Basic use of CTL middleware in PAK software

Nenad Busarac[1], Vladimir Dunić[1], Miroslav Živković[1], Radovan Slavković[1]
[1]Faculty of mechanical engineering, Kragujevac

Abstract - *This paper presents, simple example of using Component Template Library developed at the Institute of Scientific Computing at Technical University of Braunschweig, Germany as middleware software. Today, most of physical problems have software or solver which is developed for most technologically advanced actual problems. But when we come to multiphysic problems we need to couple some of those solutions, which is very difficult considering the complexity of those solvers. As example of coupling, the CTL is implemented in PAK software at basic level and algorithm of connection between client, module and processes is presented trough simple communication of imported, computed and exported data. The main advantage is that we make only client and use our existing software as modules. This implementation should be introduction to more complex communication between different modules of PAK software in the next phase of multiphysic software development during the new project cycle of Ministry of science.*

1. INTRODUCTION

Implementation of the Component Template Library (CTL) into the PAK software is the main and the most important step in the PAK Multiphysiscs development. Many physical problems can be solved using separate applications, but real complicated problems need to couple some of those solutions. Application of software for the multiphysics problems has become necessary in many branches of industry such as: automotive and aircraft industry, civil engineering, electronics, telecommunications, biomechanics etc. In order to prepare future more complex connection between PAK modules, the CTL is implemented in PAK software at basic level using one module PAKS. Algorithm of connection between imported, computed and exported data should be simple explanation for easy appliance between other PAK modules.

2.CONTENT TEMPLATE LIBRARY (CTL)

Middleware is relatively new addition to the computing world. It gained popularity in the 1980s as a solution to the problem of how to link newer application to older legacy systems. Middleware is computer software that provides a link between separate software applications. It consists of a set of services that allows multiple processes running on one or more machines to interact. Middleware sits “in the middle” between application software that may be working on different operating systems. It is layer of software that lies between the application code and the run-time infrastructure. Middleware generally consists of a library of functions. The Component Template Library (CTL) is an implementation of the component technology based on C++ generic template programming. It can be used to realize distributed component-based software systems, where a component is a piece of software which consists of a well defined interface and an implementation. Interface and implementation are connected throw a communication channel, e.g. TCP/IP or MPI. The idea behind the CTL is to provide a mechanism which makes the development of distributed systems as easy as possible, so that the differences between traditional monolithic programs and complex distributed software systems nearly vanish.

Main design aspects for the CTL are:

- easy syntax (using overloading of suitable operators)
- header only (no precompile or install)
- independance of other libraries (needs only sockets, dlopen, pthreads,
- mpi/pvm can be used optionally)
- expandable for other communication protocols (open communication interface
- and protocol)
- maximal decoupling of components (using idl, abstract types)
- covering the parallel and the distributed programming models (group and links)
- direct process to process communication (no daemon in between)
- uniform behaviour of remote (tcp/ip, mpi, pvm, pipes, daemons) and local (library,
- thread) linkage types
- extrinsic usage of user defined types
- support of user defined types

3.PAK MULTIPHISICS

Laboratory of Mechanical Software at the Faculty of Mechanical engineering in Kragujevac has been working for over 30 years on developing its own software package PAK, based on the Finite Element Method. It is home grown general purpose FEM software. It contains modules for the pre- and post-processing and for solving problems in the fields of: solid mechanics (static and dynamic structural analysis, geometrical and material nonlinear problems, geomechanics, fracture mechanics material fatigue, biomechanics), heat transfer, fluid flow etc. The original methodology, based on published papers in the world journals and books was applied in numerous fields. Owing to the limitations of existing individual program modules in solving complex coupled problems, the need for fast development of software for multiphysics problems arose in recent years. Multiphysics problems are characterized by close coupling of multiple physical proc-

esses, often studied in different scientific disciplines and as the result of the new Ministry of science project, we should develop PAK Multiphysics software for solving the coupled multyphisics problems. The programming package PAK-Multiphysics will contain individual PAK modules for solid mechanics, fluid mechanics, heat transfer and flow through the porous media, electrostatics, magnetostatics, direct current flow and acoustics. In this paper we present, basic implementation of CTL software in PAK module in order to preview future coupling in PAK Multiphysics software.

