Shenol H. Yousouf, FMI, fac. no. M-21493
Director of studies: Vasil Georgiev
24.10.2007

System for Remote Execution of Applications and Services in a Hierarchical Grid

(Abstract)

This thesis is part of the GrOSD project, aimed at the construction of a lightweight grid middleware. The research and development of the grid platform GrOSD (Grid-aware Open Service Directory) are carried out in the context of the SUGrid Project at the Faculty of Mathematics and Informatics at the Sofia University "St. Clement Ohridski" and in partnership with the European Network of Excellence CoreGRID. The main purpose of the project is the development of a lightweight grid infrastructure, the priorities being simple architecture and implementation.

The purpose of the current thesis is the development of a system module for the remote execution of applications and services, which, in the context of the GrOSD project, reside and function in the nodes of the mentioned grid system. With the help of this module, the hardware resources, supplied for the needs of the grid, are being utilized for the implementation of distributed calculations. A main functional feature of the module is to provide an infrastructure for the transport to the nodes, the execution and the returning of the result for, in practice, any kind of applications, written by the grid clients, without being restricted to some specific requirements for the source code format, as it is usual in a number of other grid systems. Two kinds of services rendered lie in the focus of the project – tasks for one-time execution and tasks that are present in the nodes for longer periods of time and are being addressed multiple times with different input data. At the same time, the module features a flexible design that, on one hand, allows its usage in the framework of different systems, and, on the other hand, is a prerequisite for its extension towards execution of various and more specialized tasks that meet newer specific needs. This module is named Node Service, according to the adopted terminology of GrOSD framework.

First chapter of presentation focuses on the main field of Node Service application – namely, the grid architecture and a few particular implementations of grid systems. The chapter studies the lightweight grids in greater details by dwelling upon their major characteristics and technologies for construction.

In the second chapter, the grid model is rendered concrete in the context of the GrOSD project by carrying out a detailed analysis of GrOSD structure, a thorough examination of its composite modules and a survey of their functions and interactions. The necessity of Node Service implementation is introduced as a functional kernel for the execution of the computing services, offered by GrOSD to its customers. The general requirements for the execution module are enumerated – extensibility for rendering newer types of services, adaptability to changes in the grid system, to be prone to errors during task execution, full-scale utilization of provided node resources and unified channel of communication with the dependant central services in the system and other immediate clients. Node Service is compared with analogous executive modules in a few other grid systems, the specialized format of the applications, intended for these systems, being in the focus of the comparative analysis.

Third chapter is dedicated to a detailed and thorough overview of the three component parts of Node Service – *the Application Programming Interface (API)* for composing and sending requests to the grid nodes; *the executive kernel* which resides on nodes and is responsible for starting of the received tasks and for collecting and sending back the results; *the task framework subsystem* which acts as a connecting section for the other two components by defining the common terms which they are working with and agree upon.

The framework part consists of a set of interfaces. With their help, the concept and structural models of execution requests are outlined, the requests being constructed with specific implementations on client side by the API and handled and served by the executive kernel on node level. This module also contains the central interface of the Node Service, which declares a set of services offered for direct usage to the API, as well as the interfaces of the listener objects and events of execution which the API clients can be notified about. The notification is performed asynchronously in relation to the moment of submitting the request which means that the clients are not being blocked until getting the result.

The purpose of the Node Service API is to act as a front-end for the services provided by the executive kernel. It has two major functions – 1) to provide concrete implementations of framework interfaces together with factory classes for composing the request elements and the request itself; 2) to provide the clients with a transparent access to the functions and resources of Node Service upon a particular node, based on the capabilities of the Java RMI technology.

The Node Service executive kernel is delegated with the logic for processing the received metadata from the request, formed according to the framework subsystem specifications, to compose the actual request, to invoke and to collect and send the results back after that. Java Reflection API is the underlying technology here. Special features in executive kernel implementation are as follows: 1) *specialized processor modules* towards which the different types of requests are being dispatched; this renders the kernel architecture extensible for serving new types of requests, which is made possible by simply adding the appropriate processor for them to the kernel; 2) task execution in separate *threads* – thus, the kernel is enabled to serve several requests at the same time; 3) a centralized control over the threads of execution by the means of a specifically organized for this purpose *registry* for tasks and services; 4) *a persistent model* of client tasks for multiple invocations with various input data supplied to the node; the persistence is achieved by storing the metadata for each service into a separate file, from which they can be restored on kernel restart.

Fourth chapter illustrates the service application in two use-cases – 1) in a simplified case, outside the grid context – in this scenario, the client code can immediately use the Node Service API capabilities for creating a request and its load for execution; 2) the details of Node Service operation in integrated mode along with the other GrOSD services, with an emphasis on the interactions with them – in this case, the Node Service clients and their functions are being specified in the environment of GrOSD.

Fifth chapter provides short instructions about Node Service installation on node side and on client side, the installation being completely simplified because of the single requirement for its operation – usage of JRE version 5.0 or higher. This guarantees the maximal coverage of potential donor nodes for the service operation.

Sixth chapter gives details for the used technologies, for which the major requirement is to be non-commercial. The emphasis is put on Java and Java RMI, which provide for code mobility and transparent communication; the Java Reflection API which makes it possible to process any precompiled class; the Java design patterns which provide effective solutions for the problems that appeared during the development of the Node Service components.

The last chapter of presentation is specifically focused on the extensibility of the services, provided by Node Service, which is achieved through its architecture flexibility, the capsulation of the concrete implementations of the request components into the API and the specializations of the processing modules inside the executive kernel. The broad perspectives for the Node Service development in relation to the GrOSD project are being outlined and a projection is being made about the possibilities for its modification for independent operation in a peer-to-peer network. In the latter case, Node Service takes over a part of the functions of some of the central services in GrOSD plus other features.

The appendix contains the source code of a client program which illustrates how to work with the Node Service API in relation to composing the request, how to send it to a particular node and how to get the result.