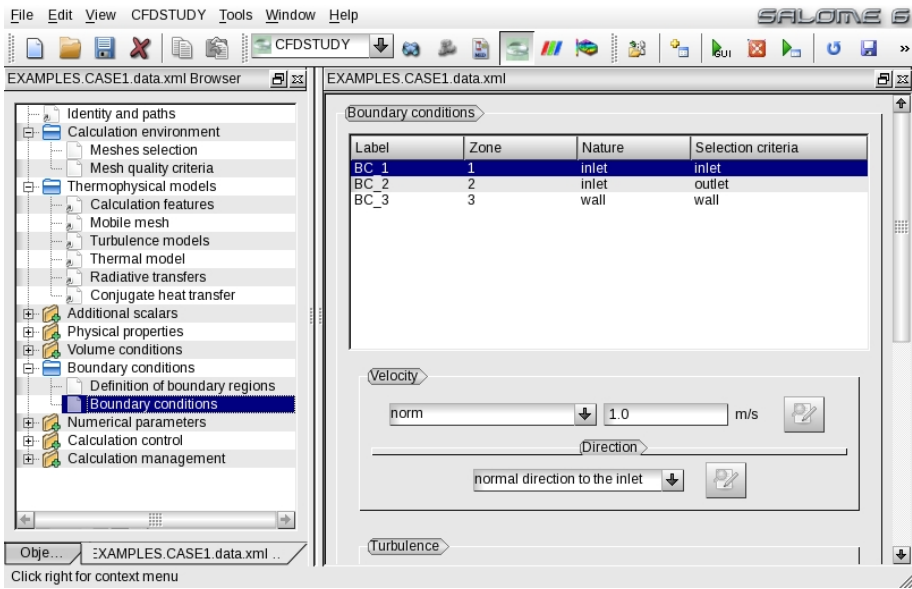


Figure 5.81: SALOME SATURNE: setting the inlet boundary conditions

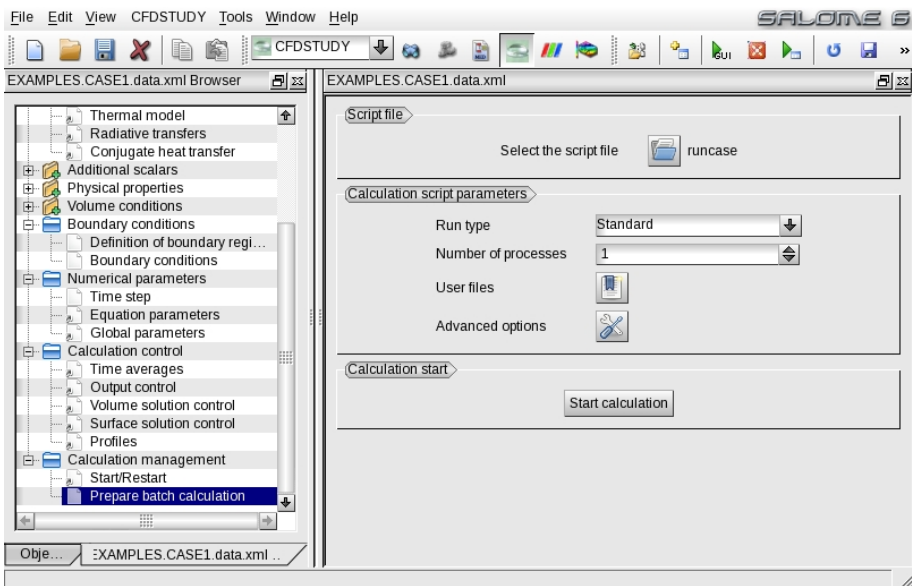


Figure 5.82: SALOME SATURNE: run from GUI

wall. For each type of different boundary conditions a page with possible values is available. For example inlet temperature and velocity must be prescribed as shown in Figure 5.81. Once the boundary conditions are set, together with all the other configuration parameters (see [29]), the code can be launched from the GUI, as shown in Figure 5.82. If the run is successful an output like the following one should appear in the running window:

```

code_saturne is running
*****

Version: 2.2.1
Path:    /home/sandro/Prog/code_saturne

Result directory:
/home/sandro/Prog/code_saturne/EXAMPLES/CASE1/RESU/20120712-0027

Single processor code_saturne simulation.

*****
Preparing calculation data
*****
*****
Preprocessing calculation
*****
*****
Starting calculation
*****
*****
Saving calculation results
*****

```

Of course other information can be displayed. Errors are often due to mistakes in the setting of the boundary conditions.

