NUREG/IA-0542

International Agreement Report

Multi-Scale Coupling of Trace and TrioCFD with the Interface for Code Coupling (ICoCo)

Prepared by:
Kanglong Zhang; Victor Hugo Sanchez-Espinoza

Institute for Neutron Physics and Reactor Technology (INR), Karlsruhe Institute of Technology (KIT) Hermann-von-Helmholtz-Platz 1
Eggenstein-Leopoldshafen, Baden-Württemberg, 76344, Germany

A. Hsieh, NRC Program Manager

Division of Systems Analysis
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001

Manuscript Completed: July 2022
Date Published: June 2025

Prepared as part of
The Agreement on Research Participation and Technical Exchange
Under the Thermal-Hydraulic Code Applications and Maintenance Program (CAMP)

Published by
U.S. Nuclear Regulatory Commission

AVAILABILITY OF REFERENCE MATERIALS IN NRC PUBLICATIONS

NRC Reference Material

As of November 1999, you may electronically access NUREG-series publications and other NRC records at NRC's Library at www.nrc.gov/reading-rm.html. Publicly released records include, to name a few, NUREG-series publications; *Federal Register* notices; applicant, licensee, and vendor documents and correspondence; NRC correspondence and internal memoranda; bulletins and information notices; inspection and investigative reports; licensee event reports; and Commission papers and their attachments.

NRC publications in the NUREG series, NRC regulations, and Title 10, "Energy," in the *Code of Federal Regulations* may also be purchased from one of these two sources.

1. **The Superintendent of Documents**
   U.S. Government Publishing Office
   Washington, DC 20402-0001
   Internet: https://bookstore.gpo.gov/
   Telephone: (202) 512-1800
   Fax: (202) 512-2104

2. **The National Technical Information Service**
   5301 Shawnee Road
   Alexandria, VA 22312-0002
   Internet: https://www.ntis.gov/
   1-800-553-6847 or, locally, (703) 605-6000

A single copy of each NRC draft report for comment is available free, to the extent of supply, upon written request as follows:

Address: **U.S. Nuclear Regulatory Commission**
Office of Administration
Digital Communications and Administrative Services Branch
Washington, DC 20555-0001
E-mail: Reproduction.Resource@nrc.gov
Facsimile: (301) 415-2289

Some publications in the NUREG series that are posted at NRC's Web site address www.nrc.gov/reading-rm/doc-collections/nuregs are updated periodically and may differ from the last printed version. Although references to material found on a Web site bear the date the material was accessed, the material available on the date cited may subsequently be removed from the site.

Non-NRC Reference Material

Documents available from public and special technical libraries include all open literature items, such as books, journal articles, transactions, *Federal Register* notices, Federal and State legislation, and congressional reports. Such documents as theses, dissertations, foreign reports and translations, and non-NRC conference proceedings may be purchased from their sponsoring organization.

Copies of industry codes and standards used in a substantive manner in the NRC regulatory process are maintained at—

The NRC Technical Library
Two White Flint North
11545 Rockville Pike
Rockville, MD 20852-2738

These standards are available in the library for reference use by the public. Codes and standards are usually copyrighted and may be purchased from the originating organization or, if they are American National Standards, from—

American National Standards Institute
11 West 42nd Street
New York, NY 10036-8002
Internet: www.ansi.org
(212) 642-4900

Legally binding regulatory requirements are stated only in laws; NRC regulations; licenses, including technical specifications; or orders, not in NUREG-series publications. The views expressed in contractor prepared publications in this series are not necessarily those of the NRC.

The NUREG series comprises (1) technical and adminis-trative reports and books prepared by the staff (NUREG–XXXX) or agency contractors (NUREG/CR–XXXX), (2) proceedings of conferences (NUREG/CP–XXXX), (3) reports resulting from international agreements (NUREG/IA–XXXX),(4) brochures (NUREG/BR–XXXX), and (5) compilations of legal decisions and orders of the Commission and Atomic and Safety Licensing Boards and of Directors' decisions under Section 2.206 of NRC's regulations (NUREG–0750),and (6) Knowledge Management prepared by NRC staff or agency contractors.

NUREG/IA-0542

International Agreement Report

Multi-Scale Coupling of Trace and TrioCFD with the Interface for Code Coupling (ICoCo)

Prepared by:
Kanglong Zhang; Victor Hugo Sanchez-Espinoza

Institute for Neutron Physics and Reactor Technology (INR), Karlsruhe Institute of Technology (KIT) Hermann-von-Helmholtz-Platz 1
Eggenstein-Leopoldshafen, Baden-Württemberg, 76344, Germany

A. Hsieh, NRC Program Manager

Division of Systems Analysis
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001

Manuscript Completed: July 2022
Date Published: June 2025

Prepared as part of
The Agreement on Research Participation and Technical Exchange
Under the Thermal-Hydraulic Code Applications and Maintenance Program (CAMP)

Published by
U.S. Nuclear Regulatory Commission

ABSTRACT

This report describes the multi-scale coupling of the system code - TRACE and the open-source CFD code – TrioCFD for a better description of the multi-dimensional thermal-hydraulic phenomena inside the Reactor Pressure Vessel (RPV) of a Pressurised Water Reactor (PWR).

The Interface for Code Coupling (ICoCo) is a generic interface for code coupling and it defines a standardized framework for the code functions. ICoCo is already a built-in module embedded in TrioCFD. To couple TRACE with the open-source CFD code - TrioCFD, a specific ICoCo-module for TRACE was developed. In this report, the implemented spatial mapping of the involved thermal-hydraulic domains and the time synchronization of the involved solvers are described. A domain overlapping approach and the open-source MEDCoupling library are utilized for this purpose. Besides, an explicit operator splitting method is implemented for the data transfers during the time advancement of both codes.

The prediction capability of the coupled code is demonstrated by the analysis of a 3D-coolant-mixing problem performed in a VVER-1000 reactor. The results obtained by TRACE standalone and by the coupled system TRACE/TrioCFD were compared together and it shows that the coupled code could predict better coolant mixing along the core height than TRACE-standalone.

FOREWORD

This assessment report deals with the development of a coupling system involving TRACE and TrioCFD using the ICoCo interface. In this report, the development of an ICoCo interface for TRACE is described and the implementation of the coupling of the code is discussed as well. The method for the spatial coupling of the computational domains as well as the synchronization of the time advancement of the two coupled codes is also discussed. The first results obtained with the coupled code TRACE/TrioCFD are presented and discussed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

The scope of this report is to present the development of the multi-scale thermal hydraulic code TRACE/TrioCFD based on the ICoCo interface. First, an ICoCo module was developed for TRACE. Second, the original ICoCo interface in TrioCFD was modified to properly link TRACE. The validation with a VVER-1000 3D coolant mixing case shows that the coupled code can better predict the coolant mixing in the lower plenum.

ABBREVIATIONS AND ACRONYMS

ASTM	Additional Source Term Method
CFD	Computational Fluid Dynamics
CHF	Critical Heat Flux
CMFD	Computational Multi-Fluid Dynamics
CPR	Critical Power Ratio
DES	Detached Eddy Simulation
DIAS	Dynamic Implicit Additional Source
DNS	Direct Numerical Simulation
ECI	Exterior Communication Interface
GUI	Graphic User Interface
IBC	Immersed Boundary Conditions
ICoCo	Interface for Code Coupling
LBLOCA	Large Break LOss of Coolant Accidents
LES	Large Eddy Simulation
LOFW	LOss of Feed Water
LWR	Light Water Reactor
MD	Molecular Dynamics
MPI	Message Passing Interface
MSLB	Main Steam Line Break
NPP	Nuclear Power Plant
OS	Operating Splitter
PTS	Pressurized Thermal Shock
RANS	Reynolds-Averaged Navier-Stokes
RHR	Residual Heat Removal
RPV	Nuclear Pressure Vessel
V.D.F.	Finite Volume Differences
V.E.F.	Finite Volume Element
SBLOCA	Small Break LOss of Coolant Accidents
SCF	SUBCHANFLOW
SGTR	Steam Generator Tube Ruptures

1 INTRODUCTION

1.1 Motivation

Thermal-hydraulic simulation tools play an increasingly crucial role in the present-day Nuclear Power Plant (NPP) safety analysis and design. The thermal-hydraulic physical phenomena occurring in the NPP components (e.g., the Nuclear Pressure Vessel (RPV) involve information on different spatial scales). Their characteristic lengths vary from meters down to nanometers. The NEPTUNE project classified those simulations to three main scales: system scale, component scale (also known as sub-channel scale or CFD in a porous medium) and average scale (also known as CFD in open medium) (Guelfi, et al. 2007) . The European NURESIM and NURISP projects follow its classification method but also refer the average scale to mesoscale and further include a microscale (Bestion, et al. 2012). Moreover, reference (Niceno, et al. 2010) extends the simulation to nanoscale which usually resorts to Molecular Dynamics (MD) modeling techniques. Some other works also use different names for simulation scales but refer to the same stuff (D'Auria, F.; Galassi, G.M. 2010). To describe the thermal-hydraulic processes undergoing at these spatial scales, various thermal-hydraulic simulation codes were developed (e.g., 1D or 3D system thermal-hydraulic codes, porous-media codes (CFD-porous media, sub-channel codes, porous-media 3D codes), open-medium-CFD, Large Eddy Simulation (LES), Detached Eddy Simulation (DES) and Direct Numerical Simulation (DNS).

The system codes were designed to simulate almost all normal and accident scenarios of the whole plant (e.g., Large Break LOss of Coolant Accidents (LBLOCA), Small Break LOss of Coolant Accidents (SBLOCA), Steam Generator Tube Ruptures (SGTR), LOss of Feed Water (LOFW), Main Steam Line Break (MSLB), loss of Residual Heat Removal (RHR) system (Petruzzi and D'Auria 2007). Traditional system codes such as RELAP (RELAP5-3D code manuals, Volumes I, II, IV, and V 1999), TRACE (US NRC 2010), ATHLET (Lerchl and Austregesilo 2006), CATHARE (Emonot, et al. 2011) has already been widely used for many years. Their simulations mostly lie on a system scale.

The simulations on the component scale are mostly carried out using the sub-channel analysis codes (e.g., COBRA-TF (Thurgood, et al. 1983), VIPRE (Sung, Schueren and Meliksetian 1999), FLICA (Toumi, et al. 2000), SUBCHANFLOW (SCF) (Sánchez-Espinoza, Imke and Ivanov 2010) to estimate the various thermal-hydraulic safety parameters in the core (e.g., Critical Heat Flux (CHF) ratio, Critical Power Ratio (CPR), fuel centerline temperature, fuel surface temperature, sub-channel maximum temperature and bulk coolant outlet temperature (Chelemer, Weisman and Tong 1972). A very detailed review of sub-channel analysis methods and codes was given in reference (Moorthi, Sharma and Velusamy 2018), where each aspect of sub-channel simulation was carefully inspected.

Reference (Cheng and Rao 2015) presents another brief review of sub-channel codes and addresses some words for CANDU codes. Recently, to take profit from the keeps-growing computer power and accelerate the simulation efficiency, some sub-channel codes were parallelized (e.g., SCF and CORBA-TF (Kucukboyaci and Sung 2015). Another trend is the development of dedicated 3D porous media two-phase flow codes with a Cartesian or unstructured grid to overcome the limitations of 1D system codes. Instances are CUPID (Jeong, et al. 2010), PORFLOW (Runchal and Sagar n.d.), TWOPORFLOW (Chavez, Imke and Sanchez-Espinoza 2018), PORFLO (Ilvonen, Hovi and Inkinen 2010), et al.

Moreover, to meet the rapidly increasing requirement of the multidimensional simulation within the Reactor Pressure Vessel (RPV) (e.g., the downcomer, lower plenum and upper plenum of aLight Water Reactor (LWR), the CFD codes (e.g., ANSYS CFX, STAR-CD, OpenFOAM, TrioCFD are now playing important role in the thermal-hydraulic analysis of NPP, focusing on simulation of microscale. Reference (Mahaffy, et al. 2007) provides a professional guideline on how to use CFD in nuclear reactor safety analysis, as well as a broad overview of such applications including some special cases such as Pipe Wall Erosion, Natural Conversion, Thermal Cycling. Some more general applications (e.g., Coolant Mixing, Stratified Flow, Hot Channel in the core, are discussed in Reference (H¨ohne, Krepper and U.Rohde 2009). Computational Multi-Fluid Dynamics (CMFD) is another most promising trend for nuclear reactor safety analysis. But due to the lack of flow schemes modeling as well as some other difficulties summarized by Reference (Bestion, D. 2014), this process is heavy, long and expensive. Reference (Yadigaroglu 2005) also emphasize those difficulties but narrows its eyesight to specific cases such as condensation of large bubbles in a pool of water. Nevertheless, it is still a promising trend thanks to the massive application of powerful and cheap computer clusters (Use and Development of Coupled Computer Codes for the Analysis of Accidents at Nuclear Power Plants 2003) and the constant effort on CMFD study.

To enhance the description and prediction of thermal-hydraulic phenomena in a more precise manner than what was done by isolated application of single codes, different thermal-hydraulic codes on different simulation scales are coupled together and such multi-scale codes and analysis have been initiated on different teams worldwide. The typical correlational research was carried out by the 6th and 7th European Framework Programs NURESIM and NURISP, in which several multi-scale couplings and simulations were developed to better investigate Pressurized Thermal Shock (PTS), Critical Heat Flux (CHF), et al. Reference (Calvin and Nowak 2010)also presents some cases with the aid of multi-scale tools such as Steam or Feed Lines Break. But it also highlights some of the multi-scale and multi-physics codes and teasing out the coupling methods. Other coupling aspects (e.g., classifications of the coupling are also investigated in Reference (Use and Development of Coupled Computer Codes for the Analysis of Accidents at Nuclear Power Plants 2003) and (D'Auria, et al. 2004). Various coupling of system thermal-hydraulic codes sub-channel codes have already been performed using different methodologies, (e.g., RELAP5/CORBA (Jeong, et al. 1997), CATHARE2/TRIO_U-MC (Anderhuber, et al. 2015), RELAP7 (Zhang, et al. 2014). Moreover, coupling of system codes and CFD-codes were also well investigated, (e.g., RELAP5-3D/CFX (Aumiller, Tomlinson and Bauer 2001), ATHLET/FLUENT (Macek and Vyskoci 2013), CATHARE/ TRIO_U (Bavière, et al. 2013).

At the Karlsruhe Institute of Technology (KIT), the multi-scale investigations are devoted to the coupling of system codes with sub-channel and CFD codes. The system thermal-hydraulic code TRACE has already been successfully coupled with the open-source CFD code TrioCFD using a standardized Interface for Code Coupling (ICoCo). The validity and rationality of the codes have been validated with a VVER-1000 coolant mixing benchmark (Zhang, et al. 2020).

1.2 Scope of the Report

This report is subdivided into seven chapters. The first chapter talks more about the motivation of the multi-scale work. The second chapter describes some basic knowledge of TrioCFD and ICoCo. The third chapter exhibits the elemental modification to TRACE and the development of ICoCo for TRACE. The detailed coupling issues are presented in Chapter 4. The validation of the coupled TRACE/TrioCFD code is discussed in Chapter 5. Finally, the main conclusions and outlook are summarized in Chapters 6 and 7. The technical details are exhibited in the appendix.

2 TRIOCFD FOR THE MULTI-SCALE COUPLING SYSTEM

Though ICoCo is already a highly implemented module embedded in TrioCFD, significant modifications are still mandatory to bring ICoCo which locates in the depths of TrioCFD source onto the logical surface. ICoCo is the most essential concept to be explained in this section since it is the cornerstone of implementing the coupling code. Nevertheless, before that, an overview of TrioCFD is the priority to be delivered.

2.1 Overview of TrioCFD

TrioCFD is an open-source CFD code based on the TRUST-platform (TRio_U Software for Thermo-hydraulics) being developed by CEA for scientific and industrial applications related to the nuclear industry (Bieder and Graffard 2007). TRUST is developed at the CEA/DEN/DANS/DM2S/STMF service. The project starts in 1994 and improved versions were built ever since:

1) 1994: start of the project Trio_U
2) 01/1997: v1.0 (VDF only, Finite Difference Volume)
3) 06/1998: v1.1 (VEF version, Finite Element Volume)
4) 04/2000: v1.2 (parallel version)
5) 07/2001: v1.3 (radiation model)
6) 11/2002: v1.4 (new LES turbulence models)
7) 02/2006: v1.5 (VDF/VEF Front Tracking)
8) 10/2009: v1.6 (data structure revamped)
9) 06/2015: v1.7 (cut into TRUST and TrioCFD + switch to open source)

TRUST was derived from cutting the Trio_U software into two pieces. Trio_U was a software brick based on the Kernel brick (which contains the equations, space discretizations, numerical schemes parallelism ...) and used by other CEA applications (Figure 2-1).

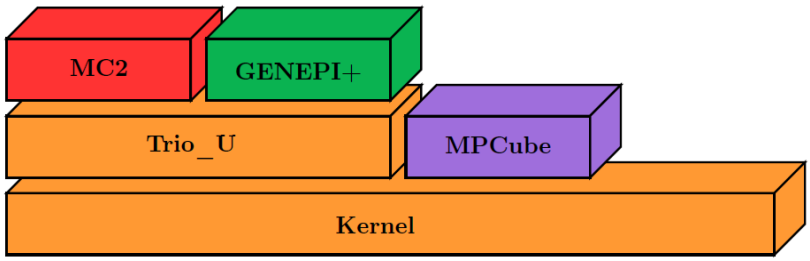


Figure 2-1 Trio_U: Brick Software

In 2015, Trio_U was divided into two parts: TRUST and TrioCFD.

- TRUST is a new platform, its name means: "TRio_U Software for Thermohydraulics",
- TrioCFD is a project based on TRUST, which contains the following models: FT, Radiation, LES, etc.

The relationship of TRUST and TrioCFD is exhibited in Figure 2-2.

