TOOLS AND TECHNIQUES FOR MULTIDISCIPLINARY COUPLED SIMULATIONS

A. Schreiber¹, H.-P. Kersken¹, Klaus Wolf², Peter Post², Reinhold Niesner³, Matthias Haupt³
¹ German Aerospace Center, Simulation and Software Technology, 51147 Cologne, Germany
² Fraunhofer Institute for Algorithms and Scientific Computing, 53754 Sankt Augustin, Germany
³ TU Braunschweig, Institute of Aircraft Design and Lightweight Structures, 38108 Braunschweig, Germany

ABSTRACT

This paper describes three kinds of tools for supporting multidisciplinary coupled simulations: A coupling library for data exchange and mesh interpolation, a script interpreter for controlling the coupled computation, and an integration system for graphical setup and start of coupled simulations in distributed environments.

A tool for enabling stand-alone applications to be used in a multidisciplinary coupled simulation is the MpCCI library. It is integrated into various simulation codes and handles the data exchange and interpolation between the meshes of the codes.

An add-on to the library has been developed to use a Python script for controlling the coupled simulation, i.e. for implementing coupling algorithms.

As a mean to simplify the error prone task of setting-up and executing multidisciplinary, coupled simulations the TENT system is presented.

1. INTRODUCTION

This paper describes software tools and libraries which can be employed to help the users in performing coupled simulation. It focuses on the usage of such tools in the simulation environment TENT.

TENT is a component-based software integration environment and workflow management system. It is used to integrate the applications for building complex workflows which are typical for multidisciplinary simulations in engineering and scientific applications where simulation codes have to be coupled to obtain reliable results. TENT provides means to simplify the error prone task of setting-up and executing multidisciplinary, coupled simulations.

Software development techniques based on modern software products like powerful scripting languages, e.g. Python, portable implementation of basic system parts by using Java, and encapsulation of data and task using an component-based approach relying on CORBA for the interaction of the components are employed for the implementation.

TENT has been developed to aid users in setting up and steering scientific workflows in distributed, non-homogeneous hardware environments. Typically, a workflow consists of simulation codes, pre- and post-processing tools, and visualisation programmes. In addition, a data server is available to store and reload simulation and configuration data. The applications of the workflow are control via TENT components which communicate with the integration system by standardised CORBA interfaces and with the application code using a socket connection. The data transfer between two applications is managed by the associated components in a peer-to-peer manner.

To support coupled computations that are using the MpCCI library for interpolation and data transfer between codes a special TENT component has been developed. This component handles the start-up of the computation and creates the files necessary to run an MpCCI-job using information supplied by the components of the coupled applications. To integrate proprietary or commercial software into the system, co-processes have been developed to make these programmes accessible to an MpCCI-based coupling.

After introducing two tools which enhance stand-alone applications to be used in coupled simulation a general overview about the integration system with special emphasis of its use for multidisciplinary simulation is given and examples in the fields of aeroplane, turbine, and spacecraft design are presented.

2. COUPLING TECHNOLOGIES

2.1. Overview

Coupling of different simulation codes, each specialized for a specific physical regime, is becoming more and more important for numerical simulations, both in industry and in research. The reason is that in many real-world applications the interaction of different physical phenomena must be taken into consideration in order to achieve high quality predictions.

While there are some vendor specific approaches to coupling codes from different applications fields via proprietary mechanism as in MSC/Nastran for example, there is also some non code specific coupling software available. But its use is restricted to special application fields like aeroelastic coupling in airplane design, see e.g. [7] or the coupling of ocean and atmosphere models in climate research, see e.g. [2].

This situation motivated the development of a software environment which provides a general solution for multidisciplinary simulations.

2.2. MpCCI Coupling Library

2.2.1. Introduction

MpCCI (Mesh based parallel Code Coupling Interface) has been developed at the Fraunhofer Institute SCAI in order to provide an application independent interface for coupling different simulation codes, usable both for commercial codes and in-house codes.
2.2.2. MpCCI Concepts

MpCCI is a software library which enables the exchange of data between the meshes of two or more simulation codes in the coupling region (see Figure 1). Since the meshes need not match point by point, MpCCI performs an interpolation and, in case of parallel codes, keeps track of the distribution of the domains onto different processes. In this way, the intricate details of the data exchange are hidden behind the concise interface of MpCCI. As a consequence, the simulation codes themselves are changed only moderately when they are prepared for coupling via MpCCI.