SALOME-POST-PRO and SALOME-PARAVIEW applications

The results of the computations can be analyzed inside the SALOME platform itself. There are two programs that allow the user to analyze the results: POST-PRO and PARAVIEW.

POST-PRO is a viewer that can read directly the MED format and therefore it is a natural tool for a SALOME application. Figure 5.83 shows the results of a SATURNE run inside the POST-PRO module.

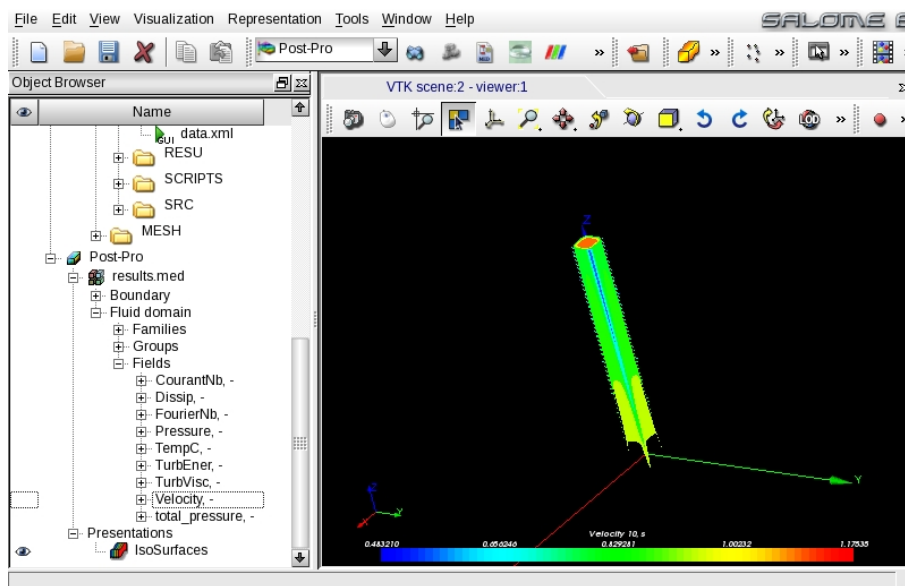


Figure 5.83: SALOME POST-PRO: view of the computations

PARAVIEW is a professional program imbedded into the SALOME GUI that cannot read the MED format directly. A conversion from MED to another HDF5 format is performed automatically inside the platform, since the original PARAVIEW (not embedded) can read HDF5 files but not the MED format. The PARAVIEW GUI inside SALOME is still in the developing stage and errors occur frequently in data reading and writing. A startup screen and a screen with the results on the SALOME-PARAVIEW application can be seen in Figures 5.84 and 5.85.

5.2.2 Integration with NURISP codes: the NEPTUNE and TRIO_U cases

At the moment the NEPTUNE, TRIO_U and CATHARE codes cannot be imbedded into SALOME even if the CEA has promised to release integration as soon as possible. However, both NEPTUNE and TRIO_U support the use of the MED format as mesh input file. The output of these codes is not in MED format but in both cases it can be read with PARAVIEW. In summary the NURISP codes have compatible mesh formats but they are still standalone codes and must be run separately from the SALOME platform.

5.2.3 Integration with in-house codes: the FEMuS case

The in-house FEMuS code can run inside the platform much in the same way as SATURNE [29]. It can read MED mesh input files as well as GAMBIT files. The output is in Xdmf format, easily readable from PARAVIEW.