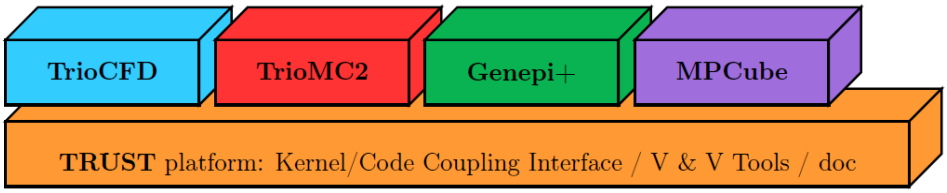


Figure 2-2 TrioCFD and the TRUST Platform as Base as Well as Other Main Projects Based Ton RUST

TrioCFD includes many physical models and it applies advanced numerical methods to solve various problems varying from local two-phase flows to turbulent flows on industrial facilities such as a nuclear reactor or part of it. In TrioCFD, the Reynolds-Averaged Navier-Stokes (RANS), the Large Eddy Simulation (LES), and the Direct Numerical Simulation (DNS) are available for the solution of fluid dynamic problems. The governing equations are solved with a staggered finite-volume approach. TrioCFD can generate robust meshes or import meshes from other software. The code supports full parallelepiped and tetrahedral structured or unstructured meshes. The spatial discretization of the parallelepipeds is based on Finite Volume Differences (V.D.F.) and of the tetrahedron is based on Finite Volume Elements (V.E.F.). A summary of the models applied by TrioCFD is listed below.

1) Incompressible single-phase flow
   a. Laminar or Turbulent flow.
   b. Navier Stokes with or without energy equation.
   c. Incompressible fluid or with a low variation for volumic mass.
2) Heat exchange
   a. Conduction.
   b. Radiation in a transparent medium.
   c. Radiation in the semi-transparent medium.
3) Transport of passive scalars
4) Porous Media
   a. Surface or volume porosities.
   b. Singular or regular pressure loss.
5) Particles transport model
   a. One-way coupling - particle motion affected by the flow.
   b. Two-way coupling - is above but particle disturbances also affect the flow and it is possible to convert droplet/bubble below a given size into particles during a Front Tracking calculation.
6) Front tracking model
   a. Two phases flow:
      · Eulerian mesh where Navier Stokes equations are solved.
      · Lagrangian moving mesh for the interface locations.
      · Coalescence or breakup models for bubbles and drops.
   b. Can be declined in TrioCFD to use an Immersed Boundary Method using IBC (Immersed Boundary Conditions).

TrioCFD is an object-oriented code written in C++ and it is composed of over 1500 classes. The data structure and functions of TrioCFD are appropriate for massively parallel computing where the data transfer and communication between the cores use the MPI protocol (Message Passing Interface).

Extensive verification and validation of TrioCFD are underway using different test data (Angeli, Bieder and Fauchet 2015). Typical applications of TrioCFD are (e.g., boron mixing, main steam-line break, and induced break severe accident (Bieder and Graffard 2007).

In the last years, TrioCFD was coupled with other codes and for this purpose powerful mesh-manipulation library such as the MEDCoupling and the coupling interface ICoCo (Interface for Code Coupling) were developed. It is worth to note that compulsive modifications and re-compilation of the source code are required to adapt the stand-alone TrioCFD code version for coupling with (e.g., system thermal-hydraulic code CATHARE (Bavière, R.; Tauveron, N.; Perdu, F.; Garré, E.; Li, S. 2014).

Since TrioCFD is open-source and already applied for nuclear engineering equipped with powerful in-built libraries such as the MEDCoupling and the generic built-in coupling-oriented and object-oriented interface named ICoCo, it has been selected for the multi-scale developments in the frame of this study. The experience gained in different European projects (e.g., NURISP and NURESAFE (NURESAFE 2019) devoted to multi-physics and multi-scale code coupling and the unique and innovative features of the code coupling approach based on the SALOME-platform, ICoCo, and the MEDCoupling libraries (SALOME, SALOME-the open source integration platform for numerical simulation 2019) have been the major reasons to rely on these methodologies. On the other hand, open-source codes are not comparable with commercial CFD-codes regarding numerical stability, user-friendliness, computational efficiency when solving very large problems. A comparison of the major CFD codes according to this report´s scope is given in Appendix A.

2.2 The Interface for Code Coupling (ICoCo)

ICoCo was selected as the interface for the multi-scale coupling of TRACE and the CFD code instead of ECI because it has some more advanced features. A simple comparison of ICoCo and ECI is discussed in Appendix B. Also, ICoCo is already a built-in module embedded in TrioCFD source, the introduction of ICoCo and some relevant knowledge about the coupling system are described as a part of this chapter where TrioCFD is dissected.

ICoCo is an object-oriented generic interface for code coupling developed under the scope of the European NURISP project (Deville and Perdu 2012). It was first used by the multi-scale thermal-hydraulic coupling of CATHARE and TrioCFD (Bavière, R.; Tauveron, N.; Perdu, F.; Garré, E. 2013). The codes in an ICoCo-based coupling system act as objects computing a time-dependent simulation. ICoCo specifies several methods that the problem has to provide as well as what they are supposed to do. It does not contain any real functional codes but poses a framework and a standard. Developers have to fulfill the frame and establish the connections between ICoCo and the target codes.

As the supplement of basic functions, ICoCo also supplies the methods to insert various input and output ports to the coupled-codes making the inter-code interaction flexible and convenient. Additionally, a supervisor has to be present to coordinate the coupled codes. This is a typical server-client system and its general form is shown in Figure 2-3. ICoCo is a cross-language interface (TRACE is in Fortran while TrioCFD is in C++) and hybrid-programming might be necessary when developing such coupling systems.

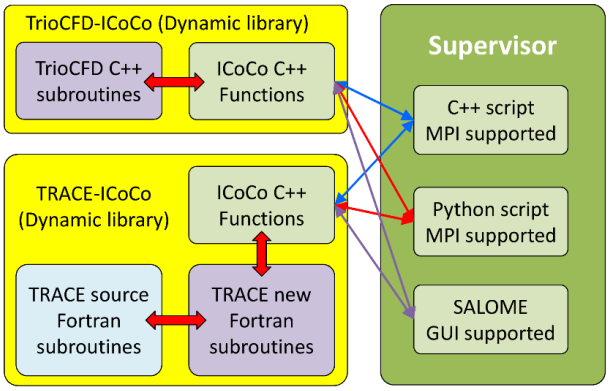


Figure 2-3 The Coupled Codes TRACE/TrioCFD with ICoCo Coordinated by the Supervisor

Three options are available for the supervisor: C++, Python, and SALOME. The first two are scripts. The third, SALOME, is an open-source integration platform for numerical simulation. It provides free pre- and post-processing functions, and supplies powerful functions for code coupling (SALOME, SALOME-the open source integration platform for numerical simulation 2019). For the first two options, users have to develop the supervisor scripts in either Python or C++, while for SALOME, the built-in YACS (SALOME, SALOME - YACS module 2019) module offers a Graphic User Interface (GUI) to organize the calculation chains and computational routes by “drag” and “click” using the mouse. SALOME is the best choice for small-scale solvers because users can monitor the calculation conveniently. However, due to the poor support of MPI protocol, SALOME is not able to manage simulations where a parallel capability is necessary. Instead, either C++ or Python script is a good choice. Since MPI is important for TrioCFD, it plays a key role in the supervisor selection. In this paper, the C++ script was selected as the supervisor because it not only supports MPI but there is a large community with accumulated experience and expertise using MPI. It is worth noting that though Python supports MPI, this capability was not that widely used as C++. Nevertheless, Python is still a promising alternative benefiting from its rapidly growing user community.

Field mapping is always one of the critical challenges for code coupling. ICoCo inherently adopts a powerful mesh-field manipulating library named MEDCoupling to handle mesh and field mapping issues. MEDCoupling is a build-in module in both SALOME and TrioCFD. Because of the peculiarities of the MEDCoupling library and the request of flexible coupling, finally, two essential prerequisites are necessary for a successful coupling according to the ICoCo standards:

- Each code should have explicitly-defined meshes which can be used for post-processing and field mapping;
- Each code has to be split into several functional components. Some of the key functions of ICoCo are listed inTable 2-1.

Table 2-1 Some of the Main Functions Defined by ICoCo

setDataFile	Specify the input file, its location, output files' location, and argument.
initialize	Do the initialization and generate the MED-format mesh in either file or memory.
Presenttime	Present the current computer time.
computeTimeStep	Calculate the time-step value for the current step.
initTimeStep	Initialize the time-step size.
solveTimeStep	Run the current time-step.
getOutputMEDField	Extract the desired field in MED format and write the field to the MED-format mesh.
getOutputFieldsNames	Get the name of the extracted MED-format field.
getInputFieldsNames	Get the name of a MED-format field from another code.
setInputMEDField	Translate and write the MED-format field from another code to the code's memory.
terminate	Terminate the execution.

These two requisites are mandatory for each code participating in the ICoCo-based coupling.

3 TRACE FOR THE MULTI-SCALE COUPLING SYSTEM

3.1 Overview of TRACE

TRACE is the reference best-estimate thermal-hydraulic system code of the U.S. Nuclear Regulatory Commission (NRC) for Light Water Reactors (LWR). A system of six balance equations in the two-fluid formulation plus additional equations to describe the transport of boron dissolved in the liquid phase and of non-condensable in the gas phase is solved for one-dimensional and three-dimensional components used to represent a nuclear power plant (TRACE TRACE V5.1051 theory manual n.d.). Besides, correlations for heat transfer in all relevant heat transfer modes of vertical and horizontal flow regimes are implemented together with a heat conduction solver for structures with and without a heat source. TRACE contains different components such as PIPE, VALVE, VESSEL, HTSTR, CHAN, POWER, CONTAN to represent various parts or systems of a Nuclear Power Plant (NPP). In all components except the 3D VESSEL, a one-dimensional single and two-phase flow can be simulated. State equations for water and steam as well as for other coolants (e.g., Sodium, CO2, Lead, Lead-Bismuth are also included. Dedicated models are also available for the description of critical flow, thermal stratification, counter-current flow, etc).

Both stationary and time-dependent thermal-hydraulic problems can be solved by TRACE. Moreover, neutronic kinetics is also implemented in TRACE using a point kinetics model. It is also coupled with a three-dimensional diffusion solver, named PARCS.

3.2 The Development of ICoCo for TRACE

As it was discussed in section 2.2, two prerequisites are necessary if ICoCo module is to implement for TRACE:

1) TRACE has to be equipped with meshing in a specific format (MED-format) since the spatial mapping of the two involved solution domains relies on the mesh superposition (MED-format) and the functionalities of the MEDCoupling library. The data of calculation parameters of each solver stored in the fields of the meshing is used for both feedback-exchange among the involved solvers and for post-processing purposes.
2) Each solver, (e.g., TRACE, must be modularized and split from one integral executable to several functional components (static or dynamic libraries) for flexible inter-communication and synchronization with the other solvers).

These two aspects are presented and discussed in the following sub-sections. The technical details on the code source can be found in Appendix C.

a) Development of the MED-Format Mesh for TRACE

For a successful coupling, physical field mapping between different code meshes is one of the critical issues, which largely determines the efficiency and even validity of the coupling. But before trying to solve this challenge, the meshes have to be properly and explicitly defined as the essential prerequisites, no matter in a file or the memory.

Almost all CFD codes have their meshes, which are logically isolated from the physical fields and the numerical definition of a problem. But for TRACE, the mesh is implicitly defined in the input file which contains a mixture of numerical data and mesh description. Hence, a meshing based on MED format must be explicitly developed for TRACE. Since the coupling between TRACE and TrioCFD focuses on the phenomena inside the RPV, a meshing for the 3D VESSEL component of TRACE is developed.

A VVER-RPV-model has been utilized to verify the newly-developed mesh generation function of TRACE. Three kinds of meshes are generated:

1) Figure 3-1 presents the polyhedron-cell or normal-cell TRACE mesh, which is for the post-processing and interpolation of cell-centered data (e.g., the coolant temperature, the pressure, and the coolant density);
2) Figure 3-2 displays the tetra-cell TRACE mesh. The TRACE vessel model is composed of lots of tetrahedron cells which is different from the polyhedron cell as the normal-cell TRACE mesh. Thanks to the good support of the tetrahedron cell by MEDCoupling, the development of the tetra-cell TRACE mesh is not that tricky than the normal-cell TRACE mesh. The motivation of this work is to simplify the mesh interpolation between different meshes;
3) Figure 3-3 displays the edge TRACE mesh. This kind of mesh is derived from the normal-cell TRACE mesh and is used to store and interpolate edge-based physical fields.

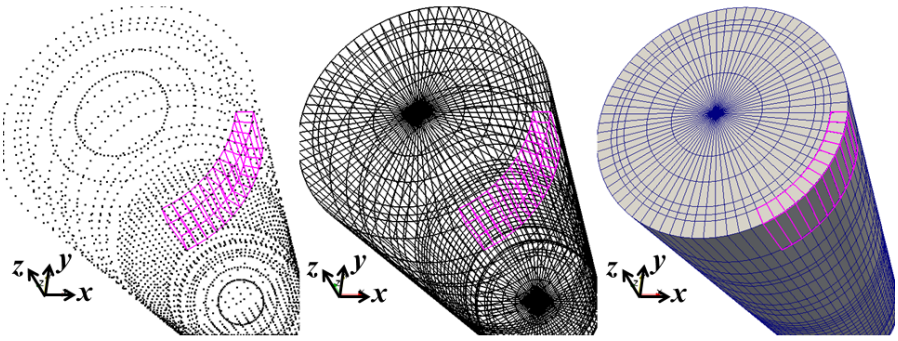


Figure 3-1 MED Normal-Cell-Mesh of TRACE for VVER-1000 Reactor

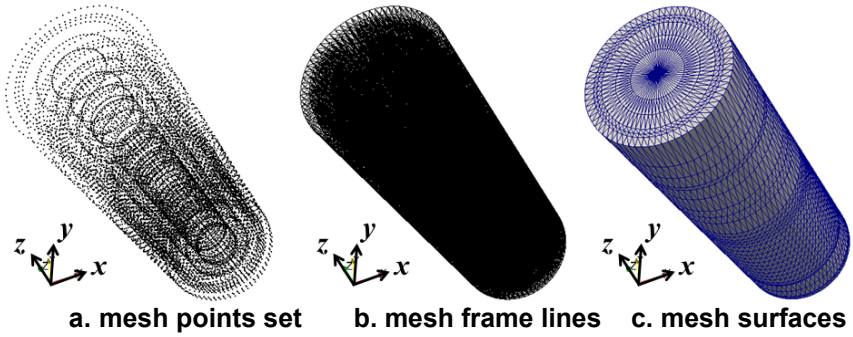
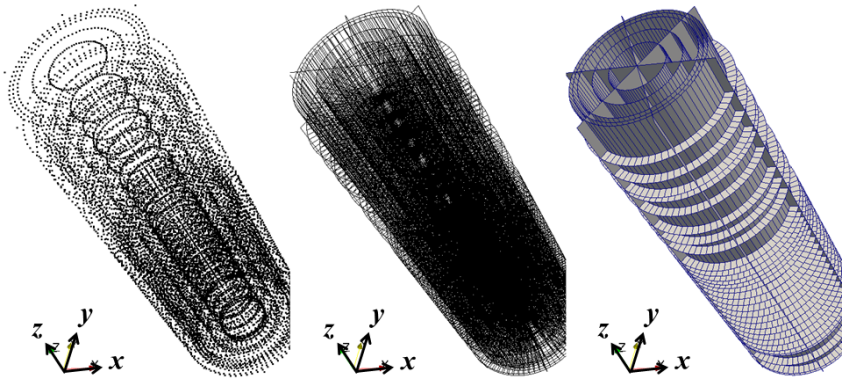


Figure 3-2 MED Tetra-Cell-Mesh of TRACE for VVER-1000 Reactor



a. mesh points set **b. mesh frame lines** **c. mesh surfaces**

Figure 3-3 MED Edge-Mesh of TRACE for VVER-1000 Reactor

The normal-cell mesh is used to store the cell-based fields (e.g., density, temperature for post-processing purposes). Pay attention that the purple-line-covered volume in Figure 3-1 stands for a single cell unit while not the simple cubic as shown in the figure. The tetra-cell mesh is developed for mesh-interpolation and coupling purposes since the interpolation tools of MEDCoupling can't recognize normal TRACE cells (fan-shaped and annular cells) and the best-recognized interpolation type is the tetrahedron. The edge mesh has to be developed for both post-processing and interpolation of edge-data (e.g., velocity and mass flow). Completed with the development of some other key functions, various kinds of data are written to the three meshes for both post-processing and field-interpolation purposes. At the last of this section, it is to note that the typical TRACE meshes, no matter in annular form or fan form, are approximated with groups of simple quadrangles or hexahedrons. Since this is an approximation, the resolution of the lines, edges, faces, and volumes could be manually defined by the users, which is quite a flexible feature for various purposes.

b) Development of the ICoCo Functional Components for TRACE

There are two steps to develop the ICoCo functional components for TRACE:

1) Re-organize the TRACE source to reflect the ICoCo functions.
2) Develop the C++ envelope to wrap the TRACE Fortran computing engine.

Finally, the TRACE-ICoCo module is developed and their descriptions are listed in Table 3-1.

Table 3-1 The ICoCo Functions for TRACE

setDataFile	inputPath	Declare the path of the input files.
	outputPath	Declare the directory where to put the output files.
	fileName	Specify the input file's name.
	argument	Input the necessary argument to control the calculation.
initialize	origin_X_m	The offset on x coordination of the generated mesh.
	origin_Y_m	The offset on y coordination of the generated mesh.
	origin_Z_m	The offset on z coordination of the generated mesh.
	theta_grad	The rotation degree of the generated mesh.
	return	Return the total transient problem time defined in the input file.

Presenttime	No input or output ports.	
computeTimeStep	run	-1 – this step will be skipped. 1 – normal condition, go ahead.
	return	Positive value – the calculated time-step size for the current step. -1.0 – the current step will be skipped.
initTimeStep	dt	Positive value – The time-step size for the current step. -1.0 – the current step will be skipped.
	return	The current problem time.
solveTimeStep	Run	-1 – this step will be skipped. 1 – normal condition, go ahead.
	return	0 – keep the iteration. 1 – stop the code. 2 – repeat the current time step. 3 – for steady-state, it means TRACE convergences; for transient, begin the iteration right now.
getOutputMEDField	field_name	The desired fields for output.
	internal_time	0.0 – write the field into the mesh for each time-step. Non 0.0 – only write the field into the mesh for the internal-time.
	height	0.0 – a three-dimensional field is desired. Non 0.0 – a two-dimensional field is desired and the field is derived from the given height of the vessel.
	return	The desired field.
getOutputFieldsNames	field	The entire field in MED format.
	return	The name of the input field.
getInputFieldsNames	field_name	The name of the input field.
	return	The name of the field.
setInputMEDField	field_name	The name of the input field.
	src_field	The entire field in MED format.
IsStationary	field	The field to be compared, for instance at a current time step, the previous field will be automatically saved.
	error	The user input convergence error criteria.
	step	The user-defined of how many steps should be skipped to compare the two fields.
	stop	Tell this routine if it is time to delete the keep-in-memory array so that to clear its occupied memory.
	return	A bool indicating whether the field converges or not.
terminate	No input or output ports.	