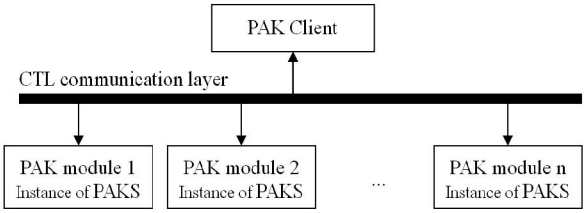

Figure 1 CTL communication layer

4.CTL COMMUNICATION LAYER

The CTL middleware can be used with different purpose: parallelizing of existing software, coupling of different applications, domain decomposition, etc. Our implementation into the PAK software is on basic level and it consists of simple communication between instances of same PAK module.

As we can see in figure 1, PAK Client communicate with other instances of PAK modules trough CTL communication layer. PAK Client initializes problem solving and read input data and delivers it to instances of PAK modules. They solve problem and give back results to PAK Client which print results into the output file.

5.COMPARISON STATIC LINKAGE VERSUS CTL LINKAGE

One main concept of the CTL is a generalization of linkage.
In the monolithic case an application is build up by the linker from a list of objects and dynamics or static libraries. For each called function the compiler wants to see its declaration. After compiling the linker first looks in the given list of objects and libraries for an definition of the called function and then binds the call to exactly one implementation. In this case all listed objects and libraries must be available at linkage time on the compiling machine. While run time this implementation will be executed on the same processor in the same process as the calling function.
The CTL gives both, the selection of the implementation and the bindings, in users hand. At run time it can be selected which implementation on which host to be linked in which mode. In the definition of the library the binding of the function signature to an implementation must be given.

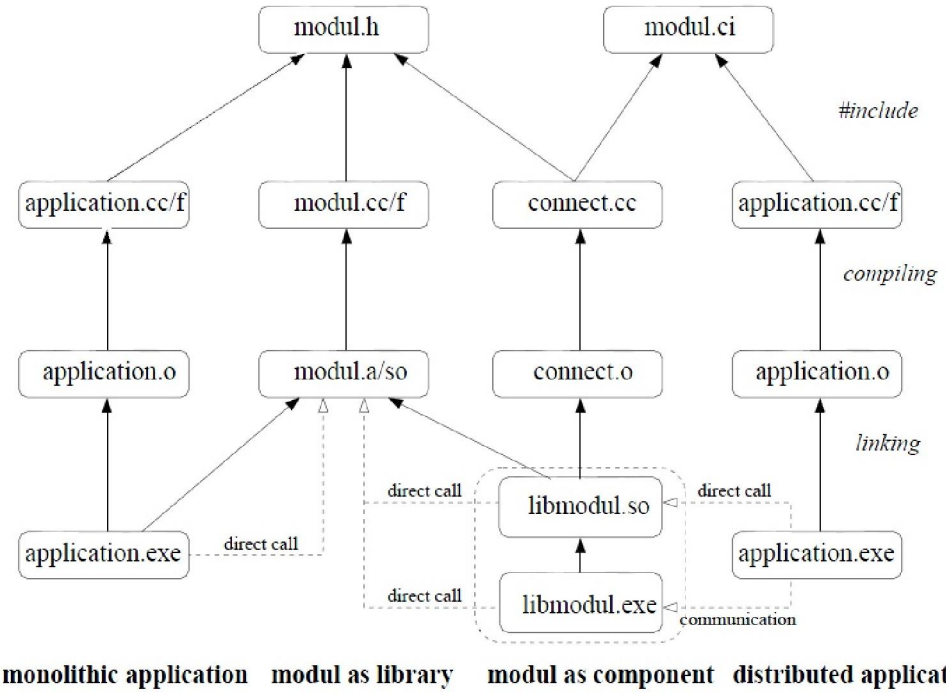

Figure 2. Dependence graph during classical connection (left) and component connection (right)