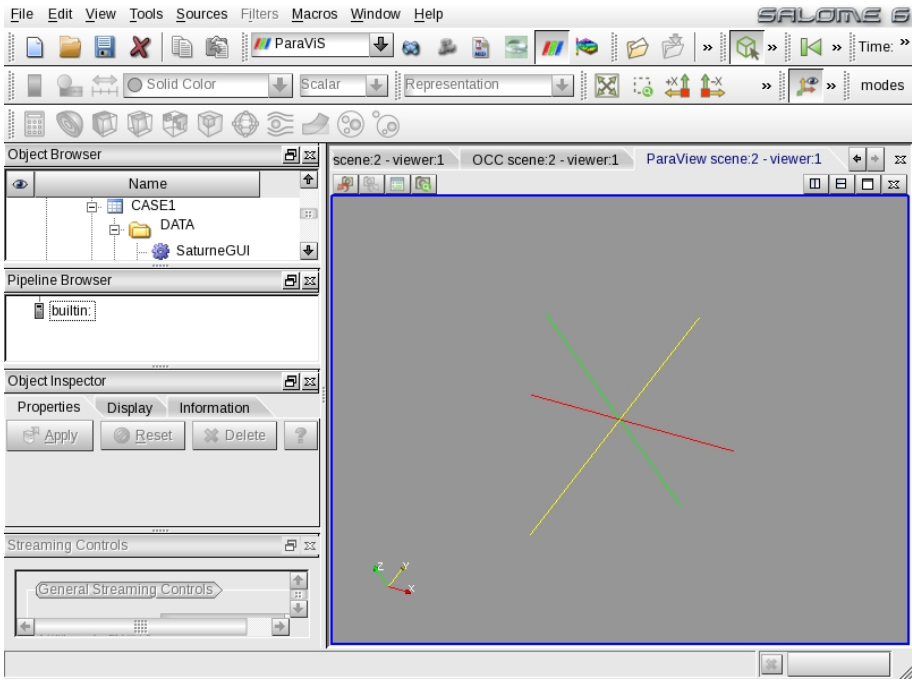


Figure 5.84: SALOME PARAVIEW: startup screen

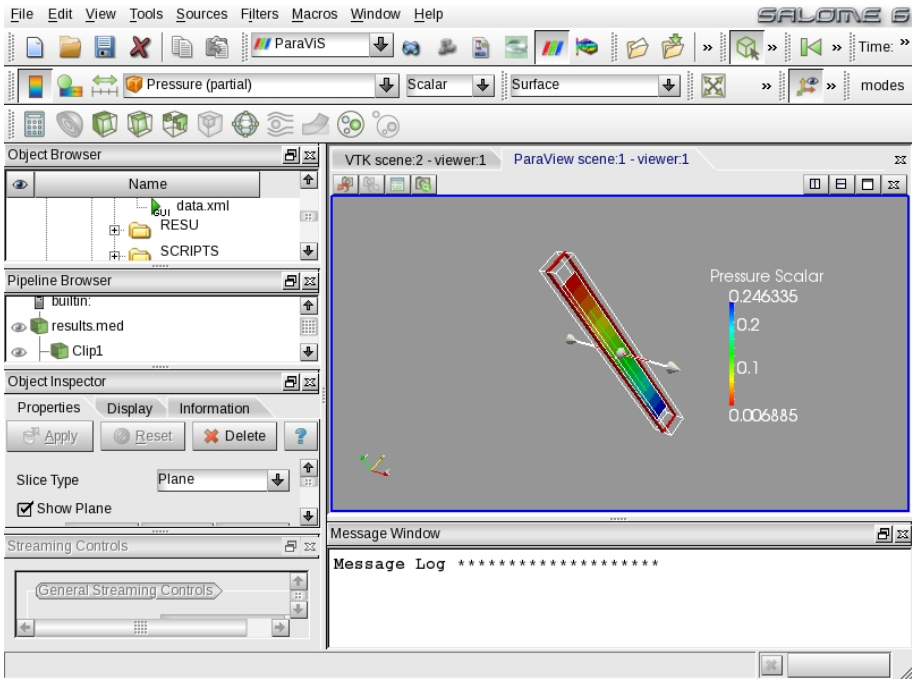


Figure 5.85: SALOME PARAVIEW: view of the computations

5.3 Integration with full coupling

Integration with full coupling is still under development. In order to have full coupling between two codes both of them must read and write in MED format on the SALOME platform. The MED format is not very popular and large code modifications should be done.