For TRACE standalone (Figure 3-4), all of the sources including subroutines and modules (mainly in Fortran) are compiled to a single executable that can directly run a simulation. Compared to the original TRACE (Figure 3-4), the structures of TRACE with ICoCo now involve more levels (Figure 3-5). To develop the ICoCo module for TRACE, the following work has to be implemented:

1) Re-organize the TRACE source to reflect the ICoCo functionalities. The new Fortran source of TRACE is represented by the first inner pie of Figure 3-5;
2) Since the ICoCo functions can not directly access TRACE data with TRACE original subroutines, new Fortran subroutines have to be developed acting as the data-transfer interface between TRACE and ICoCo. Those new subroutines are represented by the inter-ring of Figure 3-5.

The TRACE source is wrapped by the newly developed Fortransubroutines and flexible communications (bidirectional arrows) are possible between them;

3) The TRACE reorganized source and the new subroutines are compiled to a static library. The library now has suitable input and output ports with the help of the new Fortran subroutines;
4) According to the ICoCo framework, each of the functions listed in Table 3-1 has to be developed. They are in C++. These ICoCo functions are represented by the outer ring of Figure 3-5;
5) The ICoCo wrapper together with the TRACE computation engine (the static library involving TRACE reorganized source and the newly developed Fortran subroutines) are compiled to a dynamic library. This TRACE-ICoCo library now is ready to be called by the supervisor.

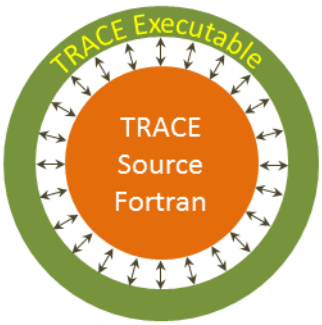


Figure 3-4 The Original Structure of TRACE with One Executable

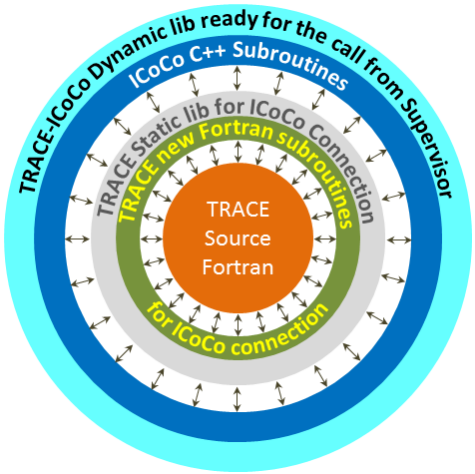


Figure 3-5 The Structure of TRACE with Newly Developed Fortran Subroutines and the C++ ICoCo Wrapper

3.3 The Dynamic-Implicit-Additional-Source (DIAS) Method for TRACE

Unlike the traditional coupled pairs that apply the domain-decomposition method and transfer data through pre-defined 2D interfaces between two separate domains, the coupling system of TRACE and TrioCFD directly overlap their 3D domains and perform the 3-dimensional field mapping or interpolation. This is a domain-overlapping approach and it will be detailed in the next chapter. One of the significant differences between this method and the domain-decomposition method is that TRACE will not only use the 2D field from the other code as the new-time boundary condition but has to handle the 3D field coming from the other code, here is TrioCFD. To solve this problem, a so-called Dynamic-Implicit-Additional-Source (DIAS) method was developed especially on TRACE's side within the coupling system (Zhang, Kanglong; Sanchez-Espinoza, Victor 2019).

In general, the DIAS method uses the 3D coolant velocity field, the 3D coolant pressure field from TrioCFD to correlate the corresponding two fields in TRACE within the overlapped area. Similarly, the coolant temperature field and the boron dissolution field will be correlated based on the corresponding field from TrioCFD as well.

a) Coolant Velocity and Pressure Correlation

Take the 1D components in TRACE for instance, the governing motion equation whose items could be characterized to four types of the pressure drop as shown in Equation 1. Similar items could also be derived for sub-channel and CFD codes (here we use the expression including the sub-channel code besides CFD because the DIAS is a general approach which can utilize fine fields from not only CFD code but also sub-channel code, this kind of expression will be kept in the following texts), Equation 2. From left to right, they are:

a) ΔP_{acc} – The acceleration pressure drop or inertial pressure drop which comes from the variation of coolant velocity along the problem time.
b) ΔP_{con} – The convective pressure drop which comes from the mass and momentum transfer along the flow path.
c) $-\Delta P_{tot}$ – The total static pressure drop which is also known as the manometer pressure drop. The minus sign stands for its negative correlations with the other three pressure drops.
d) ΔP_{fric} – The friction pressure drop which is caused by the friction between the fluid and the solid structures.

$$\frac{V_{j+\frac{1}{2}}^{n+1}-V_{j+\frac{1}{2}}^{n}}{\Delta t}+V_{j+\frac{1}{2}}^{n}\cdot\left.\frac{\partial \tilde{V}^{n+1}}{\partial x}\right|_{j+\frac{1}{2}}+\frac{1}{\langle\rho\rangle_{j+\frac{1}{2}}^{n}}\cdot\frac{\tilde{P}_{j+1}^{n+1}-\tilde{P}_{j}^{n+1}}{\Delta x}+K_{j+\frac{1}{2}}^{n}\cdot\left(2V_{j+\frac{1}{2}}^{n+1}-V_{j+\frac{1}{2}}^{n}\right)\cdot\left|V_{j+\frac{1}{2}}^{n}\right|=0$$

$$\Delta P_{acc}^{TRACE}+\Delta P_{con}^{TRACE}+\left(-\Delta P_{tot}^{TRACE}\right)+\Delta P_{fric}^{TRACE}=0 \qquad \textbf{Equation 1}$$

$$\Delta P_{acc}^{Sub/CFD}+\Delta P_{con}^{Sub/CFD}+\left(-\Delta P_{tot}^{Sub/CFD}\right)+\Delta P_{fric}^{Sub/CFD}=0 \qquad \textbf{Equation 2}$$

The two corresponding terms in Equation 1 are replaced with that from the fine results, as Equation 3 shows. The reason why the acceleration pressure drop can't be erased is because of the ineradicable time term which explicitly binds two adjacent time layers. To counteract the corresponding acceleration pressure drop $\Delta P_{acc}^{Sub/CFD}$ was dropt from the total static pressure drop $\Delta P_{tot}^{Sub/CFD}$: sub-channel or CFD codes. The new added two pressure drops could be further

integrated into one fake friction pressure drop $\Delta P_{fric,fake}^{TRACE} = \Delta P_{con}^{Sub/CFD} + \Delta P_{fric}^{Sub/CFD} = \Delta P_{tot}^{Sub/CFD} - \Delta P_{acc}^{Sub/CFD}$ by imposing a fresh friction factor $C_{j+\frac{1}{2},fake}^{n}$ which equals $\Delta x \cdot \rho \cdot K_{j+\frac{1}{2},fake}^{n}$. Now the TRACE motion governing equation becomes Equation 4.

$$\Delta P_{tot}^{TRACE} = \Delta P_{acc}^{TRACE} + \Delta P_{con}^{Sub/CFD} + \Delta P_{fric}^{Sub/CFD} = \Delta P_{acc}^{TRACE} + \left(\Delta P_{tot}^{Sub/CFD} - \Delta P_{acc}^{Sub/CFD}\right) \quad \textbf{Equation 3}$$

$$\frac{V_{j+\frac{1}{2}}^{n+1} - V_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{1}{\langle\rho\rangle_{j+\frac{1}{2}}^{n}} \cdot \frac{\tilde{P}_{j+1}^{n+1} - \tilde{P}_{j}^{n+1}}{\Delta x} + \frac{C_{j+\frac{1}{2},fake}^{n}}{\Delta x \cdot \langle\rho\rangle_{j+\frac{1}{2}}^{n}} \cdot \left(2V_{j+\frac{1}{2}}^{n+1} - V_{j+\frac{1}{2}}^{n}\right) \cdot \left|V_{j+\frac{1}{2}}^{n}\right| = 0 \quad \textbf{Equation 4}$$

The fake friction pressure drop $\Delta P_{fric,fake}^{TRACE}$ in Equation 4 is proportional to the product of $C_{j+\frac{1}{2},fake}^{n}$ and the square of velocity. Thus, the fresh fake coefficient could be calculated by Equation 5.

$$C_{j+\frac{1}{2},fake}^{n} = \frac{\Delta P_{fric,fake}^{TRACE}}{V^2} = \frac{\Delta P_{tot}^{Sub/CFD} - \Delta P_{acc}^{Sub/CFD}}{V^2} \quad \textbf{Equation 5}$$

With the determined $\Delta P_{fric,fake}^{TRACE} \Delta P_{con}^{Sub/CFD} + \Delta P_{fric}^{Sub/CFD} = \Delta P_{tot}^{Sub/CFD} - \Delta P_{acc}^{Sub/CFD}$, the remaining problem is to choose the right velocity so that the right fake friction coefficient which reflects the flow's physical resistance parameters could be calculated.

For a real flow in a determined geometry, (all physical parameters were fixed (e.g., the structure shape and the wall roughness, etc.) with determined outlet pressure and inlet flow velocity, the flow conditions (e.g., the velocity and pressure fields should be unique for most cases). To make sure the correlated TRACE could produce consistent velocity distribution and overall pressure drop with the CFD code, the fake friction coefficients should also be the same as the real geometry. Thus, the velocity should come from the CFD code.

The hydraulic correlation approach presented here behaves similarly to the traditional Additional Source Term Method (ASTM). However, the difference is obvious: the latter method directly appends the difference between the right and to-be-correlated values to the motion equation forcing the correlated values to be equal to the expected data while the added source by the former method does not explicitly introduce the difference to the motion equation. It is calculated dynamically along with the problem timeline and implicitly "guide" the system code to approximate the fine-results.

b) Coolant Temperature

Different from the velocity and pressure correlations, which modify the friction coefficient within the time advancing iteration loop, the correlation to coolant temperature takes place deep in the linearization iteration loop, which is the inner loop inside a single time step calculation. The schematic of the TRACE numeric containing the two iteration loops is shown in Figure 3-6, whose details could be found in TRACE theory manuals.

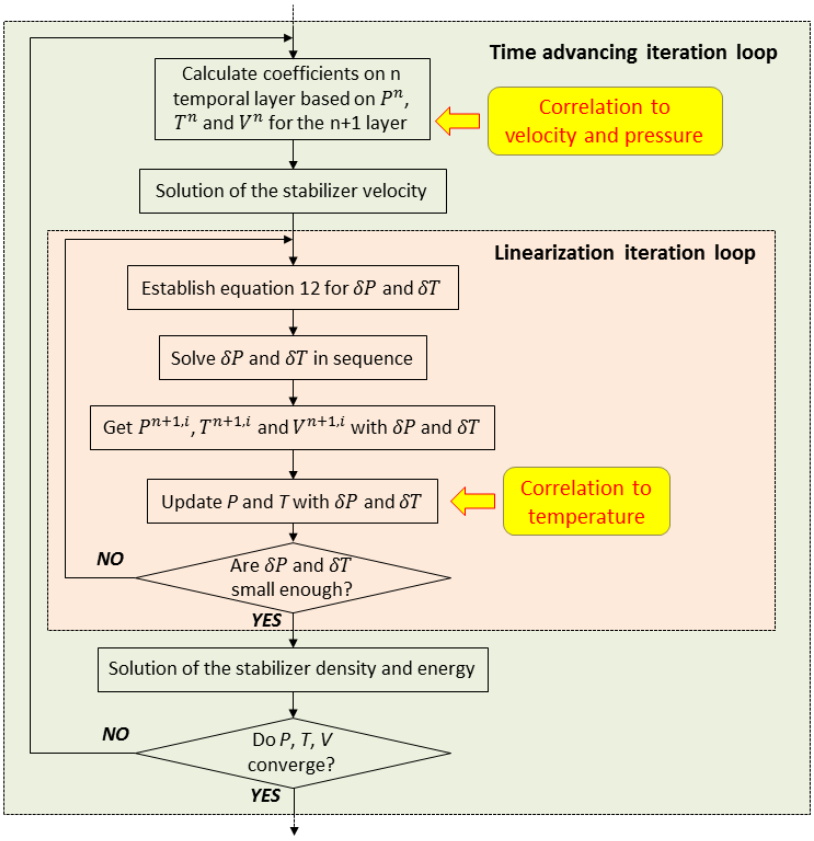


Figure 3-6 The TRACE SETS Numeric Which Contains Two Iteration Loops

The positions where the temperature and the hydraulic correlations take place are indicated in Figure 3-6. It could be figured out that the fake friction coefficient was correlated as the prior step before the launching of the solution process and the temperature correlation occurs within the linearization iteration loop, which strictly represents the semi-implicit step in the numerical method of TRACE. From the overall perspective, the new pressure and temperature got from the inner loop are used to solve the final stabilizer continuity and energy equations for the final density and energy at the current completed time step. The final pressure and temperature should be the result of further solution of the state equations based on the final density and energy. However, the computation was never performed to reduce the simulation resource burden. Normally, the pressure and temperature obtained from the semi-implicit step are good enough for updating related coefficients for the next time step.

It could be inferred from Figure 3-6 that the temperature variation – for a linearization step is the function of the sole pressure variation – , indicating an simple algebraic equation set in the form of Equation 6 which could be directly solved. The superscript *i* represents the step counter for the inner loop and the subscript *j* stands for the cell number in the model mesh. The temperature for the current linearization step is calculated by Equation 7. The tilde above the temperature indicates this is a variable in the semi-implicit step.

$$\begin{bmatrix} \delta T_1^i \\ \vdots \\ \delta T_j^i \end{bmatrix} = \begin{bmatrix} a_{1,1}^i & \cdots & a_{1,j}^i \\ \vdots & \ddots & \vdots \\ a_{j,1}^i & \cdots & a_{j,j}^i \end{bmatrix} \cdot \begin{bmatrix} \delta P_1^i \\ \vdots \\ \delta P_j^i \end{bmatrix} + \begin{bmatrix} b_1^i \\ \vdots \\ b_j^i \end{bmatrix} \quad \textbf{Equation 6}$$

$$\begin{bmatrix} \tilde{T}_1^{n+1,i+1} \\ \vdots \\ \tilde{T}_j^{n+1,i+1} \end{bmatrix} = \begin{bmatrix} \tilde{T}_1^{n+1,i} \\ \vdots \\ \tilde{T}_j^{n+1,i} \end{bmatrix} + \begin{bmatrix} \delta T_1^i \\ \vdots \\ \delta T_j^i \end{bmatrix} \quad \textbf{Equation 7}$$

The goal of the correlation is to let the final coolant temperature at the current time step equal the fine results. As a plain idea, the operation could be performed once the temperature was fixed after the inner iteration to force the fresh temperature to equal that from other codes. However, this is an incondite manipulation and the information of the right temperature fields was forbidden to propagate during the inner iteration where the coefficients (e.g., *a* in the matrix were updated for each inner step based on the calculated temperature for the current inner step). Considering this effect, the temperature correlation was moved forward to produce consistent temperature fields with sub-channel or CFD codes for each inner step calculation.

The correlation works in a rather simple way by replacing the calculated temperature $T_{TRACE}^{n+1,i+1}$ on each inner-iteration with the sub-channel or CFD results – $T_{Subchannel/CFD}^{n}$ straightforwardly. Nevertheless, the hidden rationale could be the addition of a virtual source – S as shown in Equation 8. The source could be perceived as an auto-self-adapted and dynamically calculated source set according to the difference between $T_{Subchannel/CFD}^{n}$ and $T_{TRACE}^{n+1,i+1}$, though nothing was calculated here. The direct replacement of $T_{TRACE}^{n+1,i+1}$ by $T_{Subchannel/CFD}^{n}$ could be regarded as the results of a dynamic and implicit regulation by the virtual source, which reflects the core concept of the Dynamic-Implicit-Additional-Source (DIAS) method. It is worth to note here that the coolant temperature field in the entire overlapped domain will be correlated to be identical with that from sub-channel or CFD codes, which is similar to the hydraulic correlations.

$$\begin{bmatrix} \tilde{T}_{1,TRACE}^{corr,n+1,i+1} \\ \vdots \\ \tilde{T}_{j,TRACE}^{corr,n+1,i+1} \end{bmatrix} = \begin{bmatrix} T_{1,Sub,CFD}^{n} \\ \vdots \\ T_{j,Sub,CFD}^{n} \end{bmatrix} = \begin{bmatrix} \tilde{T}_{1,TRACE}^{n+1,i+1} \\ \vdots \\ \tilde{T}_{j,TRACE}^{n+1,i+1} \end{bmatrix} + \begin{bmatrix} S_1^i \\ \vdots \\ S_j^i \end{bmatrix}$$
$$= \begin{bmatrix} \tilde{T}_{1,TRACE}^{n+1,i} \\ \vdots \\ \tilde{T}_{j,TRACE}^{n+1,i} \end{bmatrix} + \begin{bmatrix} a_{1,1}^i & \cdots & a_{1,j}^i \\ \vdots & \ddots & \vdots \\ a_{j,1}^i & \cdots & a_{j,j}^i \end{bmatrix} \cdot \begin{bmatrix} \delta P_1^i \\ \vdots \\ \delta P_j^i \end{bmatrix} + \begin{bmatrix} b_1^i \\ \vdots \\ b_j^i \end{bmatrix} + \begin{bmatrix} S_1^i \\ \vdots \\ S_j^i \end{bmatrix} \quad \textbf{Equation 8}$$

c) Boron Concentration

Due to the weak coupling relationship of the solute concentration with other key thermal-hydraulic parameters, the solubility equation is not included in the core governing equation set. Nevertheless, solute especially boron plays an essential role in some scenarios thus should also be correlated in the system code based on the fine sub-channel or CFD data. A similar approach was developed for the solute concentration correlation as that to the coolant temperature correlation, which directly replaces the TRACE data with fine results in the overlapped domains. This manipulation could be treated as adding a dynamic and implicit additional source term at the right hand of the solubility equation. As a linear system, the solution matrix can be solved directly.

