KRATOS A FRAMEWORK FOR MULTI-DISCIPLINARY CODE DEVELOPMENT

Pooyan Dadvand*, Kratos Team

* International Center for Numerical Methods in Engineering (CIMNE)
Universidad Politécnica de Cataluña
Campus Norte UPC, 08034 Barcelona, Spain
e-mail: pooyan@cimne.upc.edu , web page: http://www.cimne.com/

Key words: Multi-disciplinary, Object-Oriented framework, FEM, Numerical Anlysis

Abstract. *The world of computing simulation has experienced great progresses in recent years and requires more exigent multidisciplinary challenges to satisfy the new upcoming demands. Kratos, the framework created in this work, provides several tools for easy implementation of finite element applications and also provides a common platform for natural interaction of its applications in different ways. An innovative variable base interface, very efficient and flexible data structure, Python interface and different data are key features of this multi-disciplinary framework.*

1 INTRODUCTION

One of the relevant topics in the finite element method nowadays is the combination of different analysis (thermal, fluid dynamic, structural) with optimization methods, and adaptive meshing in one package with just one user interface and the possibility to extend the implemented solution to new types of problems, as an approach to a multi-disciplinary simulation environment. This was the origin of the Kratos as a framework which hides many difficulties in multi-disciplinary codes from applications developers and provides a standard interface for its applications which guarantees their interaction.

2 KRATOS, A MULTI-DISCIPLINARY FRAMEWORK

Kratos helps developers in implementing applications for different fields of analysis by providing input-output, data structures, solvers, basic tools, and standard algorithms. The applications implemented in this framework can be used for solving multi-disciplinary problems using any master and slave strategies or even by solving simultaneously. At this moment several solvers (incompressible fluid, structural, thermal, and electromagnetic) are implemented in Kratos. Combinations of these applications are also used to solve different multi-disciplinary problems, especially fluid-structure interaction and thermal-structural problems.

This framework provides a high level of flexibility and generality which is required for dealing with multi-disciplinary problems. Developers in different areas can configure Kratos for their needs without altering the standard interface used to communicate with other fields in coupled analysis. Also its python interface gives extra flexibility in handling nonstandard algorithms.

Several reusable components are provided to help developers allowing easier and faster implementation of their applications. Data structure, IO, linear solvers, geometries and different strategies are examples of these reusable components.

Kratos is also very extensible at different levels of implementation. Each application can add its variables, degrees of freedom, Properties, Elements, Conditions, and solution algorithms to Kratos. The object-oriented structure and appropriate patterns used in its design make these extensions easy while reducing the need for modifications. The extensibility is also validated at all levels by implementing different applications varying from standard finite element applications to optimization procedures using Kratos and its applications.

Last but not least, the performance of Kratos is comparable even to single purpose programs and different benchmarks show this in practice. This makes Kratos a practical tool for solving industrial multi-disciplinary problem.

3 USERS

Kratos is defined to be used by three groups of users at different levels:

- **Finite Element Developers** Kratos is defined to be used by finite element developers to implement a multi-disciplinary formulation easily. These developers, or users from Kratos point of view, are considered to be more expert in FEM, from the physical and mathematical points of view, than C++ programming. For this reason, Kratos provides their requirements without involving them in advanced programming concepts.
- **Application Developers** Kratos can be used as a finite element engine for other applications. This ability favors other teams of developers to work with Kratos. These users are less interested in finite element programming and their programming knowledge may vary from very expert to higher than basic. They may use not only Kratos itself but also any other applications provided by finite element developers, or other application developers. Developers of optimization programs or engineering design tools are the typical users of this kind.

- **Package Users** Engineers and designers are other users of Kratos. They can use the complete package of Kratos and its applications to model and solve their problem without getting involved in internal programming of this package. For these users Kratos provide a flexible external interface to enable them use different features of Kratos without changing its implementation. The interface to GiD provides an easy but effective way to deal with these users.