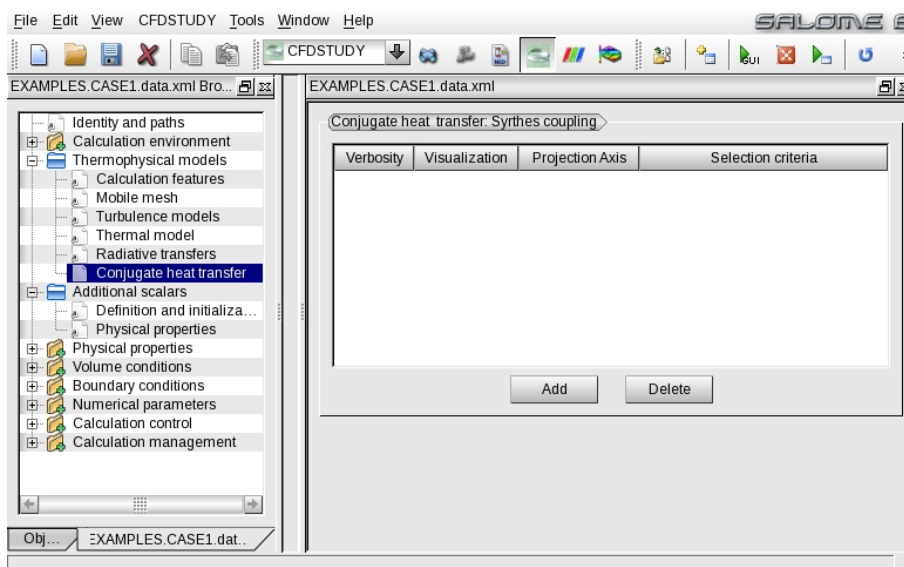




Figure 5.86: SATURNE: conjugate heat exchange with SYRTHES

A step in this direction is given by the SATURNE-SYRTHES coupling. The SATURNE-SYRTHES coupling is available through boundary conditions. Once two physical problems with different domains are generated, one for the fluid to be solved by SATURNE and another one with a solid phase by SYRTHES, the boundary conditions for SYRTHES can be taken from SATURNE and viceversa. See for example Figure 5.86 where the heat fluxes from SYRTHES must be set. Actually only open-source software can be fully integrated. The European NURISP project has planned to release a fully-integrated version of its codes into the SALOME platform. However, at the moment a full integration of NEPTUNE, CATHARE and TRIO_U is not available. We hope to be able to integrate the FEMuS library with the MED format in a short time.

 Ricerca Sistema Elettrico	Sigla di identificazione NNFISS-LP2-089	Rev. 0	Distrib. L	Pag. 113	di 121
--	--	-----------	---------------	-------------	-----------


	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	114	121

Conclusions

The present work concerns the integration and validation of numerical modules that can be used to simulate the thermal-hydraulic behavior of nuclear plants and single components. There are many codes, which have been developed and improved for many years, that work as standalone applications and cannot be coupled to solve multi-physics and multi-scale problem. The NURISP European project is aiming to integrate and develop them inside a large application based on the open-source SALOME platform. Actually this result is far from being reached and many of these codes can still work only in a standalone mode. In this report we discuss the status of different module integration inside the SALOME GUI and MPI libraries.

At the moment the three codes CATHARE, NEPTUNE and TRIO_U are completely independent from each other, with not fully-compatible input and output formats. These codes have different purposes and licenses. CATHARE is a Code for the Analysis of Thermal-hydraulics during an Accident of Reactor and safety Evaluation for LWR. This system code is used as standalone code for PWR safety analysis, accident management and definition of plant operating procedures. In this report we have presented the development of a numerical model for the simulation of the thermal-hydraulic behavior of the SPES-99 facility. The assessment consists of code-to-experiment comparison and code-to-code benchmark. The validation has highlighted the good capability of the model to predict the behaviour of the facility also in transient conditions and suggested possible improvements of the adopted models.