The solute correlation is carried out as the last item of the DIAS approach, which locates right after the solution of final energy and density in the time advancing loop. The correlation could force TRACE to reproduce identical solute distributions in the entire overlapped region as the fine data from the other two codes. Moreover, it can be stated that though the DIAS approach correlates the hydraulic parameters (coolant velocity and pressure), the coolant temperature, and the solute concentration by appending some kind of dynamic and implicit additional sources to the corresponding equations, the additional source for the hydraulic equation is a particular one. It correlates the coolant velocity and pressure synchronously and “guide” the system code to produce a better result. The additional source to thermal and solubility equations only correlate one parameter and just “force” the system code to reproduce consistent fields with that from the sub-channel or CFD codes.

4 DESCRIPTION OF THE ICOCO-BASED COUPLING OF TRACE AND TRIOCFD

It was known in section 2.2 that the coupling system based on ICoCo is coordinated by a supervisor, which could be a C++ script, a Python script, or the SALOME platform. Here, in this case, the C++ script was selected because it supports MPI well. When the coupled code is running, two key items determine whether the right result could be got. They are discussed here below.

4.1 The Data Transfer Between TRACE and TrioCFD Thermal-Hydraulic Domain

In section 3.2, the different kinds of meshes developed for TRACE have been discussed, namely:

1) The normal-cell mesh or polyhedron-mesh (Figure 4-1a) represents the real TRACE's cells in fan-shape or annular-shape. It is used for storing the normal cell-based data (e.g., the coolant temperature, the pressure, etc. for post-processing);
2) The tetrahedron-mesh (Figure 4-1c) is used for the mesh or field interpolation and mapping process for cell-based data and it is especially used for the coupling purposes;
3) The edge-mesh (Figure 4-1b) describes all the inner faces within the TRACE vessel component. It is first used for the edge-based field interpolation and mapping (e.g., the coolant velocity, the pressure drop. It is also used for the edge-based data post-processing).

All TRACE-meshes in this chapter are based on the VVER-1000 nuclear power plant model.

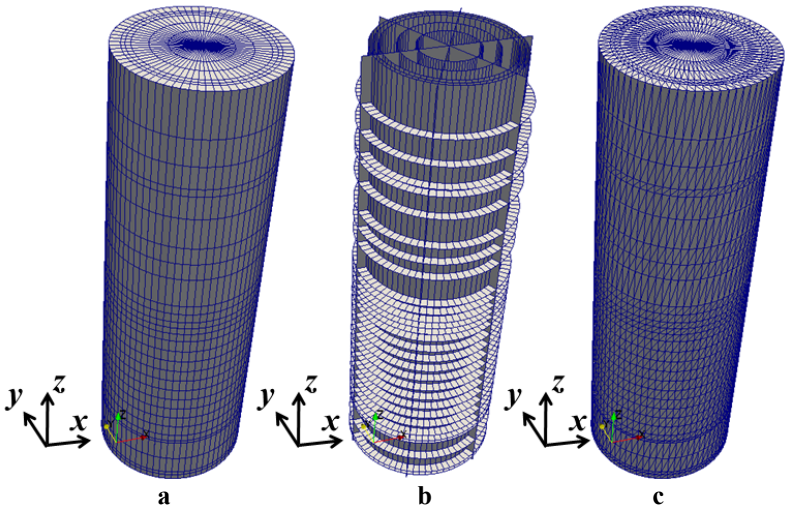


Figure 4-1 Geometric Structure of the Normal-Cell Mesh (a), Tetrahedron-Cell Mesh (b), and Edge Mesh (c) for TRACE

For demonstration and explanation, TrioCFD only simulates the downcomer while TRACE represents the whole reactor vessel. So the overlapped region only covers the downcomer region. Besides, the TrioCFD model is simplified to a cylindrical shell as shown in Figure 4-2. This simplified TrioCFD model is only used for general illustration purposes. So no geometrical or mesh details are given.

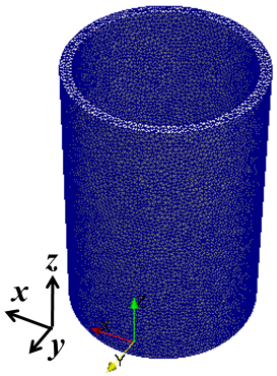


Figure 4-2 Downcomer Mesh of TrioCFD for Demonstration of Data Flow Within the Coupled Code TRACE/TrioCFD

Since the DIAS method includes correlations to four system parameters in the overlapped domain, (e.g., the coolant velocity, the pressure, the coolant temperature, and the boron concentration, the data transfers between TRACE and TrioCFD are correspondingly specified into two categories):

1) The first one includes both the edge mesh and tetrahedron-cell mesh because the correlations to the two hydraulic parameters are performed simultaneously;
2) The second one is for the correlations of the coolant temperature and boron concentration where the tetrahedron-cell mesh is involved.

Figure 4-3 presents the data flow for hydraulic correlations of the coupled code. The data flow follows the procedures listed below for the hydraulic parameter correlations.

1) TRACE runs a time step forward and saves the three-dimensional pressure fields in the normal-cell mesh for post-processing and in the tetrahedron-cell mesh for the field mapping in the next step. In the meantime, the three-dimensional coolant velocity fields are saved in the edge mesh for both post-processing and field mapping purposes.
2) With the help of the MEDCoupling library, insert the upper boundary 2D edge mesh of TrioCFD into the TRACE's edge mesh and derive the desired inlet velocity boundary conditions for TrioCFD by performing the edge-to-edge mesh interpolation. Then, insert the lower boundary 2D edge mesh of TrioCFD into the TRACE's tetrahedron-cell mesh and derive the desired outlet pressure boundary conditions for TrioCFD by performing the cell-edge mesh interpolation.
3) With the newly updated inlet and outlet boundary conditions, TrioCFD runs a step forward and write all of the calculated fields in its mesh.
4) TRACE assemblies the tetrahedron-cell and edge meshes to the TrioCFD 3D mesh, derives the three-dimensional pressure and velocity fields, and writes them to the tetrahedron-cell and edge meshes.
5) Since the tetrahedron-cell mesh does not represent the real TRACE cells, the pressure field in it is first integrated and then written to the normal-cell mesh.
6) The pressure field in the normal-cell mesh is first used to calculate the pressure drop across each edge and then the pressure drop field together with the velocity field is used by the DIAS method to correct the friction and form losses coefficients at the edges.
7) With the updated parameters, TRACE movers a step forward and generate the fresh new fields.

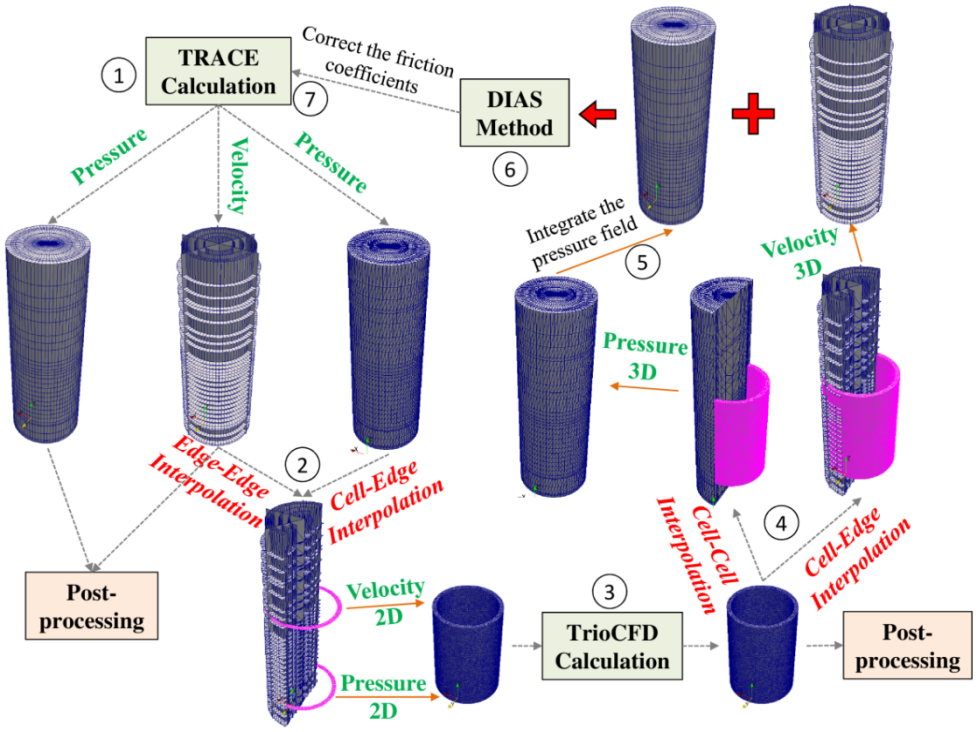


Figure 4-3 The Data Flow for Hydraulic Correlations Between TRACE and TrioCFD Within the Coupling System

Figure 4-4 presents the data flow for coolant temperature and boron concentration correlations. The data flow follows the procedures listed below for the coolant temperature and born concentration correlations.

1) TRACE runs a time step forward and saves the three-dimensional coolant temperature or born concentration fields in the normal-cell mesh for post-processing and the tetrahedron-cell mesh for the field mapping in the next step.
2) With the help of the MEDCoupling library, insert the upper boundary 2D edge mesh of TrioCFD into the TRACE's edge mesh and derive the desired inlet coolant temperature or born concentration boundary conditions for TrioCFD by performing the cell-edge mesh interpolation.
3) With the new updated inlet boundary condition, TrioCFD runs a step forward and write all of the calculated fields in its mesh. Those fields are for both post-processing and field mapping.
4) TRACE assemblies its tetrahedron-cell mesh to the TrioCFD 3D mesh, derives the desired three-dimensional coolant temperature or born concentration fields, and writes them back to the tetrahedron-cell mesh.
5) Since the tetrahedron-cell mesh does not represent the real TRACE cells, the coolant temperature or born concentration fields in it is first integrated and then written to the normal-cell mesh.
6) The coolant temperature or born concentration field is used by the DIAS method to correct the corresponding fields in TRACE at each cell within the entire overlapped domain.
7) With the updated parameters, TRACE moves a step forward and generates the new fields.

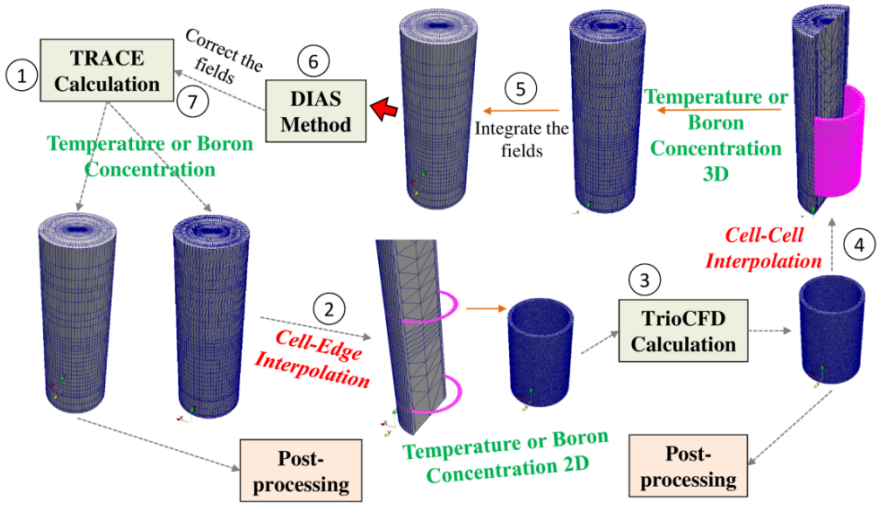


Figure 4-4 The Data Flow for Coolant Temperature and Boron Concentration Correlations Between TRACE and TrioCFD Within the Coupling

4.2 System The Explicit Temporal Coupling of TRACE/TrioCFD

According to the discussion in the previous chapters and sections, the ICoCo-based multi-scale coupled code TRACE/TrioCFD is a server-client system coordinated by a C++ supervisor in a parallel manner. The mesh and field mapping and interpolation are handled by the third-party mesh-field – MEDCoupling library, which is closely related to SALOME and it is an in-built module of TrioCFD. The data synchronization during the time advancement between the involved solvers is based on an explicit approach. The overview of the explicit temporal coupling scheme for TRACE/TrioCFD based on ICoCo is displayed in Figure 4-5. The calculation procedure coordinated by the C++ supervisor is described hereafter:

1) The supervisor launches both TRACE and TrioCFD.
2) The two codes read in their input file and do the initialization at the same time. The MPI setting up is finalized if necessary.
3) The two codes run into their time step loops and they calculate first their time step size. Then, the supervisor gathers the two-time step size, selects the smaller one, and sends the public time step back to the two codes. With that, the two codes reset their current time step.
4) The codes inform the supervisor which field they expect from the other code and prepare specified templates for the mesh-interpolation or field-mapping processes. The supervisor sends the desired fields' names to the involved solvers.
5) The codes recognize and check the fields' names from the other code and depending on the names, the corresponding fields are extracted from the current codes' memories. The supervisor receives the physical fields and interpolates them with the already-prepared templates thus generating the final fields. Then they are sent to the destinations.
6) The codes receive their desired fields from the supervisor and use them to update the conditions for the current time step.
7) The codes calculate their current time step and iterate unless the problem time is achieved.
8) Once the problem is done, the supervisor terminates both TRACE and TrioCFD runs and then it finalizes the whole execution.

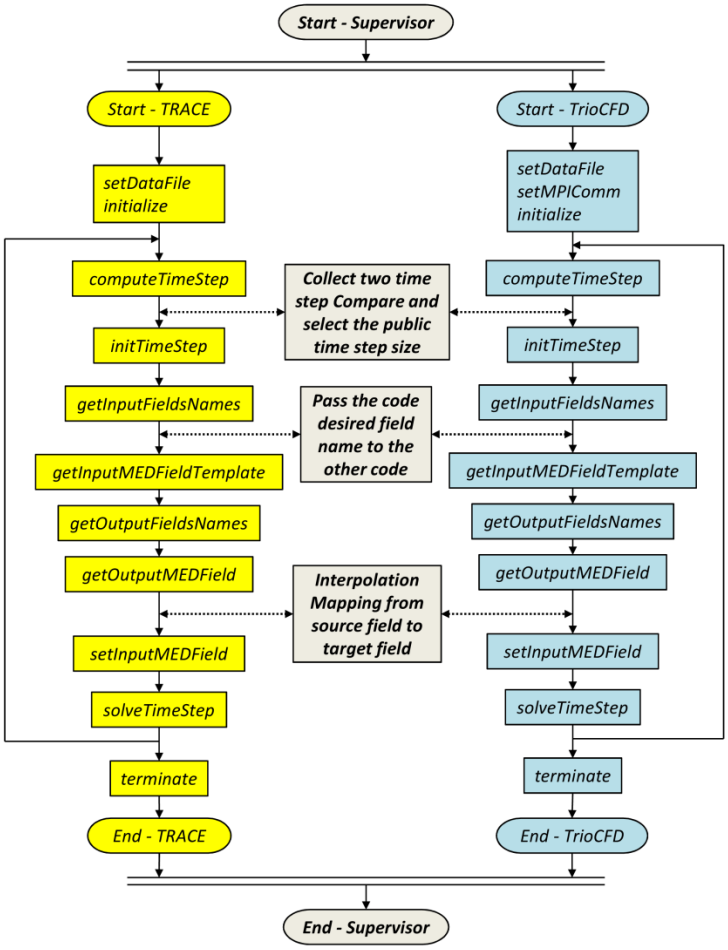


Figure 4-5 Schematics of the Execution of the Explicit Temporal Coupling Scheme of TRACE/TrioCFD based on ICoCo

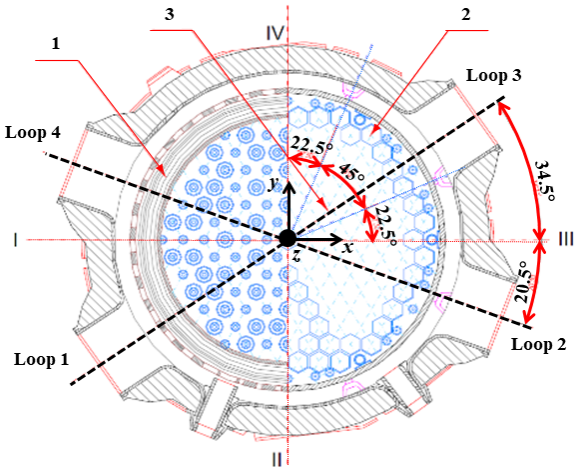
In this explicit temporal coupling approach, the data transfer is only performed within each time step. The implementation of a semi-implicit temporal coupling is not a significant task because only the supervisor script needs to be re-organized while the ICoCo functions shown in Figure 4-5 remain untouched.

5 VALIDATION OF THE COUPLED CODE

The validation of the new TRACE/TrioCFD capability is done by the analysis of the VVER-1000 coolant mixing benchmark problem. Selected parameters predicted by the coupled code are compared with TRACE standalone and measured data at the Kozloduy nuclear power plant.

5.1 Description of the VVER-1000 Coolant Mixing Experiment

The coolant mixing experiment was performed during the commissioning phase of Kozloduy NPP to study the mixing of loop flows in the reactor vessel of VVER-1000 V320 (Kolev, et al. 2010). In the test, the steam isolation valve was closed and the steam generator was isolated resulting in the heat-up of primary coolant in the sector of the downcomer linked to the affected steam generator (first loop). Due to the coolant mixing in the downcomer, the temperature of the other primary loop is partly also affected. The mixing pattern established in the downcomer propagates to the core where also coolant mixing takes place. VVER-1000 is a four-loop pressurized water reactor with hexagonal core geometry and horizontal steam generators. The core is open-type and contains 163 hexagonal fuel assemblies. The location of the main inlet and outlet nozzles of the reactor vessel is non-uniform in the azimuthal direction and asymmetric concerning the core symmetry axes. The cross-section sketch of the reactor vessel is described in Figure 5-1.



1 – Block of Shielding Tubes (BST)
chimney 2 – Fuel assemblies
3 – Lower spacer grid
LEFT – Cut at hot legs
RIGHT – Cut at cold legs

Figure 5-1 Vessel Cross-Sectional Sketch of VVER-1000 Including the Vessel Inlet and Outlet

The main initial operating parameters before the test are summarized in Table 5-1. The transient test consists of the following events:

1) Closure of the steam isolation valve and isolation of the steam generator from feed water of loop 1;
2) Coolant temperature at the cold leg of loop 1 increases by about 14 degrees, Figure 5-2;
3) Coolant mixing occurs first in the downcomer;

4) Coolant mixing takes place in the lower plenum, core and upper plenum;

Coolant temperature at the other three loops all increase by different degrees, see Figure 5-2 and Figure 5-3.