2.2.3. MpCCI 2.0

MpCCI is a registered trademark of Fraunhofer SCAI. MpCCI Version 2.0 will be released in January 2003. Distribution will be organized through Pallas GmbH, further details and software download are available on [5]. Leading software vendors like CD adapco, Intes, AEA, MSC/Marc, ANSYS, HKS and FLUENT will support the MpCCI interface in their new product releases. MpCCI is running on all major UNIX systems, on LINUX, and on Windows. MpCCI 2.0 will be put for sale in three levels:

- MpCCI 2.0 Standard provides moderate advancements to MpCCI 1.3. The intention is to enable couplings through 2D-surfaces. A graphical user interface and the Playback-Tool are part of the Standard level.
- MpCCI 2.0 Professional provides more accurate 2D interpolations and allows 3D-coupling. Mesh adaptation in the codes will be supported through improved neighbourhood algorithms. The MpCCI Visualiser offers a detailed graphical insight into any coupling action and data passed through the coupling regions.

2.2.4. New Features in MpCCI 2.0

There are a lot of new features in MpCCI 2.0 included:

- Volume coupling in the two- and three dimensional space
- Mappings based on intersection methods for every supported coupling region
- Coupling on the sphere including new element types
- Module for coordinate transformation
- Definition of interpolation methods for pairs of meshes via the MpCCI input file

2.2.5. Synchronization Points

The concept of synchronization points is especially designed for the case of commercial codes where the end user can configure the coupled computation only by an input file, not by recompiling. A synchronization point is a well defined point in the algorithm of the code where MpCCI communication can happen. If a code sets a synchronization point at each important stage of its algorithm (before the solver, end of the coupling step, etc.) the end user can define in the MpCCI 2.0 input file what quantities should be exchanged at what point of the algorithm.

The concept of synchronization points is a step forward in the direction of a general coupling capability of a simulation code (independent of a specific code on the other side). A synchronization point of a code can be mapped onto several synchronization points of the remote code. As shown in Figure 2, this is needed for setting up a Gauss-Seidel like coupling algorithm.

2.2.6. Interpolation

MpCCI 2.0 supports different kinds of coupling regions and spaces. In the two- and three-dimensional space surface and volume coupling based on element definitions are possible. Meshes on the sphere consisting of spherical triangles or quadrilaterals are treated as special cases. Furthermore MpCCI 2.0 is able to perform data transfer on nodes without element definitions.

The available mapping schemes are based on the calculation of local coordinates of points (standard interpolation - linear, quadratic), intersection methods or the calculation of the nearest neighbour of a node. MpCCI supports non-conservative and conservative (i.e. sum-preserving ) mappings. The new conservative mapping scheme based on intersection computations may improve the accuracy of the transferred data compared with the former conservative schemes.

2.3. Controlling the Coupled Computation Using Python

To meet the desire for a flexible, easy-to-use environment for coupled computations, a software architecture based on a controlling module has been developed. By introducing a "control code" following characteristics can be achieved:

- flexible simulation steering (e.g. starting the fluid/structure solver, checking for convergence),

![Figure 1: Meshes in the coupling region need not match and can have different element types.](image1)

![Figure 2: Synchronization point concept.](image2)
different techniques, such as the Dirichlet-Neumann Iteration or the Gradient Method, can be easily implemented,
• modularity: smooth integration of further modules (for mesh generation, adaption, post-processing etc) without changes to the simulation codes.

The control code is implemented in Python, thus benefiting from the advantages of this powerful scripting language, such as clear syntax, rapid development or the vast collection of Python modules. Most notably among the Python modules in this context is Numerical Python, which provides very efficient data structures and algorithms.

Figure 3 illustrates the architecture used for the software concept. It is based on an MpCCI environment which, in this case, consists of two simulation codes (MSC/NASTRAN and the DLR-code Tau [3]) and the control code.

For the communication between the control code and the simulation codes an interface specification has been worked out which defines a set of basic instructions for tasks like:
• send/receive interface state quantities to/from the remote simulation code;
• direct control of the analysis codes (start solver, get/set code parameters);
• operations on the interface state arrays (such as dot product, multiply, add).

These instructions are send by the control code via the MPI-Communicators provided by MpCCI to the simulation codes. The simulation codes either have to implement this specification or make use of so-called "co-processes". These are independent codes that are connected to their respective analysis codes either directly (socket connection or similar) or by file transfer. The software architecture presented in Figure 3 uses two such co-processes, one for MSC/NASTRAN (NCP) and one for Tau (TCP).