4 GENERAL STRUCTURE

An object-oriented structure has been designed to maximize the reusability and extensibility of the code. This structure is based on finite element methodology and many objects are designed to represent the basic FEM concepts. In this way the structure becomes easily understandable for developers with a finite element method background.

Kratos uses a multi-layer approach in its design which reduces the dependency between different parts of program. It helps in maintenance of the code and also helps developers in understanding the code. These layers are defined in a way such as each user has to work in the smallest number of layers as possible. In this way the amount of code that each user has to be familiar with is minimized and the chance of conflict between users of different categories is reduced. The implementation difficulties needed for each layer is also tuned for the knowledge of users working in it. For example the finite element layer uses only basic to average features of C++ programming but the main developer layer use advanced language features in order to provide desired performance.

5 VARIABLE BASE INTERFACE

Kratos uses a new variable base interface. All information about a concept or variable to be passed through this interface is encapsulated in the Variable class. The information about components of a variable also is encapsulated in the VariableComponent class which gives an additional flexibility to this interface. This interface is used at different levels of abstraction and proved to be very clear, flexible, and extensible.

Variable provides the type of data statically and objects can use it to configure their operations for a given type of data via template implementation. This type information also prevents the use of variables in procedures that cannot handle their type of data. Each variable has a unique key which can be used as the reference key in data structures. The name of variable as a string helps routines like IO to read and write them without requiring additional parameters. Finally it provides a zero value which can be used for initializing data independent of its type in generic algorithms. Beside this information, variable provides different methods for raw memory manipulations. These methods are excellent tools for low level generic programming, especially for writing heterogeneous containers.

6 DATA STRUCTURE

New heterogeneous containers have been implemented in order to hold different types of

data without any modifications. The DataValueContainer can be used to store variables of any type without even explicitly defining the list of them. This container is very flexible but uses a search mechanism to retrieve given variable's data. The VariablesListContainer only stores the variables defined in its variables list which can be have any type but its advantage is its fast indirection mechanism for finding the variables data. In Kratos these two containers are used alternatively in places where performance or flexibility is more important. Being able to store even the list of neighbor Nodes or Elements shows their flexibility in practice.

An entity base data structure has been developed in Kratos. This approach gives more freedom in partitioning the domain or in creating and removing Nodes and Elements, for example in adaptive meshing. Several levels of abstraction are provided to help users in grouping model and data information in different ways. In Kratos the Model contains the whole model, divided to different ModelParts. Each model part can have different Meshes which hold a complete set of entities in Kratos. These objects are effectively used for separating domain information or sending a single part to some process.

7 FINITE ELEMENT IMPLEMENTATION

The Element and Condition classes are designed as the extension points of Kratos. Their generic interfaces provide all information necessary for calculating their local components and also are flexible enough for handling new arguments in the future.

Several processes and strategies have been developed to handle standard procedures in finite element programming. These components increase the reusability of the code and decrease the effort needed to implement new finite element application using Kratos.

8 INPUT-OUTPUT

A flexible and extensible IO module for finite element programs has been developed. It can handle new concepts. Any application built with Kratos can use IO for reading and writing its own concepts without making any change to it. This IO is multi-format. It can support new formats just by adding a new IO derived class and without changing any other part of IO.

Kratos uses Python as its script language. This flexible interpreter with its object-oriented high level language can be used to implement and execute new algorithms using Kratos. In this way the implementation and maintenance cost of a new sophisticated interpreter is eliminated.

More information about Kratos can be found in Kratos webpage[i], Kratos developers webpage[ii] and Kratos wiki[iii].

REFERENCES

[i] Kratos Webpage, http://www.cimne.com/kratos/

[ii] Kratos' developers webpage , http://kratos.cimne.upc.es/trac/

[iii] Kratos Wiki, http://kratos.cimne.upc.es/kratoswiki/index.php/Main_Page