Table 5-1 Main Initial Operating Parameters Before Test

Parameter	Initial State	Accuracy
Thermal power, MW	281	± 60
Pressure above the core, MPa	15.593	± 0.3
Pressure drop over RPV, MPa	0.418	± 0.043
Coolant temperature at core inlet #1, K	541.75	± 1.5
Coolant temperature at core inlet #2, K	541.85	± 1.5
Coolant temperature at core inlet #3, K	541.75	± 1.5
Coolant temperature at core inlet #4, K	541.75	± 1.5
Coolant temperature at core outlet #1, K	545	± 2.0
Coolant temperature at core outlet #2, K	545	± 2.0
Coolant temperature at core outlet #3, K	544.9	± 2.0
Coolant temperature at core outlet #4, K	545	± 2.0
Mass flow rate of loop #1, kg/s	4737	± 110
Mass flow rate of loop #2, kg/s	4718	± 110
Mass flow rate of loop #3, kg/s	4682	± 110
Mass flow rate of loop #4, kg/s	4834	± 110

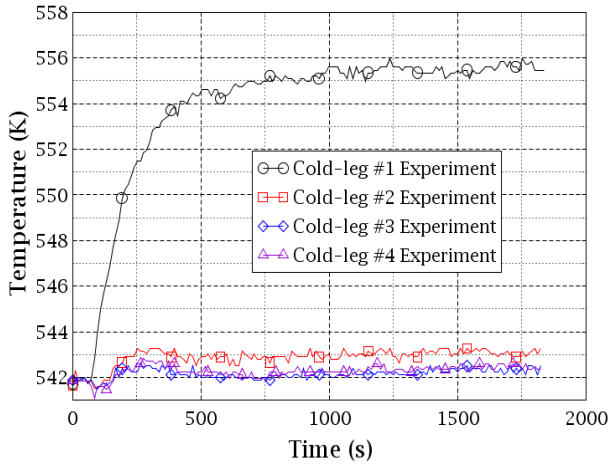


Figure 5-2 Measured Evolution of the Coolant Temperature at Cold Legs During the Transient of the VVER-1000 Coolant Mixing Benchmark

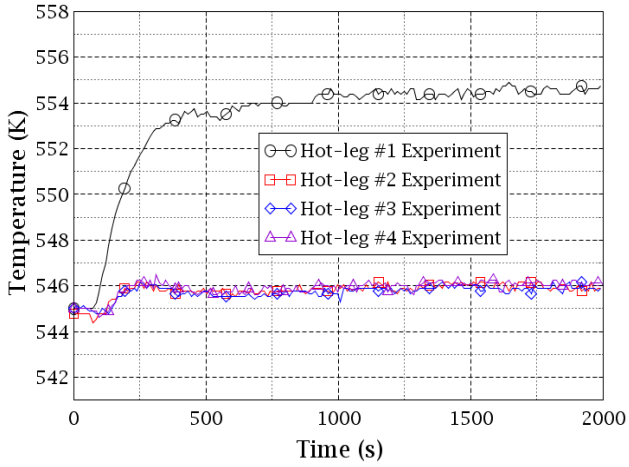


Figure 5-3 Measured Evolution of the Coolant Temperature at Hot Legs During the Transient of the VVER-1000 Coolant Mixing Benchmark

The test lasts for 1800 seconds and the final core power increased up to 286 MW. Due to the isolation of the steam generator 1, the heat transfer from the primary to the secondary side decreases leading to the increase of the coolant temperature on the cold-leg-1 and finally on the hot-leg-1. The coolant temperature of the other three loops also increases due to the coolant mixing in the downcomer and lower plenum. Reverse heat transfer occurs in the isolated steam generator #1. At about 300s, the temperature of cold-leg-1 exceeds that of hot-leg-1, see Figure 5-2 and Figure 5-3.

The difference stabilizes to 0.6~0.8 degrees in about 20 min. Cold-leg-2 has a slightly higher coolant temperature than cold-leg-3 and 4 during the transient, while the outflows on the four hot-legs almost have the same temperature, indicating the coolant well mix.

5.2 Description of the Thermal-Hydraulic Models of TRACE and TrioCFD

TRACE/TrioCFD employs a domain overlapping approach in which TrioCFD simulates the downcomer (Figure 5-4) in the vessel while TRACE simulates the whole vessel region (Figure 5-5).

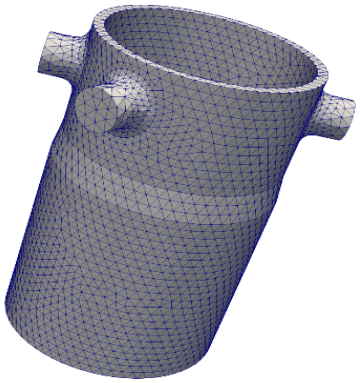


Figure 5-4 TrioCFD Downcomer-Mesh of the VVER-1000 Coolant Mixing Benchmark for the Validation of TRACE/ TrioCFD-ICoCo

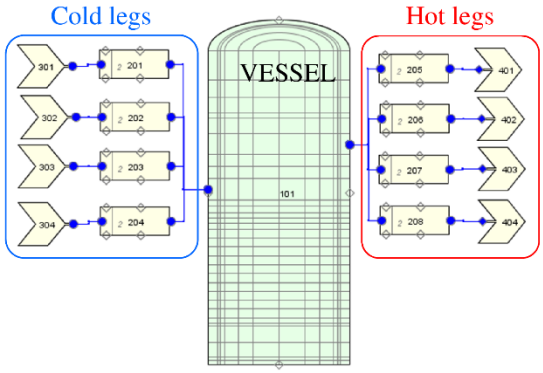


Figure 5-5 TRACE Model of the VVER-1000 Coolant Mixing Benchmark for the Validation of TRACE/ TrioCFD-ICoCo

Thanks to the implemented ICoCo functionalities, TRACE now has meshes for the VESSEL component. The assembled meshes of TRACE (here is the edge mesh) and TrioCFD are displayed in Figure 5-6 where an obvious overlapped thermal-hydraulic domain in the downcomer can be observed. In this case, the inlet boundary condition of TrioCFD is pre-defined in the input file. The outlet boundary condition of TrioCFD comes from TRACE dynamically through the 2D interface where mesh interpolation and field mapping are handled by the MEDCoupling library. The 3D volumetric fields of TrioCFD are translated by MEDCoupling and passed to TRACE to correlate its corresponding fields through the DIAS method.

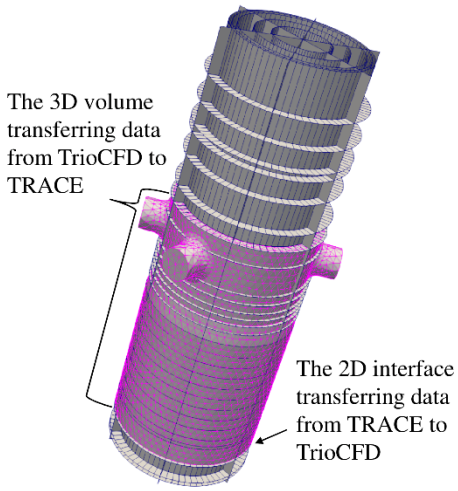


Figure 5-6 The Assembled Meshes of TRACE and TrioCFD of the VVER-1000 Coolant Mixing Benchmark for the Validation of TRACE/TrioCFD-ICoCo

According to the benchmark specification, the four loops are not symmetrically arranged. The real unsymmetrically arranged meshes are shown in Figure 5-7 where the TRACE inlets (blue arrows and lines) are positioned symmetrically according to TRACE azimuthal sectors (yellow texts) while TrioCFD inlets (red arrows and lines) shift an anticlockwise degree. This mesh arrangement precisely reflects reality. It is expected to predict the temperature distribution of the coolant flowing into the core in a more precise manner, by considering the unsymmetrical vessel geometry. However, this mesh arrangement leads to a mismatch between the vessel inlet and outlet nozzles, since the positions of the problem inlets are now defined by TrioCFD while the positions of the outlets are still by TRACE (TrioCFD simulates only the downcomer while the upper part of the vessel is still handled by TRACE). Due to the unsymmetrically arranged inlets and outlets, the assembled mesh of Figure 5-7 can't be used for the analysis of the coolant temperature distribution at the hot legs. Thus, the case where the TRACE mesh is also shifted an anticlockwise degree and symmetrically assembled to the TrioCFD mesh (see Figure 5-8) have to be simulated as well.

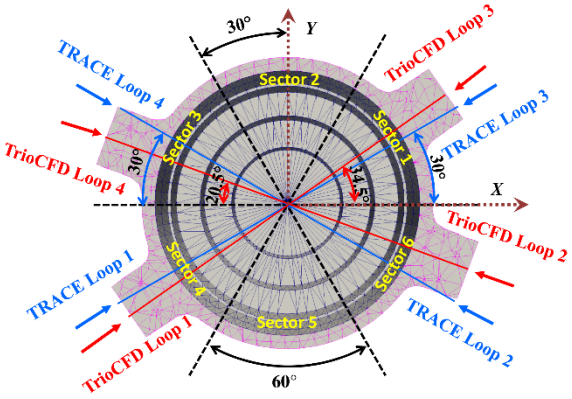


Figure 5-7 Real Unsymmetrically Assembled Meshes for the alidation of TRACE/ TrioCFD-ICoCo

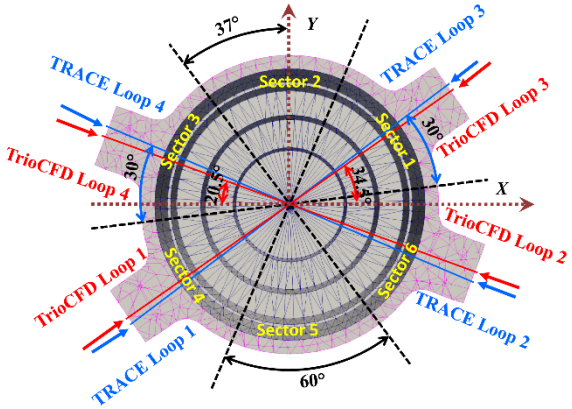


Figure 5-8 Idealized Symmetrically Assembled Meshes for the Validation of TRACE/ TrioCFD-ICoCo

The symmetrically assembled meshes are used to investigate the coolant temperature distribution at the hot legs while the unsymmetrically assembled meshes are used to investigate the coolant temperature distribution at the core outlets. It is worth to note that the symmetrical mesh-pair has been developed to ensure the position-matching of inlet and outlet nozzles. It is obsolete if TrioCFD also takes the upper plenum into account. However, since the upper plenum involves complex geometry which leads to difficult mesh, it is reasonable to use simple models to demonstrate the capabilities of the coupled code.

5.3 Discussion of the Selected Results

The correlated fields of TRACE within the coupled code during the simulation include the coolant temperature, the pressure, and the coolant velocity. The correlations are performed in the entire overlapped domain. Figure 5-9 illustrates the coolant temperature distributions on the fine downcomer mesh, the TRACE mesh, and the overlapped meshes in sequence. The coolant temperature matching between TRACE and TrioCFD is evident, indicating the temperature translation is executed correctly. It can be observed that the hot coolant enters the downcomer from one loop and gradually diffuses over a larger area along the main flow direction.

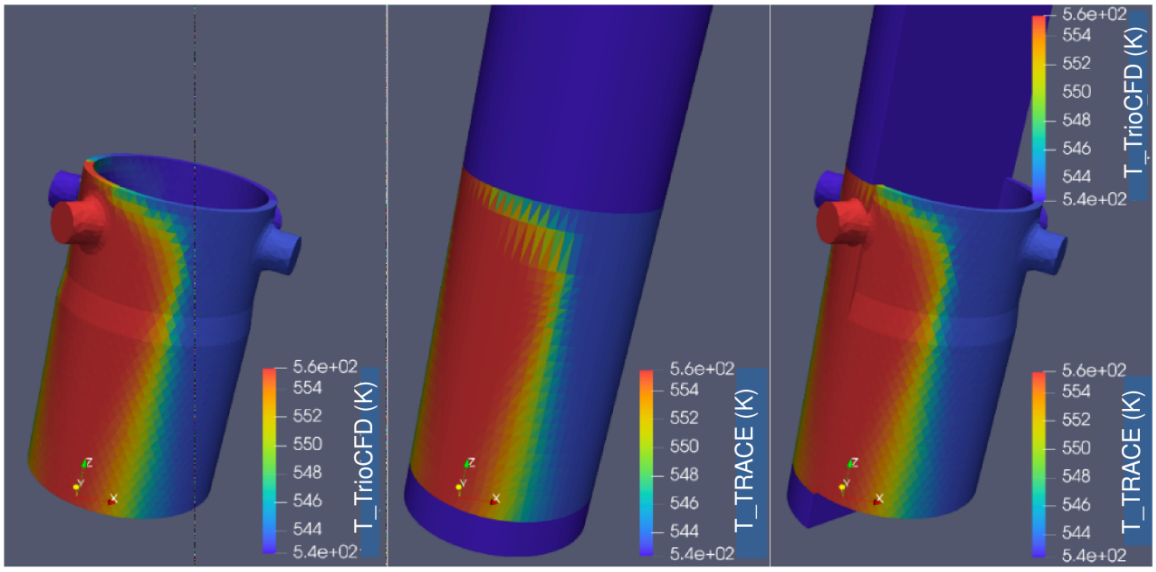


Figure 5-9 The Coolant Temperature Distribution in the Downcomer on Meshes of TRACE and TrioCFD

Figure 5-10 depicts the pressure distributions on the fine downcomer mesh, the TRACE mesh, and the overlapped meshes in sequence. The pressure matching between TRACE and TrioCFD is evident, indicating the pressure translation from TrioCFD to TRACE is executed correctly. The pressure decreases along the main flow direction.

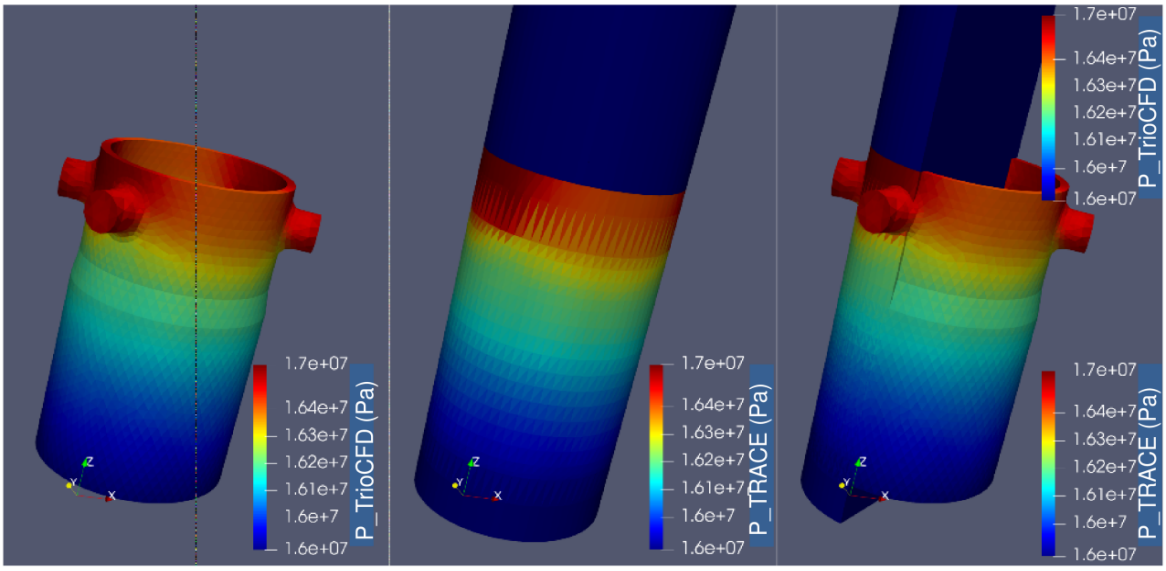


Figure 5-10 The Pressure Distribution in the Downcomer on Meshes of TRACE and TrioCFD

Figure 5-11 displays the coolant velocity distributions on the fine downcomer mesh, the TRACE mesh, and the overlapped meshes in sequence. The velocity matching between TRACE and TrioCFD is also evident, indicating the coolant velocity translation from TrioCFD to TRACE is executed correctly. The flow streamlines tell that the coolant enters the vessel through the four inlet nozzles and spreads to the downcomer immediately. The main flow is downward while a limited amount of flow is upward and then reverse to contribute to the downward flow.