The control code consists of a modified Python interpreter (MPI-enabled) and a script which is executed at runtime. The following code fragment in Figure 4 shows the implementation of the Dirichlet-Neumann Iteration. The specification mentioned above also includes instructions to get/set the quantity values directly from the control code, making it possible to perform post-processing tasks or implement more complex techniques for the information transfer between grids.

```python
for i in range(nIter):
    # send flux from FluidCode to StructCode:
    FluidCode.SendInterfaceQuantity(FluxFluid, StructCode)

    # receive flux from FluidCode:
    StructCode.ReceiveInterfaceQuantity(FluxStruct, FluidCode)

    # solve structure:
    StructCode.SetBCQuantity(FluxStruct)
    StructCode.Solve(1, ())
    StructCode.GetResultQuantity(TempStruct)

    # send temperature from StructCode to FluidCode:
    StructCode.SendInterfaceQuantity(TempStruct, FluidCode)

    # receive temperature from StructCode:
    FluidCode.ReceiveInterfaceQuantity(TempFluid, StructCode)

    # relaxation:
    TempFluid = (1-f)*TempFluidOld + f*TempFluid

    # copy TempFluid to TempFluidOld
    TempFluidOld = TempFluid

    # solve fluid:
    FluidCode.SetBCQuantity(TempFluid)
    FluidCode.Solve(1, ())
    FluidCode.GetResultQuantity(FluxFluid)
```

Figure 4: Example Python code for the control code. FluxFluid, FluxStruct, TempFluid, TempFluidOld and TempStruct represent the state vectors on the coupling interface.
3. THE INTEGRATION SYSTEM TENT

To improve the building and managing of process chains for complex simulations in distributed environments, the integration and simulation environment TENT has been developed. All tools of typical simulation workflows can be integrated in TENT and controlled from the users desktop using a graphical user interface (GUI). The key features of TENT are as follows:

- Flexible configuration, online steering, and visualization of simulations.
- Utilization of distributed computing resources such as PCs, workstations, cluster, supercomputers, and Computational Grids.
- Integration of a project based data management with support for cooperative working.

Additional features are:

- Visual composition of process chains (workflows).
- Monitoring and steering of running simulations from any computer in the internet.
- Easy integration of existing applications.
- Automatic and efficient data transfer between the stages in the process chain.

3.1. Technology

The TENT system has been developed as a component system using several modern software technologies, for example:

- Java: All essential parts of TENT are developed in Java to be independent from different platforms.
- CORBA: TENT uses CORBA for specifying Interfaces and for communication between the different parts of the system. For example, CORBA is used or controlling the workflow in the distributed environment.
- XML: All file formats in TENT (e.g., for configuration files) are defined using XML. Also the data management stores the project information in XML format on a data server.
- Python: TENT uses Python as an integrated scripting language for controlling workflows and for integration of applications.
- WebDAV: The data management system stores the project information and the data files on a WebDAV-enabled data server.

Figure 5 shows the general architecture of the TENT system.

3.2. Packages of the TENT systems

The TENT system consists of several software packages, each containing related components and tools. Figure 6 shows an overview of the existing TENT packages. The **Base System** covers all basic functionality of the system needed to use it as an integration environment. This includes components for controlling workflows, factories for starting components and applications in the distributed environment, the name server as the central information service, and a graphical user interface (GUI). **Support Components** are additional services for special applications scenarios which are not covered by the basic functionality. Examples of TENT support components are a data server for storing data files, a monitoring and reporting component, and several special control components (e.g., for coupled simulations). The **Component** package contains wrappers for integrated applications. Wrappers are the interface between the actual application and the CORBA side of the TENT system. Currently, there are wrappers for simulations applications such as scientific CFD codes (e.g., FLOWer or TRACE), as well as commercial CFD codes such as MSC/NASTRAN and ANSYS or visualization tools such as Tecplot or AVS. It contains also wrappers for all necessary support applications such as pre- and post processing tools (filters).

The **Software Development Kit (SDK)** is necessary for developing new TENT components. On top of packages are the **Application Systems**, which are a proper set of configurations and integrated tools for doing specific tasks. The most complex single part of TENT is the graphical user interface that allows the user to build workflows graphically and block oriented, to configure properties of components and applications, to start and control simulations, and to monitor running applications. The GUI consists of the following parts (see Figure 7):

- The **component repository** for selecting the needed components.
- The **project browser** for selecting the necessary components.
- The **workflow editor** for managing the user projects and data on the data server.
- The **workspace manager** for connecting components to workflows.
- The **property editor** for setting properties and parameters of components and applications.
- Some additional panels such as a logger or a scripting console.