In our case, we have ***pak.exe*** instead of ***application.exe*** and definition of interface are given in paks.ci file as:

```
#include <ctl.h>
#ifndef _PAKS_CI
#define _PAKS_CI
#define CTL_Class paksci
#include CTL_ClassBegin
#define CTL_Constructor1 (const array<double>), 1
#define CTL_Method1 array<double>, structure, (const int4, const int4) const, 2
#include CTL_ClassEnd
#endif //__PAK_CI
```

This code defines connection between client and module, and it has to contain all subroutines which will be invoked remotely by client along with necessary parameters which are needed by those subroutines, followed by a number of parameters.

6.PROCESS OF DATA FLOW

There are several supported possibilities for data transfer between instances of modules. They are all defined in CTL and could be used for coupling of independent applications. In our case, we used shared object, and tcp/ip for coupling between PAK module and PAK Client as it is presented in Figure 3.

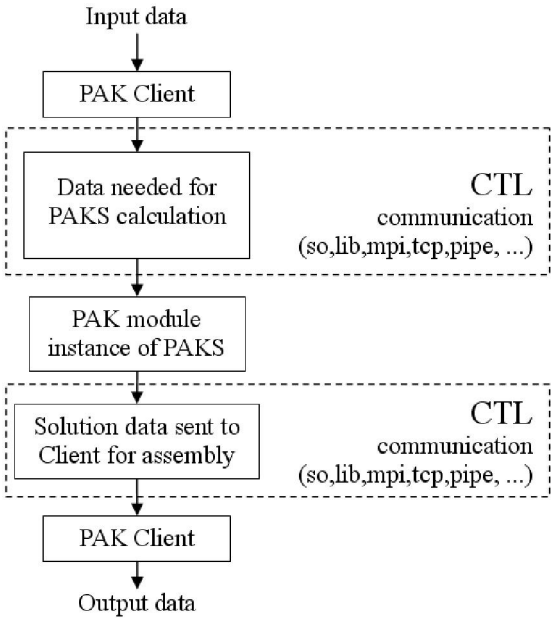

Figure 3 Data flow diagram

It presents flow diagram of data and possibilities for their delivery between PAK client and modules. CTL supports several ways of communication between client and modules, and they are:

- lib – behavior like classical static linkage
- thread – start in a separate thread each function
- tcp – use socket for communication
- pipe – use ssh to invoke services
- pvm – uses pvm for communication
- mpi – uses mpi for communication
- damons – uses sockets for communication

All those types of linkage are available, and existing of different linkage variants are thereby, in the sense of transparency, independent in syntax and program flow of the chosen linkage.

7.CONLUSIONS

CTL offers prepared library which has defined template for user's problem implementation. Process of assembling of our existing code into CTL template is simple because of good support and documentation, and because it is open source and it can be used without charge. Advantages of CTL as middleware component in development of PAK-Multiphysics product are great and the first step in its full implementation is made through this experiment. Possibility to couple modules through TCP/IP gives many advantages for network use and online accessibility. Presented information is necessary for further development and application in other fields of interest. Understanding of presented work is important for many programmers in order to expand benefit of theirs legacy software its and implementation in new applications which they are developing.

Note: This paper is written in frame of Ministry of science and technologic development project.

8.REFERENCES

[1] CTL Manual for Linux/Unix for the Usage with C++
[2] Miloš Kojić, Radovan Slavković, Miroslav Živković, Nenad Grujović, *Metod konačnih elemenata, Linearna analiza, Osnove nelinearne analize,* Faculty of Mechanical Engineering, University of Kragujevac
[3] Miloš Kojić, Radovan Slavković, Miroslav Živković, Nenad Grujović, *PAK-S, Program for FE Structural Analysis,* Faculty of Mechanical Engineering, University of Kragujevac