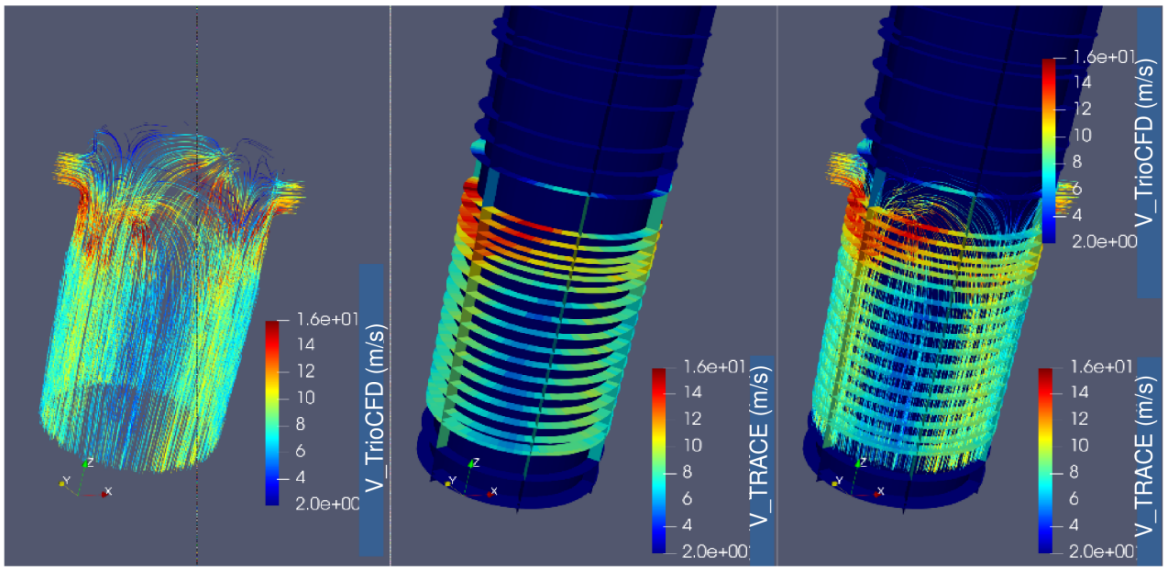
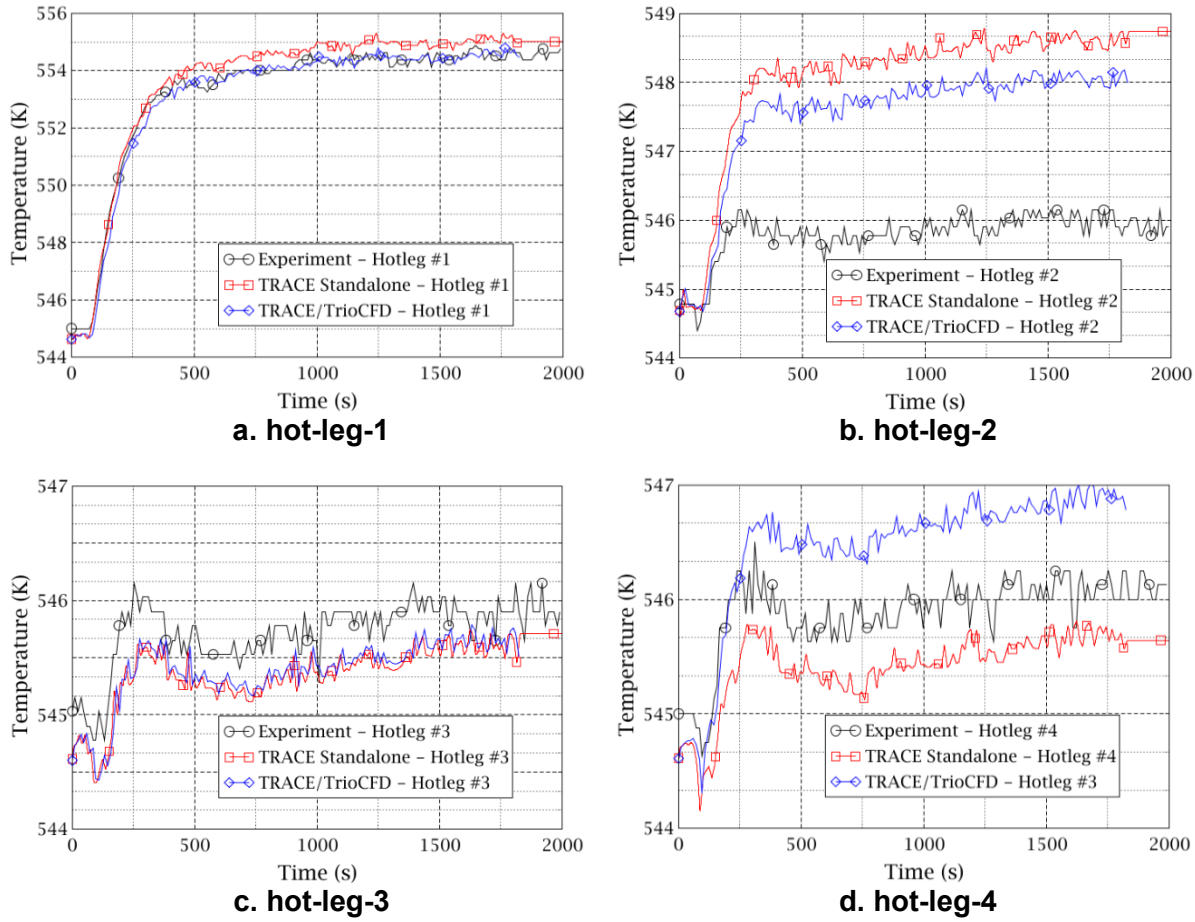


Figure 5-11 The Coolant Velocity Distribution in the Downcomer on Meshes of TRACE and TrioCFD

The coolant temperature at the four hot legs predicted by TRACE and TRACE/TrioCFD is compared to the measured data in Figure 5-12. The temperature rise of hot-leg-1 can be observed in Figure 5-12a, where TRACE standalone tends to over-predict the heat-up while the coolant temperature calculated by TRACE/TrioCFD is very close to the plant data. In Figure 5-12b, both of TRACE standalone and TRACE/TrioCFD over-predict the coolant temperature at hot-leg-2. Nevertheless, the coupled code exhibits an improvement and the curve is located closer to the measured data than that of TRACE standalone. At hot-leg-3 (Figure 5-12c), the predictions of TRACE and TRACE/TrioCFD are all not satisfactory, where the coolant temperature is both under-predicted compared to the measured data. In Figure 5-12d, TRACE standalone under-predicts the heat-up at hot-leg-4 while TRACE/TrioCFD over-predicts the heat up. It could be concluded that TRACE/TrioCFD performs better than TRACE standalone on the coolant temperature prediction at the hot-legs, though there is still room for improvement.



a. hot-leg-1 **b. hot-leg-2**

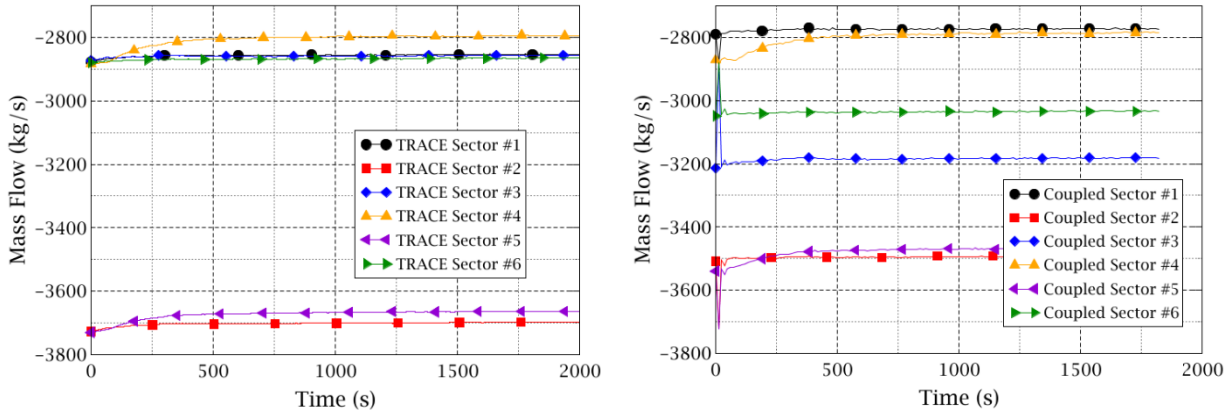
c. hot-leg-3 **d. hot-leg-4**

Figure 5-12 Comparison of Calculated Coolant Temperatures at the Hot-Legs of TRACE, TRACE/TrioCFD with the Measured Data

The encouraging result indicates TrioCFD indeed introduces a positive influence to TRACE in the overlapped domain within the coupled code. To more clearly repeal this effect, the coolant mass flow rate and temperature distribution at the downcomer outlet of TRACE within the coupled code are carefully inspected and compared with that of TRACE standalone. Figure 5-13 depicts the simulated coolant mass flow distribution. The curves on the left figure represent data from TRACE.

standalone. The curves on the right figure display the data from TRACE within the coupled code. The “sector” formulation in the graph legends corresponds to the six azimuthal sectors of the TRACE model (see Figure 5-7 and Figure 5-8).

The downward mass flow across sectors 1, 3, 4, and 6 of TRACE standalone are roughly the same and are between 2800 and 2900 kg/s, while the mass flow across sectors 2 and 5 are as large as 3700 kg/s. Compared with that, the mass flow predicted by TRACE within the coupled code is much flatter. Moreover, the inlet differences are reflected more clearly by the coupled code. This kind of hydraulic correlation is handled by the DIAS method which uses the coolant velocity and pressure fields in the entire overlapped domain to correlate the friction coefficient of TRACE, thus “guide” TRACE produce the quasi-CFD hydraulic fields.

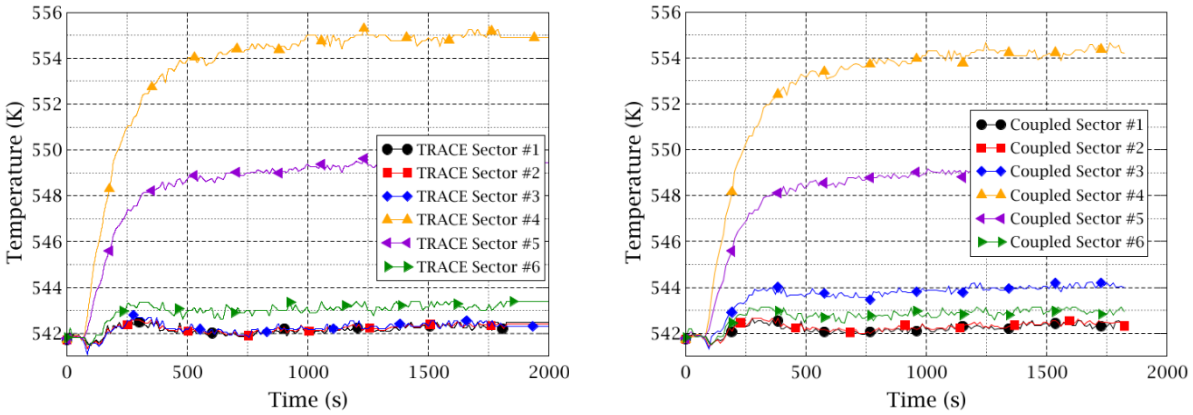


a. TRACE standalone

b. TRACE in the coupled codes

Figure 5-13 Computed Coolant Mass Flow Rate Distribution at the Downcomer Outlet of TRACE Standalone and TRACE Within the Coupled Code

Figure 5-14 illustrates the simulated coolant temperature distribution at the downcomer outlet as a function of time. The curves on the left figure represent data from TRACE standalone. The curves on the right figure display the data from TRACE within the coupled code. The “sector” formulation in the graph legends also corresponds to the six azimuthal sectors of the TRACE. Similar curve trends are observed as that at the four hot-legs shown in Figure 5-12. The coolant temperature on sector 4 (corresponds to loop 1) of TRACE within the coupled code is lower than that of TRACE standalone (Figure 5-12a depicts this phenomenon as well). The coolant temperature on sector 6 (corresponds to loop 2) of TRACE within the coupled code is slightly lower than that of TRACE standalone (Figure 5-12b depicts this phenomenon as well). The coolant temperature on sector 1 (corresponds to loop 3) of the two calculations shown no significant difference (Figure 5-12c depicts this phenomenon as well). The coolant temperature on sector 3 (corresponds to loop 4) of TRACE within the coupled code is higher than that of TRACE standalone (Figure 5-12d depicts this phenomenon as well). Additionally, the coolant temperature on sector 5 (the sector between loop 1 and loop 2) locates right in the middle of the graph scale telling that the heat-up of sector 6 (loop 2) comes from sector 4 (loop 1) through sector 5.



a. TRACE Standalone

b. TRACE in the Coupled Codes

Figure 5-14 The Coolant Temperature Distribution at the Downcomer Outlet of TRACE Standalone and TRACE Within the Coupled Code

The data analyzed by the previous part originate all from the symmetrically assembled meshes of TRACE and TrioCFD (see Figure 5-8). They are appropriate for the analysis of the coolant distribution at the hot-legs by omitting the mismatch of inlet and outlet nozzles. However, the inherent asymmetry between the vessel and the core is missing. This leads to a distortion of the flow conditions entering the core. Thus the actual case where the unsymmetrical assembled meshes are applied has to be simulated when the flow condition in the core area is concerned.

Figure 5-15 provides the calculated coolant temperature distribution predicted by TRACE standalone at the core outlet. A sharp temperature change over sectors 3 and 4 is observed indicating a weak coolant mixing between the two sectors. The high coolant temperature in sole sector 4 not only signifies the limited mass and heat transfer from this sector to others but is also partly due to the wrong location of the inlet nozzle defined by TRACE. By applying the correlation from TrioCFD to TRACE whose meshes precisely reflect the right VVER-vessel geometry, the coolant temperature at the core outlet now distributes much flatter (see Figure 5-16), which follows the profile of the measured data shown in Figure 5-17. There, the hottest coolant occupies both sector 4 and sector 5, and the centerline of the hottest part locates closer to the boundary of the two sectors. This more realistic coolant temperature distribution at the core outlet is believed to come from the right unsymmetrical geometry modeled by TrioCFD as well as the enhanced coolant mixing introduced from TrioCFD to TRACE in the overlapped downcomer region.

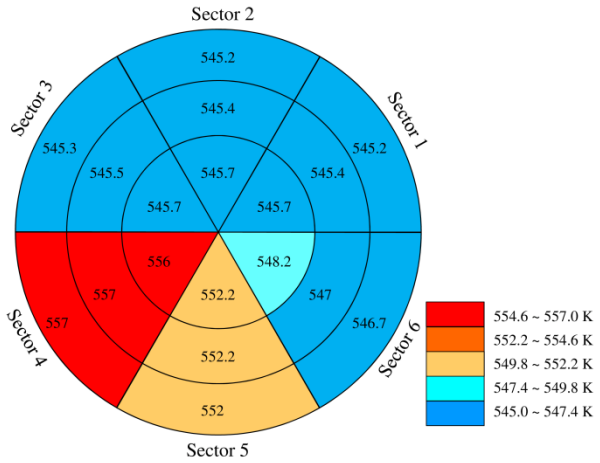


Figure 5-15 The Coolant Temperature Distribution at the Core Outlet Predicted by TRACE Standalone at the End of the Problem

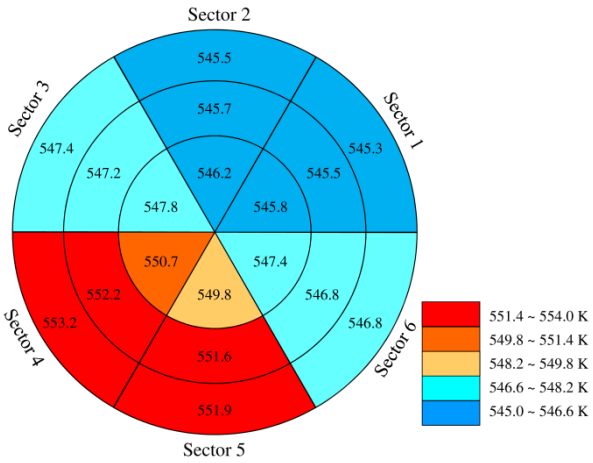


Figure 5-16 The Coolant Temperature Distribution at the Core Outlet Predicted by TRACE/TrioCFD at the End of the Problem

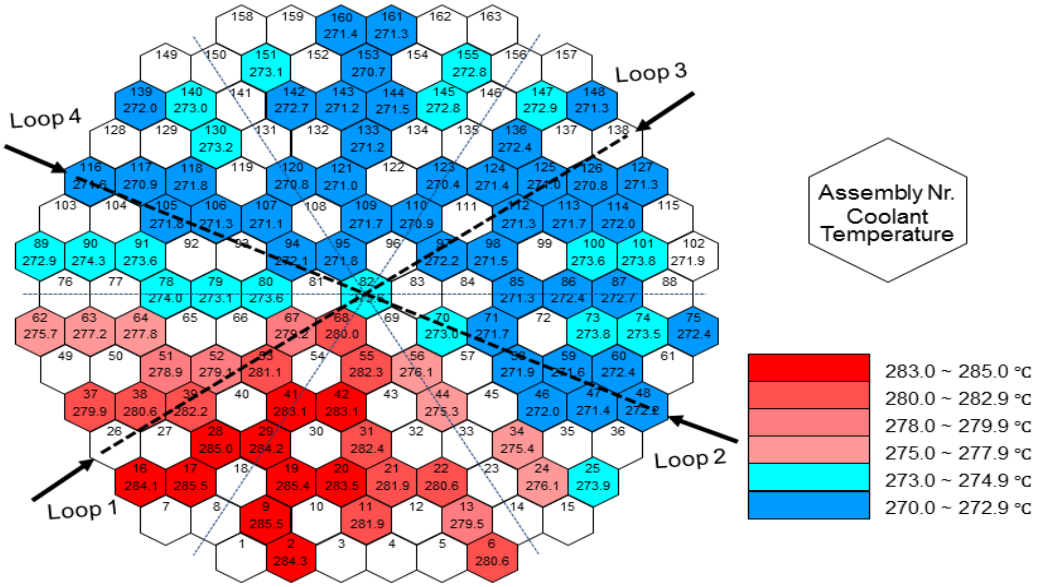


Figure 5-17 The Measured Coolant Temperature Distribution at the Core Outlet

6 CONCLUSION

This report presents the coupling of TRACE using the newly developed ICoCo-module with an open-source CFD code TrioCFD to improve the prediction of the thermal-hydraulic behavior in the reactor pressure vessel. The peculiarities of ICoCo and the prerequisites for ICoCo development are discussed. A new ICoCo-module was developed for TRACE with which TRACE now is equipped with various meshes for post-processing and field mapping. Meanwhile, TRACE now is divided into several functional components according to the ICoCo standard. A domain overlapping approach was adopted and the open-source MEDCoupling library is used for field mapping to ensure a consistent data transfer between the computational domains. The Operator Splitting (OS) coupling approach (temporal coupling) was implemented for data synchronization during the time advancement. The coupled code was validated with a VVER-1000 coolant mixing benchmark. The obtained results are very promising and indicate that the codes were properly coupled to each other.

7 OUTLOOK

The next steps are the selection of more experimental data performed under realistic conditions to validate the prediction capability of the new-coupled code and in this connection to further optimize the coupling regarding accuracy, usability, and robustness. Moreover, the multi-scale coupling system could extend to involve other physical pieces (e.g., the reactor neutronics. As a result, we will get a multi-scale and multi-physics coupling system).

8 REFERENCES

[1] Anderhuber, M., A. Gerschenfeld, N. Alpy, J. Perez, and JM. Seiler. 2015. "Simulation of GR19 Sodium Boiling Experiments with CATHARE 2 System Code and TRIO_U MC Subchannel Code." NURETH-16, 16th International Topical Meeting on Nuclear Reactor Thermal Hydraulics. Chicago, U.S.

[2] Angeli, P. E., U. Bieder, and G. Fauchet. 2015. "Overview of the TrioCFD code: main features. V&V procedures and typical applications to nuclear engineering."NURETH-16: The 16th International Topical Meeting on Nuclear Reactor Thermal Hydraulics, ISBN:978-0-89448-722-4. Chicago, U.S.: NURETH-16: The 16th International Topical Meeting on Nuclear Reactor Thermal Hydraulics, August 30-September 4, 2015, ISBN:978-0-89448-722-4, Ppaer ID:13006.252-265.

[3] Aumiller, D.L., E.T. Tomlinson, and R.C. Bauer. 2001. "A coupled RELAP5-3D:CFD methodology with a proof-of-principle calculation." Nuclear Engineering and Design 205 (1-2): 83–90.