TRIO_U and NEPTUNE codes solve the fluid dynamics equations on multidimensional domains, with particular attention to two-phase flows. In the present work a coupled CATHARE-NEPTUNE_CFD simulation of the PERSEO experiment has been performed. As a first step a reference solution for the system evolution by using the CATHARE code has been obtained. From this solution proper boundary conditions to simulate the Overall Pool and the injection system components by using the NEPTUNE_CFD code have been derived. A solution was obtained on a simplified geometry for the first part of the PERSEO Test 9 transient. The numerical results seem to explain the temperature stratification which is present in the experimental data taken from probes below the injector, with the wall presence, which is only 0.5 m far from the probes. Further investigations on the real geometry and on a computational grid in the convergence regime are necessary to draw final conclusions on this topic. The analysis could not be completed since the version of

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	115	121

the code which has been released to ENEA could not exploit the Infiniband network present in the CRESCO system. This prevented the users from exploiting the full parallel capabilities of CRESCO HPC system. In order to complete the work, a code version compiled with different MPI libraries should be released to ENEA.


The TRIO_U code has been used to analyze the dependence of the departure diameter and its cycle frequency from a number of physical quantities which appear in several correlations, such as gravity, surface tension and contact angle. The results are in agreement with the experimental correlations and the two-dimensional Lattice Boltzmann numerical results. CATHARE, NEPTUNE and TRIO_U are developed by CEA and EDF and they can be used only under NURISP project agreement. The integration of these codes is in the hands of the developers but we have shown that we can reach a good level of integration with open-source software.

In very recent years some other codes, such as SATURNE and SYRTHES that have been developed by EDF and CEA for applications in the nuclear field, have become open-source and can be accessed by all developers in a joint effort to improve the usability and the accuracy in modeling LWR nuclear reactors. SATURNE is now a general purpose CFD free software. Its basic capabilities can deal with incompressible or compressible flows with different turbulence models. In this report we have studied the integration of the SATURNE code inside the SALOME platform. We have shown how to integrate this code inside the SALOME GUI and integrate the input/output format inside the standard SALOME formats. SYRTHES is another open-source finite element code that can be used to handle different kinds of problems such as conduction, radiation in solids. SATURNE can be coupled to the thermal software SYRTHES for conjugate heat transfer. It can also be used jointly with the structural analysis software CODE_ASTER, in particular in the SALOME platform. Both SYRTHES and CODE_ASTER are developed by EDF and distributed under the GNU GPL license. These new codes can be added to the platform improving in this way the possibility to study the very complex behavior of nuclear reactor systems. By using all these available codes the reactor models can be analyzed simultaneously from different points of view. In order to have a more accurate analysis of the behavior of a nuclear reactor system a strong coupling between the various codes is required. Further steps in this direction should be taken to develop the basic tools needed for accurate multi-physics and multi-scale studies.


	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	116	121

Bibliography


- [1] *CRESCO-ENEA GRID Project*, Official documentation at <http://www.cresco.enea.it>. 8
- [2] F. Bassenghi, S. Bna, G. Bornia, A. Cervone, S. Manservisi and R. Scardovelli, *Fissicu platform on CRESCO-ENEA grid for thermal-hydraulic nuclear engineering*, rapporto AdP ENEA-MSE CIRTEN , CERSE-UNIBO RL1302/2010 (2010) 6, 8, 43
- [3] F. Bassenghi, G. Bornia, L. Deon, S. Manservisi, P.Meloni and M. Polidori, Implementation and validation of the NURISP platform, CERSE-UNIBO RL1308/2011 (2011) 6, 8, 15, 16, 27, 61, 67, 71, 72, 73
- [4] *OpenMPI library*, Official documentation at <http://www.open-mpi.org>
- [5] *PETSc (Portable Extension Toolkit for Scientific Computation)*, Official documentation at <http://www.mcs.anl.gov/petsc>
- [6] *HDF5 library*, Official documentation at <http://www.hdfgroup.org/HDF5/>
- [7] *VTK visualization software*, Official documentation at <http://www.vtk.org> 11
- [8] *XDMF library*, Official documentation at <http://www.xdmf.org>
- [9] *SALOME platform*, Official documentation at <http://www.salome-platform.org>.
- [10] *PARAVIEW visualization software*, Official documentation at <http://www.paraview.org> 8, 12, 102
- [11] *The CATHARE code*, Official documentation at <http://www-cathare.cea.fr> 9, 14
- [12] CEA, *CATHARE2 V2.5_1: User's Manual*, SSTH/LDAS/EM/2005-035, (2006) 9

