ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Bari

INFN/TC-02/17
11 Luglio 2002

SIMULATING A DATA GRID ENVIRONMENT

Marcello Castellano[1,2], Giacomo Piscitelli[1,2], Fabrizio Tarricone[1]
Domenico Di Bari[2,3], Daniela Cozza[3]

[1)] *Department of Electrical and Electronic Engineering, Polytechnic of Bari, Via Orabona 4, 70126 Bari, Italy*

[2)] *INFN-Sezione di Bari, Via Orabona 4, 70126 Bari, Italy*

[3)] *Dipartimento Interateneo di Fisica dell'Univ. di Bari, Via Orabona 4, 70126 Bari, Italy*

Abstract

In this paper a Java-based, multi-thread toolset is proposed for data-intensive grid simulations. The toolset is a Java package composed by class library, one for each grid entity such as Computing Element, Storage Element, Scheduler, Replica Catalog, Information Server and Users. Moreover a kit has been implemented to describe a wide-area network infrastructure. The implemented Process Network model assumes each process equipped by inner structure oriented to multi-tasking and time-sharing processing. The simulator framework is based on multi-threads technology both to obtain a natural code description of the model and to avoid bottleneck problems due to centralised structures. The toolset is also adequate to allow the description of more general distributed system. The package has been with successfully validated considering real configurations. It is also running to evaluate scheduling algorithms for a simulated scenario in agreement with a scientific data-intensive applications foreseen by a founded European data grid project.

PACS.: 07.05
Keywords: Computational Grids, DataGrid, Simulation, Distributed System

Published by ***SIS–Pubblicazioni***
Laboratori Nazionali di Frascati

1 INTRODUCTION

A grid computing system is composed by entities belonging to the following types: fabric, low level middleware, user level middleware, application. The type *fabric* concerns with computers, clusters, network, scientific instruments, and their resource management systems. Low and User level middleware regard the core services that a grid must provides to reach the goal and tools for grid programming, respectively. Problem solving activity using a grid defines the type *application*3).

Designing components for grid based systems is particularly challenging: both the software and hardware resources of the underlying system may exhibit heterogeneous performance characteristics, resources may be shared by other users, and networks, computers and data may exist in distinct administrative domains. Moreover, data-intensive, high-performance computing applications require the efficient management and transfer of terabytes or petabytes of information in wide-area, distributed computing environments. Examples of data-intensive applications include experimental analyses and simulations in several scientific disciplines, such as high-energy, climate modeling, earthquake engineering and astronomy. In this paper a Java-based, multi-thread toolset is proposed for data-intensive grid simulations. In section 2 is described the logical plane of the simulator with the components. The implementation and the validation results are discussed in the last section.

2 MODELLING DATA GRID LOGICAL COMPONENTS

The grid based system is built by collaborative components each ones is an active entity devoted to solve specific tasks. Processes Network (4) (5) strategy is taken into account to design the grid. In this conceptual plane of description, the entities are represented by concurrent processes that communicate each other by messages. Each consumer process refers to a specific buffer to pick up the incoming messages that have been queued in the time by producers processes. This is an asynchronous message passing mechanism that avoid the producer wait for the receiver ready state. Each component shows an architecture pipeline oriented to provide a multi-task environment. If a stage named A needs of collaboration of another entity to reach its goal, it query such entity and push the request in next stage B queue. At the external entity completion time the stage B resumes the processing. In meantime the component can continue with processing of other requests. The most of Grid components are time shared resources. To take into account a such behaviour, the resources are modelled like servers with Processor Sharing (PS) service policy (6). In other words, each request is served for a quantum, expired it the control is passed to service next ready state pending request.

2.1 Storage Elements

Storage Element (SE) grid entity allows the uniform access to data. It is a time shared component based on PS policy with in a not pipelined service. However the one-stage component requires a pending input queue to retain more requests in hold state than (more limited) available “at the same time” in the execution phase.

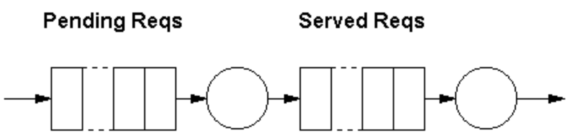

Fig. 1: Storage Element model.