[4] Bavière, R., N. Tauveron, F. Perdu, and E. Garré. 2013. "System-CFD Coupled Simulation of the PHENIX Reactor Natural Circulation Test." NURETH-15 The 15th International Topical Meeting on Nuclear Reactor Thermal - Hydraulics. Pisa, Italy.

[5] Bavière, R.; Tauveron, N.; Perdu, F.; Garré, E. 2013. "System-CFD coupled simulation of the PHENIX reactor natural circulation test." NURETH-15: The 15th International Topical Meeting on Nuclear Reactor Thermal - Hydraulics, ISBN 978-88-902391-2-0. Pisa, Italy: NURETH-15: The 15th International Topical Meeting on Nuclear Reactor Thermal - Hydraulics, May 12-17, 2013, ISBN 978-88-902391-2-0, Paper ID: 288.

[6] Bavière, R.; Tauveron, N.; Perdu, F.; Garré, E.; Li, S. 2014. "A first system/CFD coupled simulation of a complete nuclear reactor transient using CATHARE2 and TRIO U. Preliminary validation on the Phénix reactor natural circulation test." Nuclear Engineering and Design 277: 124-137. https://doi.org/10.1016/j.nucengdes.2014.05.031.

[7] Bestion, D. 2014. "The difficult challenge of a two-phase CFD modelling for all flow regimes."Nuclear Engineering and Design 279: 116–125.

[8] Bestion, D., D. Lucas, H. Anglart, B. Niceno, and L Vyskocil. 2012. "Multi-Scale Thermalhydraulic Analyses Performed in NURESIM and NURISP Projects." 20th International Conference on Nuclear Engineering and the ASME 2012 Power Conference. Anaheim, California, U.S.

[9] Bieder, U., and E. Graffard. 2007. "Qualification of the CFD code Trio_U for full scale reactor applications." Nuclear Engineering and Design 238 (3): 671-679, https://doi.org/10.1016/j.nucengdes.2007.02.040.

[10] Calvin, C., and D. Nowak. 2010. Handbook of Nuclear Engineering: 12 High-Performance Computing in Nuclear Engineering. Springer.

[11] Chavez, V. J., U. Imke, and V.H. Sanchez-Espinoza. 2018. "TWOPORFLOW: A two-phase flow porous media code, main features and validation with BWR-relevant bundle experiments."Nuclear Engineering and Design 338: 181-188.

[12] Chelemer, H., J. Weisman, and L.S. Tong. 1972. "Sub-channel thermal analysis of rod bundle cores." Nuclear Engineering and Design 21 (1): 35–45.

[13] Cheng, Z., and Y.F. Rao. 2015. "Strategies for Developing Subchannel Capability in an Advanced System Thermal hydraulic Code: A Literature Review." AECL Nuclear Review 4 (1): 23-41.

[14] D'Auria, F., A. Bousbia Salah, G.M. Galassi, J. Vedovi, F. Reventós, A. Cuadra, J.L. Gago, et al. 2004. Neutronics/Thermal-hydraulics Coupling in LWR Technology, State-of-the-art Report. Nuclear Energy Agency, Organisation for Economic Co-Operation and Development.

[15] D'Auria, F.; Galassi, G.M. 2010. "Scaling in nuclear reactor system thermal-hydraulics." Nuclear Engineering and Design 240 (10): 3267–3293.

[16] Deville, E., and F. Perdu. 2012. "Documentation of the Interface for Code Coupling: ICoCo.Report." Saclay, France: NURISP-NUclear Reactor Integrated Simulation Project, DEN/DANS/DM2S/STMF/LMES/RT/12-029/A.

[17] Emonot, P., A. Souyri, J.L. Gandrille, and F. Barre. 2011. "CATHARE-3: a new system code for thermal-hydraulics in the context of the NEPTUNE project." Nuclear Engineering and Design 241 (11): 4476-4481.

[18] Guelfi, A., D. Bestion, M. Boucker, P. Boudier, P. Fillion, M. Grandotto, J. Hérard, E. Hervieu, and P. Péturaud. 2007. "NEPTUNE: A New Software Platform for Advanced Nuclear Thermal Hydraulics." Nuclear Science and Engineering 156 (3): 281-324.

[19] H¨ohne, T., E. Krepper, and U.Rohde. 2009. "Application of CFD Codes in Nuclear Reactor Safety Analysis." Science and Technology of Nuclear Installations 2010 (3).

[20] Ilvonen, M., V. Hovi, and P. Inkinen. 2010. PORFLO development, applications and plans in 2008-2009. Finnland: VTT.

[21] Jeong, J.J., H.Y. Yoon, I.K. Park, and H.K. Cho. 2010. "The CUPID Code Development and Assessment Strategy." Nuclear Engineering and Technology 42 (6): 636–655.

[22] Jeong, J.-J., S. K. Sim, C. H. Ban, and C. E. Park. 1997. "Assessment of the COBRA/ RELAP5 Code using the LOFT L2-3 Large-Break Loss-of-Coolant Experiment." Annual of Nuclear Energy 24 (14): 1171-1182.

[23] Kolev, N., S. Aniel, E. Royer, U. Bieder, D. Popov, and T. Topalov. 2010. "VVER-1000 Coolant Transient Benchmark Phase 2 (V1000CT-2): Final Specifications of the VVER-1000 Vessel Mixing Problem. Report." Paris: OECD Nuclear Energy Agency, NEA/NSC/ DOC(2010)10, ISBN: 978-92-64-99152-1.

[24] Kucukboyaci, V., and Y. Sung. 2015. "COBRA-TF Parallelization and Application to PWR Reactor Core Subchannel DNB Analysis." ANS MC2015 – Joint International Conference on Mathematics and Computation(M&C), Supercomputing Nuclear Applications (SNA) and the Monte Carlo (MC) Method. Nashville.

[25] Lerchl, G., and H. Austregesilo. 2006. ATHLET Mod 2.1 Cycle A, User's Manual. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH.

[26] Coupling System of ATHLET, DYN3D and FLUENT Codes." NURETH-15 The 15th International Topical Meeting on Nuclear Reactor Thermal - Hydraulics. Pisa, Italy.

[27] Macek, J., and L. Vyskoci. 2013. "Simulation of SDA Opening Test at VVER-1000 NPP by Coupling System of ATHLET, DYN3D and FLUENT Codes." NURETH-15 The 15th International Topical Meeting on Nuclear Reactor Thermal - Hydraulics. Pisa, Italy.

[28] Mahaffy, J., B. Chung, F. Dubois, F. Ducros, E. Graffard, M. Heitsch, M. Henriksson, et al. 2007 Best Practice Guidelines for the use of CFD in Nuclear Reactor Safety Applications. Nuclear Energy Agency, Committee on Safety of Nuclear Installations.

[29] Moorthi, A., A. K. Sharma, and K. Velusamy. 2018. "A review of sub-channel thermal hydraulic codes for nuclear reactor core and future directions." Nuclear Engineering and Design 332: 329–344.

[30] Niceno, B., Y. Sato, A. Badillo, and M. Andreani. 2010. "Multi-scale Modeling and Analysis of Convective Boiling Towards the Prediction of CHF in Rod Bundles." Nuclear Engineering and Technology 42 (6): 620-635.

[31] NURESAFE. 2019. Nuclear reactor safety simulation platform. Accessed 04 16, 2019. http://www.nuresafe.eu/index.php?art=24.

[32] Petruzzi, A., and F. D'Auria. 2007. "Thermal-Hydraulic System Codes in Nuclear Reactor Safety and Qualification Procedures." Science and Technology of Nuclear Installations 2008 (3).1999. RELAP5-3D code manuals, Volumes I, II, IV, and V. Idaho National Engineering and Environmental Laboratory.

[33] Runchal, A.K., and B. Sagar. n.d. PORFLOW: A multifluid multiphase model for simulating flow,heat transfer, and mass transport in fractured porous media. Washington, DC, U.S.: Nuclear Regulatory Commission.

[34] NSALOME. 2019. SALOME - YACS module. Accessed 04 16, 2019. https://www.salome-platform.org/user-section/about/yacs. —. 2019. SALOME-the open source integration platform for numerical simulation. CEA, EDF, Open Cascade. Accessed 08 07, 2019. https://www.salome-platform.org/.

[35] Sánchez-Espinoza, V.H., U. Imke, and A. Ivanov. 2010. "SUBCHANFLOW: A Thermal-Hydraulic Sub-Channel Program to Analyse Fuel Rod Bundles and Reactor Cores." 17th Pacific Basin Nuclear Conference. Cancún, Q.R., México.

[36] Sung, Y.X., P. Schueren, and A. Meliksetian. 1999. VIPRE-01 Modeling and Qualification for Pressurized Water Reactor Non-LOCA Thermal-Hydraulic Safety Analysis. Westinghouse Electric Company.

[37] Thurgood, M.J., J.M. Kelly, T.E. Guidotti, R.J.Kohrt, and K.R. Crowell. 1983. COBRA/TRAC—AThermal-Hydraulics Code for Transient Analysis of Nuclear Reactor Vessels and Primary Coolant Systems. Washington, DC, U.S.: Pacific Northwest Laboratory.

[38] Toumi, I., A. Bergeron, D. Gallo, E. Royer, and D. Caruge. 2000. "FLICA-4: A Three-DimensionalTwo-Phase Flow Computer Code with Advanced Numerical Methods for Nuclear Applications." Nuclear Engineering and Design 200 (1-2): 139–155.

[39] n.d. TRACE TRACE V5.1051 theory manual. Washington, DC, U.S.: U. S. Nuclear Regulatory Commission.

[40] U.S. NRC. 2010. TRACE V5.0 Theory Manual - Field Equations, Solution Methods, and Physical Models. U.S. Nuclear Regulatory Commission.

[41] 2003. Use and Development of Coupled Computer Codes for the Analysis of Accidents at Nuclear Power Plants. Vienna, Austria: Proceedings of a Technical Meeting Held in Vienna.

[42] Yadigaroglu, G. 2005. "Computational Fluid Dynamics for nuclear applications: from CFD to multi-scale CMFD." Nuclear Engineering and Design 235 (2-4): 153–164.

[43] Zhang, H., L. Zou, H. Zhao, and R. Martineau. 2014. Developing Fully Coupled Subchannel Model in RELAP-7. Idaho National Laboratory Originating Organization.

[44] Zhang, Kanglong, Xilin Zhang, Victor Sanchez-Espinoza, and Robert Stieglitz. 2020" Development of the coupled code – TRACE/TrioCFD based on ICoCo for simulation of nuclear power systems and its validation against the VVER-1000 coolant-mixing benchmark." Nuclear Engineering and Design, 06.

[45] Zhang, Kanglong; Sanchez-Espinoza, Victor. 2019. "The Dynamic-Implicit-Additional-Source(DIAS) method for multi-scale coupling of thermal-hydraulic codes to enhance the prediction of mass and heat transfer in the nuclear reactor pressure VESSEL." Internatinal Journal of Heat and Mass Transfer 50-62.

APPENDIX A COMPARISON OF THE CFD CODES ACCORDING TO THE MULTI-SCALE DEVELOPMENT UNDER THE SCOPE OF THIS REPORT

Normally, there are two general categories of CFD codes: one is commercial tools and the other is open-source tools. With decades of consistent development, lots of CFD codes are available nowadays. The link https://www.cfd-online.com/Wiki/Codes works out a comprehensive overview of both free and commercial CFD software. Some information on pre and post-processing tools is also included.

Among the various CFD software, we select the three most well-known and widely-used codes as the representatives of commercial CFD. They are ANSYS-CFX, ANSYS-FLUENT, and STAR-CCM+. OpenFOAM and TrioCFD are selected to represent the free CFD because the former one has overwhelming superiority over other free CFD codes on popularity and TrioCFD has a built-in ICoCo interface which is the selected interface for this multi-scale development. The typical CFD software consists of three essential elements: pre-processor, solver and post-processor. The pre-processor will do the pre-processing which normally includes CAD and grid generation. The post-processor is in charge of the post-processing which mainly refers to the result visualization. The solver is the kernel which will perform the actual calculation and generate the result.

Generally speaking, commercial codes have a more powerful pre-processor and post-processor than free codes. Nevertheless, the free pre- and post-processing tools like SALOME and ParaView et al are experiencing rapid expansion and are closing the gap nowadays. Then again, the pre- and post-processing are not the key issue for the CFD codes selection. Since the final goal is the coupling of TRACE and CFD codes, which concerns much about the codes' numeric or solution scheme, thus the key point is to select the most appropriate solver surpassing the integrated CFD codes.

For that purpose, four major evaluation criteria are put forward. For each of them, several sub-criteria may be applied. Each of the criteria or sub-criteria will be evaluated by three scores: 0 – Poor, 1 – Good, 2 – Very Good. The total scores will be the final evaluation reference.

1) C1 – Capability. There are two sub-criteria – C1.1 and C1.2.
    a. C1.1 – The sub-criteria focus on physics. Could the solver handle the common thermal-hydraulic problems and produce reasonable results? Could the solver deal with some peculiar and unusual problems while at the same time produce reasonable results (Since most of the solver are more or less good at some specific areas, it is unpractical to have one single solver handling all kinds of cases.)?
    b. C1.2 – This sub-criteria relates more to the computer. It is to validate the solver's ability to utilizing computation resources. Normally, there is not a best solver which has a significant advantage over the others on CPU time and memory occupation during the runtime for the same case. Because the codes of them are all well-optimized. The possible difference could be their ability to shorten the elapsed time, which is the real running time. Practically speaking, it depends whether the solver could run in parallel or not.

ANSYS-CFX. This solver covers the majority of thermal-hydraulic problems and is proved to be robust enough. Thus 2 points got here (C1.1). It can run in parallel to save computing time. So another 2 points got (C1.2). The total score is 4.

ANSYS-FLUENT. It is the most classical solver around the world. It has quite abundant build-in physical models. 2 point got here (C1.1). It can run in parallel. So, another 2 points got (C1.2). The total score is 4.

STAR-CCM+. It is another classical solver and applied the most advanced computational continuum mechanics algorithms. Plenty of build-in physical models also enable the solver to deal with various kinds of thermal-hydraulic problems. So, 2 points got here (C1.1). Also, the parallel capability was developed. Thus, another 2 points got (C1.2). The total score is 4.

OpenFOAM. As the most popular open-source CFD software, various models were and are developed for various purposes within OpenFOAM. The physical diversity and richness in this solver are incomparable. However, those models are not that well-developed and sometimes depend heavily on the expertise of the users. So 1 point got here (C 1.1). Parallelization is also possible in OpenFOAM by following the MPI standards. However, this MPI capability is not that user-friendly as the commercial software. Thus, another point got here. The total score is 2.

TrioCFD. It supports various physical models and can manage a wide range of flow conditions. Besides, it can sufficiently utilize the MPI capability for parallel computation. However, when conditions fall into reality, TrioCFD faces the same problem with OpenFOAM, which mainly due to the imperfection of the models and the complexity of the usage. When the parallel calculation is desired, the transformation from serial and parallel is tricky. So each of the sub-criteria picks up one point. The total score is 2.

2) C2 – Availability. There are two sub-criteria – C2.1 and C2.2.
    a. C2.1 – The criteria refers to whether the code is open-source or closed-source. Since the open-source codes enable developers to manipulate the codes' key logics visibly, new numerical algorithms or new modules could be flexibly developed and merged into the codes. While for closed-source codes, it is usually impossible to explore their kernels. Thus, the secondary development of such codes could be quite limited.
    b. C2.2 – The other aspect of availability could be the completeness or richness of the codes related instruction and tutorial documentation.

ANSYS-CFX. The software is closed-source and expensive. Thus, no point got here (C2.1). While the user manual is well organized and there are also sufficient learning materials and shared experience on various forums. So, 2 points got (C2.2). The total score is 2.

ANSYS-FLUENT. For the same reason with ANSYS-CFX (the learning materials and tutorials are dramatically rich for both online and offline). No point got for C2.1 and 2 points got for C2.2. The total score is 2.

STAR-CCM+. For the same reason with ANSYS-CFX but the educational stuff is not that accessible and rich than CFX and FLUENT. No point got for C2.1 and 1 point got for C2.2. The total score is 1.

OpenFOAM. The software is open-source and free. Thus 2 points got here (C2.1). The learning materials of OpenFOAM is not as sufficient as the above three commercial codes. However, useful and practical information and experience are indeed available online. So 1 point got here (C2.2). The total score is 2.

TrioCFD. It is completely open-source and free. So it gets 2 scores for C2.1. The learning material is sharply limited to a small scale of training stuff developed by the developing team and the community is rather small and the technical support normally comes only from the developing team directly. So, C2.2 scores 0.

3) C3 – Extensibility. There are three sub-criteria – C3.1, C3.2, and C3.3.
    a. C3.1 – The flexibility of the manipulation of existing modules' data. For instance, define the user-specified boundary condition, revise the time step size and manipulate the physical parameters during the runtime.
    b. C3.2 – The possibility to modify the solution schemes of existing modules. Compared with C3.1, this sub-criteria aims to explore the modules in-deep working flows.
    c. C3.3 – The possibility to develop new functions. For instance, define new physical models, and define new solution schemes.

ANSYS-CFX. Two approaches are available for the CFX's existing modules' data manipulation. One is the user CEL (CFX Expression Language) functions and routines, the other one is the user junction box routines. 2 points got here (C3.1). While CFX is closed-source, neither exploring its workflow nor developing new functions are possible. So no more point got for C3.2 and C3.3. The total score is 2.

ANSYS-FLUENT. The classical UDF (User Defined Functions) is used to manipulate the FLUENT existing modules' data. 2 point got here (C3.1). While FLUENT is closed-source, neither exploring its workflow nor developing new functions are possible. So no more point got for C3.2 and C3.3. The total score is 2.

STAR-CCM+. A JAVA interface was published with the solver and aims to assist the data manipulation. 2 points got for C3.1. While STAR-CCM+ is closed-source, neither exploring its workflow nor developing new functions are possible. So no more point got for C3.2 and C3.3. The total score is 2.

OpenFOAM. Since the software is open-source, it is quite convenient to explore the code's data structure, working flow, and low-level programming work. Users or developers have great flexibility to manipulate the code's data, modify the working flow and even develop new functions. So C3.1, C3.2, and C3.3 could get the highest score. The total score is 6.

TrioCFD. The condition is the same with OpenFOAM. So, 6 scores are made here.