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	117	121


- [13] CEA, *CATHARE2 V2.5.1: User's Guidelines*, DER-SSTH-LDAS-EM-2005-034, (2006) [9](#)
- [14] G. Geffraye et al., *CATHARE 2 V2.5.2: a Single Version for Various Applications*, Proc. 13th International Topical Meeting on Nuclear Reactor Thermal-Hydraulics (NURETH-13), Kanazawa, Japan (2009)
- [15] R. F. Kulak and C. Fiala, *NEPTUNE: A System of Finite Element Programs for Three-Dimensional Nonlinear Analysis*, Nuclear Engineering and Design, 106, pp. 47-68 (1988) [9](#), [13](#)
- [16] D. Bestion and A. Guelfi, *Status and perspective of two-phase flow modelling in the NEPTUNE multi-scale thermal hydraulic platform for nuclear reactor simulation*, Nuclear Engineering and Technology, vol. 37 (6), (2005). [9](#)
- [17] J. Lavieville, E. Quemerais, S. Mimouni, and N. Mechitoua, *NEPTUNE CFD V1.0 theory manual*, EDF (2006)
- [18] Botjan Konar and Borut Mavko, *Simulation of Boiling Flow Experiments Close to CHF with the NEPTUNE CFD Code*, Hindawi Publishing Corporation Science and Technology of Nuclear Installations, Article ID 732158, 8 (2008)
- [19] TRIO_U CFD software, Official documentation at <http://www-trio-u.cea.fr>. [9](#), [13](#), [71](#)
- [20] T. Hohnea, S. Kliema and U. Bieder, *Modeling of a buoyancy-driven flow experiment at the ROCOM test facility using the CFD codes CFX-5 and TRIO_U*, Nuclear Engineering and Design Volume 236, Issue 12, pp. 1309-1325 (2006) [9](#), [67](#)
- [21] U. Biedera, and E. Graffardb, *Benchmarking of CFD Codes for Application to Nuclear Reactor Safety Qualification of the CFD code TRIO_U for full scale reactor applications the FISSICU platform*, Nuclear Engineering and Design Volume 238, Issue 3, pp 671-679 (2008) [67](#)
- [22] U. Biedera, G. Faucheta, S. Betinb, N. Kolevc, and D. Popovd, *Simulation of mixing effects in a VVER-1000 reactor*, Nuclear Engineering and Design Volume 237, Issues 15-17, pp. 1718-1728 (2007) [67](#)
- [23] C. Medich, M. Rigamonti, M. Tarantini *SPES-2 the experimental test facility simulating the AP600 plant* Energia Nucleare, anno 13, gennaio-aprile 1996
- [24] M. Rigamonti *SPES-2 Facility description* SIET 00 183 RI 92, Rev.1. Piacenza, 6-12-95
- [25] R. Ferri *SPES-99 10" IB-LOCA in Cold Leg. Experimental data report* SIET 00777 RP 99, Piacenza; Nov. 26th, 1999