Figure 6: Packages of the TENT system.
3.3. MpCCI and TENT

To set-up and to conduct coupled simulations result in a non negligible increase in complexity compared to the execution of simple linear workflows. Separate sets of configuration data have to be maintained which may contain implicit or explicit dependencies induced by the coupled nature of the simulation. Additionally, meta data have to be managed which describe the coupling mechanism and the start-up of the coupled simulation. The coupling algorithm itself is coded into the control code, as described in section 2.3, which has to be included into the MpCCI environment and to be represented in TENT by a component. The launching of the application coupled via the MpCCI library differs from the normal start-up process because the applications are not started by their respective component but by the TentCouplingControl component via the MpCCI start-up mechanism. This complication is caused by the fact that MpCCI is a MPI-based library which requires the joint start of all involved programmes. A further difference to “normal” workflows is the data exchanges between the applications connected via the MpCCI mechanism. The data transfer not managed by the components of the application as described in the last section but completely handled by the applications themselves.

To use programmes where the integration of the MpCCI interface is not possible because the source code is not available (like MSC/NASTRAN or ANSYS) or because a direct integration would result in a more involved logical structure of the programme in a coupled simulation co-processes has been developed. These co-processes communicate via the MpCCI library with other MpCCI enabled applications and with the application itself via a file based interface, i.e. they are able to parse input and output data of the application and generate additional input files from the data received via the MpCCI interface. So far, using files to transfer between the co-process and the application coupling data has not been proven to be a bottleneck in the coupled simulation.

4. APPLICATIONS

This section describes the application of TENT and MpCCI to various coupling problems in the field of airplane and space craft design.

4.1. Fluid-Thermal-Coupling

The thermal load of a turbine blade becomes an increasingly important aspect in the design phase of a new turbine. The applied loads are an important factor in determining the lifetime of a blade. To take the heat loads into account already during the design process a coupled simulation of the gas flow around the blade and the heat transfer in the blade itself has to be conducted. To this end, the CFD-solver Trace [6] has been coupled to MSC/NASTRAN used as heat flow solver. Trace supplied the temperature and heat transfer coefficients at the coupling faces as boundary conditions for the computations performed by MSC/NASTRAN which in turn
computes the temperature in the interior of the blade and returns the temperature at the surface for Trace to be used as boundary conditions in the CFD computation. Figure 9 shows the simulated configuration and Figure 10 the results for a convectively cooled turbine stator blade. A screen shot of the graphical user interface for this simulation is displayed in Figure 11. It shows the two application components, a post-processing tool, a visualization tool, and the MpCCI-Control component.

4.2. Fluid-Structure-Coupling
An important application of coupled simulation are aerolestic computations where the pressure of the flow around an airplane or a blade is used as input into a structural mechanics solver to compute the deformation of the structure. Figure 13 shows result for the computation of an elastic wing using FLOWer [3] for the aerodynamic computation and MSC/NASTRAN for the structural part. A detailed discussion of the results can be found in [8].

4.3. Fluid-Structure-Thermal-Coupling
In the project IMENS TENT is the platform to develop a simulation environment for the simulation of, mechanically and thermal, highly loaded parts of a re-entry space vehicle. For the aerodynamic computation the CFD solvers TAU und DAVIS-VOL are used and for the computation of heat transport and deformation in the structure ANSYS and MSC/NASTRAN have been integrated into the environment. Figure 12 shows results for a two dimensional computation of the temperature distribution across a gap at the control surfaces of the space vehicle. For detailed results refer to [8]. Figure 14 shows a screenshot of the respective graphical user interface for this simulation.
As a last example the project SikMa should be mentioned.
which will integrate applications with TENT to compute the
time dependent coupling of aeroelasticity, i.e. the coupling
of aerodynamics and structural mechanics, and flight
mechanics for simulating a flight manoeuvre.

ACKNOWLEDGEMENTS

The authors like to thank Edmund Kügeler, Jens
Neumann, Roger Schäfer, Ralf Heinrich, and Andreas
Mack for providing results of the applications examples.

REFERENCES

TENT, Concurrency And Computation: Practice And
Experience, 14, 2002, pp. 1553-1568
atmosphère. La Météorologie, 10, 1995, pp 50-61
MEGAFLOW Project, Aerosp. Sci. Technol., Vol. 4,
2000, pp. 223-237
[4] A. Beckert, Ein Beitrag zur Strömung-Struktur-
Kopplung für die Berechnung des aeroelastischen
Gleichgewichtszustandes, DLR Fachbericht 97-42,
1997
Counter Rotating Vortices and the Application for
Fan-Shaped Film-Cooling Holes, Proceedings of the
14th ISABE, 1999, ISABE-Paper 99-7144
W. Krueger, J. Neumann, Aeroelastic Computation
Using the AMANDA Simulation Environment
Proceedings of CEAS Conference on
Multidisciplinary Design and Optimization (DGLR-
Bericht 2001-05), 25-26 June 2001, Cologne, pp. 19-
30
Changes due to Fluid-Structure Interaction in
Hypersonic Flow using ANSYS and TAU,
Proceedings 13. DGLR-Fach-Symposium der STAB,
published)