4) C4 – Adaptability to ICoCo. There are two sub-criteria – C4.1 and C4.2.
    a. C4.1 – does the code already has an ICoCo interface or not.
    b. C4.2 – The possibility to develop an ICoCo interface for the code as well as the complexity of the developing process.

For ANSYS, FLUENT, and STAR-CCM+, no one of them has ICoCo and it is impossible to develop ICoCo for them because they are all close-source. So, the two sub-criteria for all three commercial codes are zero. As to OpenFOAM, since it does not has the ICoCo interface, no score is made for C4.1. But it is possible to develop a new ICoCo module for it thanks to its open-source though the heavy workload is necessary. So C4.2 makes 1 point. As to TrioCFD, the ICoCo interface is already inside. And the modification and extension to ICoCo are convenient thanks to its open-source. So both of the two sub-criteria get 2 scores, totally 4.

The score details are summarized in Table 1. It presents an obvious advantage of TrioCFD over the other four representative CFD codes. So, TrioCFD was selected to be the right CFD codes for TRACE and CFD coupling. The commercial codes are normally industry-oriented while the free codes are research-oriented. What we are going to implement for TRACE and CFD coupling is exactly for research purposes. Once again, it is essential to note that the comparison was only for the CFD solvers. The pre-processor and post-processor are not included in the evaluation criteria. The last point calling for attention is that OpenFOAM does not lose too much compared with TrioCFD. The huge user community and rich learning materials are quite attractive. So, OpenFOAM is a most promising code for further multi-scale coupling development in the future.

Table A-1 Total Score of CFD Software Comparison

CFD Software		ANSYS-CFX	ANSYS-FLUENT	STAR-CCM+	OpenFOAM	TrioCFD
C1	C1.1	2	2	2	1	1
	C1.2	2	2	2	1	1
C2	C2.1	0	0	0	2	2
	C2.2	2	2	1	1	0
C3	C3.1	2	2	2	2	2
	C3.2	0	0	0	2	2
	C3.3	0	0	0	2	2
C4	C4.1	0	0	0	0	2
	C4.2	0	0	0	1	2
Total		8	8	7	12	14

After selecting TrioCFD as the to-be-coupled CFD codes, one vital problem arises which is how to handle the CAD model and mesh for TrioCFD. The situation is that we already have the model's CFX mesh, but how to transfer it to TrioCFD? Normally, several such tools can properly handle the mesh transform from CFX to TrioCFD, like the SALOME platform. Well, SALOME is indeed bound to the coupling system tightly.

APPENDIX B COMPARISON OF ICOCO AND ECI ACCORDING TO THE MULTI-SCALE DEVELOPMENT UNDER THE SCOPE OF THIS REPORT

Mechanisms of the two coupling interfaces are present in the following figures.

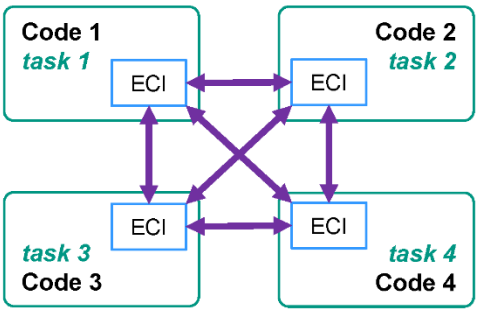


Figure B-1 Working Mechanism of ECI

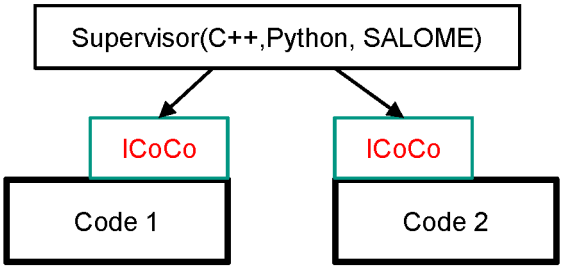


Figure B-2 Working Mechanism of ICoCo

There are three major evaluation criteria. For each of them, several sub-criteria may be applied. Each of the criteria or sub-criteria will be evaluated by two scores: 0 – Poor, 1 – Good. The total scores will be the final evaluation reference.

1) C1 – Capability. There are three sub-criteria – C1.1, C1.2, and C1.3.
   a) C1.1 – The score of the criteria mostly comes from the variety of the coupling implementations. For instance, the running mode of the coupling codes – can they run in serial or parallel?
   b) C1.2 – The numerical depth of the coupling – can the interfaces exchange data between codes in an input-output level or in a time-step level or deep into the solution scheme level?
   c) C1.3 – The last essential aspect could especially focus on the data communication capability of the interfaces.
   d)
      - ECI. It is originally designed for parallel running due to the applied socket technology, which makes it possible to run the codes on distributed computer systems. Thus 1 point is got (C1.1). As to the numerical depth, ECI could go deep into the solution scheme level. However, in most cases, the coupling levels are determined by the similarity of the codes' solution schemes. Anyway, ECI could get another point here (C1.2). Thanks to the genius design of ECI, more than 2 codes could be included in the coupling system. Moreover, each two ECI or codes could directly communicate with each other, which indicates a serverless system. Thus another point got here (C1.3). The total score is 3.
      - ICoCo. Coupling codes implemented by ICoCo could run in parallel mode and on distributed systems. While the serial mode is also available. Thus 1 point got here (C1.1). Additionally, ICoCo could also make the coupling within solution schemes, just like ECI. So another point got here (C1.2). Normally, ICoCo is applied for two code coupling systems. However, it has the potential for a multi-codes included system. The possible drawback is that the data management process could be a little bit complicated since an isolated data transfer driver is in charge of all the

incoming and outgoing messages from or to the included codes, indicating a typical server-client system. So, no more point is got here (C1.3). The total score is 2.

2) C2 – Availability. There are three sub-criteria – C2.1, C2.2, and C2.3.
   a) C2.1 – The criteria refers commonly to that whether the code is open-source or closed-source. Since the open-source codes enable developers to manipulate the codes' key logics visibly, new numerical algorithms or new modules could be flexibly developed and merged into the codes. While for closed-source codes, it is usually impossible to explore their kernels. Thus the secondary development of such codes could be quite limited.
   b) C2.2 – The other aspect of availability could be the completeness or richness of the codes related instruction and tutorial documentation.
   c) C2.3 – The sub-criteria could be understood as the difficulty of secondary development. Easier the interface's logic is, easier the development will be. Moreover, it also depends on the relationship between the interface and the codes. If the interface is originally designed for the code, the development work could be relatively simple.
      - ECI. It is open-source and developed with Fortran and C. The instruction text is also well written. So, ECI could make 1 point (C2.1). However, only the coupling between TRACE and SUBCHANFLOW was successfully implemented so far. No more samples are available. So ECI could make no more points (C2.2). The ECI logic is a little bit complicated since it directly handles the bottom layer of network communication. However, since it is inherently designed for TRACE, its data structure highly agrees with that of TRACE. Thus one more point is got here (C 2.3). The total score is 2.
      - ICoCo. It is also an open-source interface and developed with pure C++. There are a lot of texts illustrating its functionality. Thus ICoCo gets 1 point here (C2.1). Better than ECI, ICoCo was implemented to relatively abundant coupling cases which could be a nice reference for the coupling of TRACE and CFD codes. Hence ICoCo makes 1 more point (C2.2). ICoCo's logic is simpler than ECI, however, it is a third party interface that has no apparent relationship with any other known codes. Thus the development work could be heavier and no more point is got here (C2.3). The total score is 2.
3) C3 – Extensibility. There are two sub-criteria – C3.1 and C3.2.
   a) C3.1 – The sub-criteria is related to the standardization and modularity. The higher degree of standardization and modularity implemented in the interface, the better extensibility and portability will the interface have and the easier it will be to develop the coupling.
   b) C3.2 – Integration of well-developed platforms. The sub-criteria is similar to the standardization and modularity of C3.1. But its directivity is more specific. The platforms themselves could be treated as general tools to combine various modules or as general interfaces to combine various functions. The interface's integration to platforms normally means that the coupling would take benefit from the platform and the coupling implementation could be much easier.
      - ECI. The most powerful function which makes ECI differ from other interfaces is the serverless network communication ability which was achieved by manipulating the sockets in a quite flexible way. Despite the outstanding performance, it is not a standardized programming way (like the MPI standard). Additionally, the modularity of ECI is poor since its subroutines have complicated relationships with that of the coupled code. It is quite difficult to draw a clear called and calling map for ECI and the coupled code. So, no point got here (C3.1). Moreover, ECI is not designed for any platform, which means developers could only make the coupling with the single interface without the help of any platforms. Normally, this would lead to various coupling pairs with various coupling data structures, which is not a standardized way. So, no more points got here (C3.2). The total score is 0.

- ICoCo. The subroutines of ICoCo contain less coding work and have clear logic. Thus it can be used in a more modularized way. Especially, ICoCo follows MPI standards for the parallel running mode. Developers don't have to be an expert in network communication to struggle with its bottom layer. Following the MPI instructions will make the coupling more modularized, more standardized and more robust. Thus 1 point got here (C3.1). The other advantage of ICoCo over ECI is that it is inherently designed for code coupling within the SALOME platform. Every code must be developed to an ICoCo module first as the preparation for the coupling in SALOME. Theoretically, all of the ICoCo modules are standardized SALOME modules, which means that not only the ICoCo modules which are developed from the currently used codes are possible, their coupling with already existing ICoCo modules which were developed from other codes are also possible. Because of the excellent coupling flexibility of ICoCo in SALOME, another point is got here (C3.2). The total score is 2.

The score details are summarized in Appendix Table 2. It could be inferred that there is no distinct advantage for one interface over the other one.

Table B-1 Total Score of Interface Comparison

Codes Coupling Interface		ECI	ICoCo
C1	C1.1	1	1
	C1.2	1	1
	C1.3	1	0
C2	C2.1	1	1
	C2.2	0	1
	C2.3	1	0
C3	C3.1	0	1
	C3.2	0	1
Total		5	6

ICoCo can also be used outside of SALOME. One more driver script is desired to manage the data transfer, instead of SALOME. The driver could be developed with either C++ or Python. Another mechanism that deals with the data mapping process between different meshes of different codes should also be developed. This mechanism is not a must in SALOME coupling since the SALOME platform will handle this issue. Appendix Figure 3 presents a clearer picture of the ICoCo coupling options.

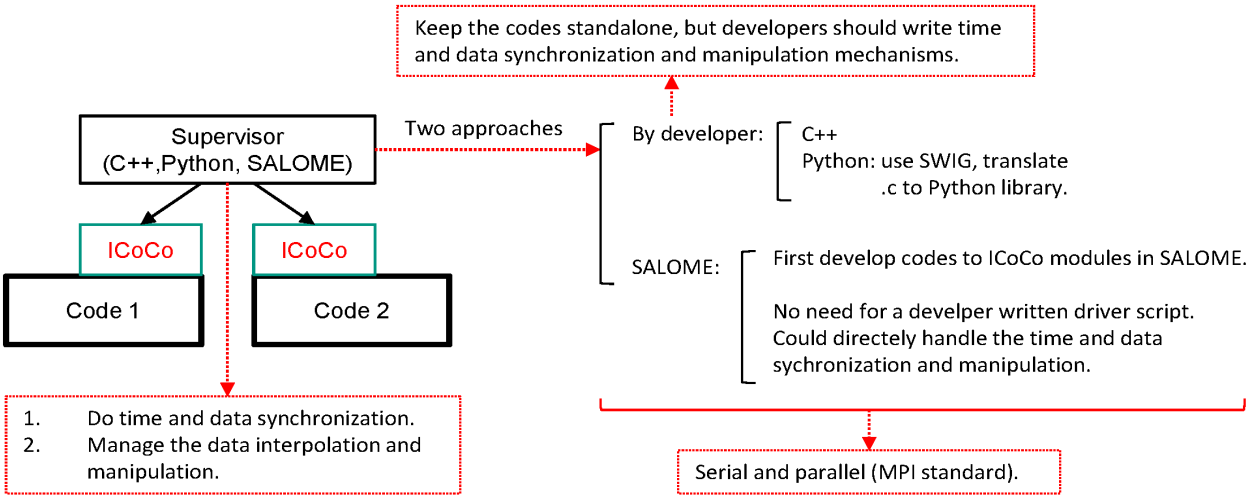


Figure B-3 ICoCo Coupling Options

Though two options are available for ICoCo coupling, the coupling within SALOME is highlighted due to the high degree extensibility. This is also the key criterion that we emphasize for the selection of the best coupling interface for TRACE and CFD coupling – ECI or ICoCo. Though ECI may perform better than ICoCo in specific cases, it has to face the defeat when the situation comes to extensibility which is exactly the aspect ICoCo good at. So ICoCo is selected for TRACE and CFD coupling.

APPENDIX C MODIFICATION AND NEW ROUTINES TO TRACE FOR ICOCO DEVELOPMENT

To implement the TRACE/TrioCFD-ICoCo system, dozens of modifications and fresh subroutines have to be accomplished to the code's source, mainly for TRACE. Because the operation to TrioCFD source is quite complicated and the work focuses more on bringing the in-depth ICoCo module to the logical surface of TrioCFD (because of the large size and complexity of TrioCFD source and structure). They are summarized in the following table. The bold texts are the real files to be modified or added and the normal texts are the subroutines within the real files.

Important – Use the MELD tool or other tools for codes comparison, there the newly-developed and modified subroutines compared to the source codes that could be easily found. Please note that the TRACE used in this report includes the fixed ECI. So you may find the changes to the file – ExTransferM.f90.

Table C-1 Modifications and New Subroutines to TRACE in TRACE/TrioCFD-ICoCo

Modification to TRACE	New Fortran files to TRACE	New C++ files to TRACE
src/ExTransferM.f90	trace_input.f90	TRACEICOCO.hxx
cpVarGet	trace_init.f90	TRACEICOCO.cxx
cpVarPut	trace_terminate.f90	setDataFile_TRACE
AcceptTransmits	trans_init.f90	initialize_TRACE
ExtTranserialI	trans_gettimestep.f90	terminate_TRACE
ExtTranGet	trans_runstep.f90	presenttime_TRACE
ExtTranSend	trans_terminate.f90	computeTimeStep_TRACE
GetCPvarPtr	ICOCOdataM.f90	initTimeStep_TRACE
ReadRequests	ICOCOfuncM.f90	solveTimeStep_TRACE
SchedTransmits	rename_files_icoco	getInputMEDFieldTemplate_TRACE
SetTasks	trans_puttimestep	setInputMEDField_TRACE
Statuscheck	get_trace_array	getInputFieldsNames_TRACE
src/FluidVolumesM.f90	allocate_interstore_array	getOutputMEDField_TRACE
FinalBackSub	trace_set_face	getOutputFieldsNames_TRACE
LoadDensities	trace_set_cell	isStationary_TRACE
src/VessInputM.f90		
RVssl		
src/post.f90		
src/VectDragM.f90		
prefwd		
src/CmdLineArgsM.f90		
ProcessArgs		
src/TprVesselM.f90		
ReTprVessel		
src/SoluteM.f90		
PlatedSolute		
src/TimeStepInputM.f		
ReadTimeStepData		
src/VessTF3DSM.f90		
VdV3D		
trac.f90		
common/IoM.f90		

NRC FORM 335
(12-2010)
NRCMD 3.7

U.S. NUCLEAR REGULATORY COMMISSION

BIBLIOGRAPHIC DATA SHEET

(See instructions on the reverse)

1. REPORT NUMBER
(Assigned by NRC, Add Vol., Supp., Rev., and Addendum Numbers, if any.)

NUREG/IA-0542

2. TITLE AND SUBTITLE

Multi-Scale Coupling of TRACE and TrioCFD with the Interface for Code Coupling (ICoCo)

3. DATE REPORT PUBLISHED

MONTH	YEAR
June	**2025**

4. FIN OR GRANT NUMBER

5. AUTHOR(S)

Kanglong Zhang; Victor Hugo Sanchez-Espinoza

6. TYPE OF REPORT

Technical

7. PERIOD COVERED (Inclusive Dates)

8. PERFORMING ORGANIZATION - NAME AND ADDRESS (If NRC, provide Division, Office or Region, U. S. Nuclear Regulatory Commission, and mailing address; if contractor, provide name and mailing address.)

Institute for Neutron Physics and Reactor Technology (INR), Karlsruhe Institute of Technology (KIT)
Hermann-von-Helmholtz-Platz 1
Eggenstein-Leopoldshafen, Baden-Württemberg, 76344, Germany

9. SPONSORING ORGANIZATION - NAME AND ADDRESS (If NRC, type "Same as above", if contractor, provide NRC Division, Office or Region, U. S. Nuclear Regulatory Commission, and mailing address.)

Division of Systems Analysis
Office of the Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555-0001

10. SUPPLEMENTARY NOTES

A. Hsieh, Program Manager

11. ABSTRACT (200 words or less)

This report describes the multi-scale coupling of the system code - TRACE and the open-source CFD code – TrioCFD for a better description of the multi-dimensional thermal-hydraulic phenomena inside the Reactor Pressure Vessel (RPV) of a Pressurised Water Reactor (PWR).

The Interface for Code Coupling (ICoCo) is a generic interface for code coupling and it defines a standardized framework for the code functions. ICoCo is already a built-in module embedded in TrioCFD. To couple TRACE with the open-source CFD code - TrioCFD, a specific ICoCo-module for TRACE was developed. In this report, the implemented spatial mapping of the involved thermal-hydraulic domains and the time synchronization of the involved solvers are described. A domain overlapping approach and the open-source MEDCoupling library are utilized for this purpose.

Besides, an explicit operator splitting method is implemented for the data transfers during the time advancement of both codes.

The prediction capability of the coupled code is demonstrated by the analysis of a 3D-coolant-mixing problem performed in a VVER-1000 reactor. The results obtained by TRACE standalone and by the coupled system TRACE/TrioCFD were compared together and it shows that the coupled code could predict better coolant mixing along the core height than TRACE-standalone.

12. KEY WORDS/DESCRIPTORS (List words or phrases that will assist researchers in locating the report.)

Multi-scale Coupling, TRACE, TrioCFD;.ICoCo

13. AVAILABILITY STATEMENT

unlimited

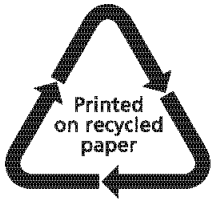
14. SECURITY CLASSIFICATION

(This Page)
unclassified

(This Report)
unclassified

15. NUMBER OF PAGES

16. PRICE



Federal Recycling Program

UNITED STATES
NUCLEAR REGULATORY COMMISSION
WASHINGTON, DC 20555-0001

OFFICIAL BUSINESS