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	118	121


- [26] CEA *CATHARE2 V2.5_1: User's Manual* SSTH/LDAS/EM/2005-035, March, 2006
- [27] M. Zaccarelli *Sviluppo di un modello termo-fluidodinamico dell'impianto sperimentale SPES-99 con il codice di sistema nucleare CATHARE-2* Thesis, University of Bologna (Italy), Engineering Faculty, March, 2010 [16](#)
- [28] F. D'Auria, G.M. Galassi, F. Bianchi, P. Meloni, G. Cattadori, R. Ferri *SPES-99 IBLOCA Relap5/Mod3 pre-test and post-test analysis* CAMP 2000, Palermo (I) April 17-19, 2000 [20](#)
- [29] *SATURNE CFD software*, Official documentation at <http://www.code-saturne.org>. [10](#), [87](#), [103](#), [109](#), [110](#)
- [30] F. Archambeau, N. Mehitoua and M. Sakiz, *Code SATURNE: A Finite Volume Code for Turbulent flows*, Int. J. Finite Volumes (2004) [10](#)
- [31] *SYRTHES software*, Official documentation at <http://research.edf.com> [10](#)
- [32] W. M. Rohsenow, A Method of Correlating Heat Transfer Data for Surface Boiling of Liquids, *Trans. ASME*, vol. 74, pp. 969-976, 1952 [62](#), [63](#)
- [33] K. Stephan and M. Abdelsalam, Heat Transfer Correlation for Natural Convection Boiling, *Int. J. Heat Mass Transfer*, vol. 23, pp. 73-87, 1980 [62](#)
- [34] M. G. Cooper, Saturation Nucleate Pool Boiling - A Simple Correlation, *Institute of Mechanical Engineers, London, IchemE Symposium Series*, vol. 86, pp. 786-793, 1984 [62](#), [63](#)
- [35] D. E. Forster and R. Greif, R., Heat Transfer to a Boiling Liquid— Mechanism and Correlation, *ASME J. Heat Transfer*, vol. 81, pp. 43-53, 1959 [62](#)
- [36] G. Son, V. K. Dhir and N. Ramanujapu, Dynamics and Heat Transfer Associated With a Single Bubble During Nucleate Boiling on a Horizontal Surface, *ASME J. Heat Transfer*, vol. 121, pp. 623-632, 1999 [62](#)
- [37] A. Mukherjee and S. G. Kandlikar, Numerical study of single bubbles with dynamic contact angle during nucleate pool boiling, *Int. J. Heat Mass Transfer*, vol. 50, pp. 127-138, 2007 [62](#)
- [38] D. K. Agarwal, S. W. J. Welch, G. Biswas and F. Durst, Planar Simulation of Bubble Growth in Film Boiling in Near-Critical Water Using a Variant of the VOF Method, *ASME J. Heat Transfer*, vol. 126, pp. 329-338, 2004 [62](#)
- [39] T. Inamuro, T. Ogata, S. Tajima and N. Konishi, A lattice Boltzmann method for incompressible two-phase flows with large density differences, *J. Comput. Phys.*, vol. 198, pp. 628-644, 2004 [62](#)

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	119	121

- [40] G. Hazi and A. Markus, On the bubble departure diameter and release frequency based on numerical simulation results, *Int. J. Heat Mass Transfer*, vol. 52, pp. 1472-1480, 2009 [62](#), [72](#), [74](#), [84](#)
- [41] V. K. Dhir, Mechanistic Prediction of Nucleate Boiling Heat Transfer - Achievable or a Hopeless Task?, *ASME J. Heat Transfer*, vol. 128, pp.1-12, 2006 [63](#)
- [42] R. L. Judd and K. S. Hwang, A Comprehensive Model for Nucleate Boiling Heat Transfer, Including Microlayer Evaporation, *ASME J. Heat Transfer*, vol. 98, pp.623-629, 1976 [63](#)
- [43] B. B. Mikic and W. M. Rohsenow, A New Correlation of Pool Boiling Data, Including the Effect of Heating Surface Characteristics, *ASME J. Heat Transfer*, vol. 9, pp.245-250, 1969 [63](#)
- [44] W. Fritz, Maximum Volume of Vapor Bubble, *Phys. Z.*, vol. 36, pp. 379-384, 1935 [64](#)
- [45] N. Zuber, Hydrodynamic Aspect of Boiling Heat Transfer, Ph.D. thesis, University of California, Los Angeles, 1959 [64](#)
- [46] S. Shin and D. Juric, Modeling Three-Dimensional Multiphase Flow Using a Level Contour Reconstruction Method for Front Tracking without Connectivity, *J. Comput. Phys.*, vol. 180, pp. 427-470, 2002 [66](#)
- [47] G. David, T. Chevalier and G. Meunier, *Unification of Physical Data Models. Application in a Platform for Numerical Simulation: SALOME* the FISSICU platform Magnetics IEEE Transactions, Volume: 43 (4), pp. 1661-1664 (2007)
- [48] A. Achilli et al., *PERSEO Project: Experimental Facility Set-up and RELAP5 Code Calculations*, Proc. 2nd EMSI and 40th European Two-Phase Flow Group Meeting, Stockholm, Sweden, June 10-13, Royal Institute of Technology (KTH), Paper F3 (2002) [29](#)
- [49] G. Bandini, C. Lombardo, P. Meloni and M. Polidori, *Validation of cathare v2.5 thermal-hydraulic code against full-scale perseo tests for decay heat removal in LWRs*, Proceedings of the 18th International Conference on Nuclear Engineering, ICONE18, Xi'an, China (2010) [35](#)
- [50] F. Bianchi et al., *Thermal Valve System for LWR Applications*, Proc. Post-SMIRT14 Seminar, Pisa, Italy (1997) [29](#)
- [51] F. Bianchi et al., *Assessment of RELAP5 MOD3.3 and CATHARE 2 V1.5a against a Full Scale Test of PERSEO Device*, Proc. 12th International Conference on Nuclear Engineering (ICONE12), Arlington, Virginia, USA (2004) [36](#)

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	120	121

- [52] Y.A. Buyevich, B.W. Werbon, *Dynamics of vapour bubbles in nucleate boiling*, Int. J. Heat Mass Transfer 39 2409 (1996)
- [53] R. Ferri et al., *PERSEO Project Experimental Data Report*, SIET 01 014 RP 02, Piacenza, Italy, (2002) [30](#)
- [54] R. Ferri, *SPES-99 10: IB-LOCA in Cold Leg*. Experimental data report, SIET 00 777 RP 99, Piacenza; Nov. 26th, (1999)
- [55] G. Hazi, A. Markus, *On the bubble departure diameter and release frequency based on numerical simulation results*, International Journal of Heat and Mass Transfer 52, pp. 1472–1480 (2009)
- [56] J. Kim, M.H. Kim, *On the departure behaviors of bubble at nucleate pool boiling*, Int. J. Multiphase Flow 32, 1269 (2006).
- [57] P. Meloni et al., *Theoretical Design and Assessment of Isolation Condenser System Controlled with Thermal Valve*, Proc. 6th International Conference on Nuclear Engineering (ICONE6), San Diego, USA (1998) [29](#)
- [58] P. Meloni et al., *Experimental Campaign and Numerical Analysis for the In-pool Energy Removal System for Emergency Operation*, Proc. 13th International Conference on Nuclear Engineering (ICONE13), Beijing, China, (2005)
- [59] C. Medich, M. Rigamonti, M. Tarantini, *SPES-2 the experimental test facility simulating the AP600 plant*, Energia Nucleare, anno 13, gennaio-aprile (1996)
- [60] M. Rigamonti, *SPES-2 Facility description*, SIET 00 183 RI 92, Rev.1. Piacenza, 6-12-95 (1995).
- [61] USNRC, *Three Mile Island Accident*, Backgrounder, Office of Public Affair

	Sigla di identificazione	Rev.	Distrib.	Pag.	di
	NNFISS-LP2-089	0	L	121	121

Notes on the Working Group of the University of Bologna

The working group consists of an associate professor and a researcher in nuclear plants from the University of Bologna (Department DIENCA), R. Scardovelli and S.Manservisi, by a third year student enrolled in DIENCA PhD program at ENEA-Bologna, Federica Bassenghi and the Ph.D. George Borna.

S.Manservisi and R.Scardovelli have carried out research activities at the University of Bologna for several years in the field of Nuclear Engineering, with particular reference to the thermohydraulics and computational fluidynamics. They are authors of several reports in the past PAR activities regarding the computational platform CRESCO-ENEA and the European project NURISP. F. Bassenghi is a third year PhD student at the ENEA Bologna and works in simulations with the code NEPTUNE. The Bassenghi has conducted courses and exchanges with the CEA pursuing collaboration with the above mentioned French research institution. Dr G. Borna took his doctorate at the University of Bologna and now is taking a Visiting Professor position at Texas Tech University.

More details and a list of recent publications can be found in the Web site of the University of Bologna (<http://www.unibo.it>).