The necessary time to execute one request depends on the nature of operation requested to SE. If it is processing a reading request:

$$\text{Execution Time}\,[\text{s}] = \frac{\text{Data Volume}[\text{MB}]}{\text{Speed Reading}\,[\text{MB/s}]}$$

Indeed, the SE execution time for a write request is:

$$\text{Execution Time}\,[\text{s}] = \frac{\text{Data Volume}\,[\text{MB}]}{\text{Speed Writing}\,[\text{MB/s}]}$$

2.2 Computing Elements

Computing Element (CE) grid entity executes the jobs. The execution of a job is arranged in phases as follow:

1. query of a SE to obtain input data file;
2. execution;
3. writing of results on a SE.

The CE model is shown in **Fig. 2**. The stage 1 queries the suitable SE for input data file. The stage 2 waits for requested data from the SE. The stage 3 is designed for the execution with PS policy in time sharing mode. The input stage S carries out functions of shunting of messages: it sends execution request to stage 1 and forward the answers from SEs to stage 2.

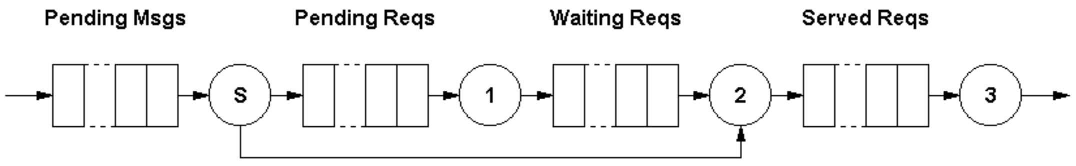

Fig. 2: Computing Element model.

The CPU time to execute a job is evaluated as:

$$\text{CPU Time [s]} = \frac{\text{Job Weight [SPECint95} \times \text{s/MB]}}{\text{Power [SPECint95]}} \times \text{Data Volume [MB]}$$

2.3 Information System

Grid Information System is a grid structured component composed by three entities: *Replica Catalog* (RC), *Information Server* (IS) and *Reporter*. RC localises data files and the replicated versions of the same files disseminated around the grid. IS takes care to supply and to update information on physical resources (CEs and SEs) of Grid environment. The Reporter periodically sends to ISs the updated information on physical resources. Both the RC and the IS execute tasks in atomic way; The model is based on the simple First Come First Served service policy queue system [(6)].

2.4 Scheduler

The Scheduler plans execution of jobs on Computational Grid. Phases of scheduling request processing are:

1. RC query to obtain SEs list that hold a copy of input data file necessary to the execution of the job;
2. IS query to obtain CEs list “near” to the SEs obtained in previous phase;
3. IS query to obtain information on CEs obtained in the previous phase;
4. choice of CE / SE couple which to demand the execution of the job.

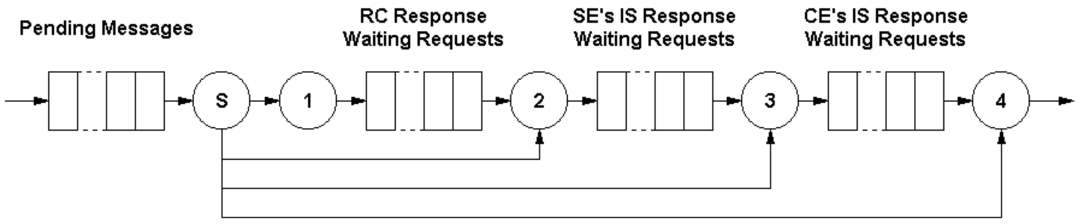
Pending Messages
RC Response Waiting Requests
SE's IS Response Waiting Requests
CE's IS Response Waiting Requests
S
1
2
3
4

Fig. 3: Scheduler model.

In **Fig. 3** is reported the S stage that carries out shunts messages towards the suitable stages.

2.5 Network

Each network *segment* uses a Time Division Multiplexing technique and a full duplex mechanism to allows a bi-directional transmission. The segment model is based on a couple of processes: one for the forward communication and the other one for the communication in backward as shown in **Fig. 4**

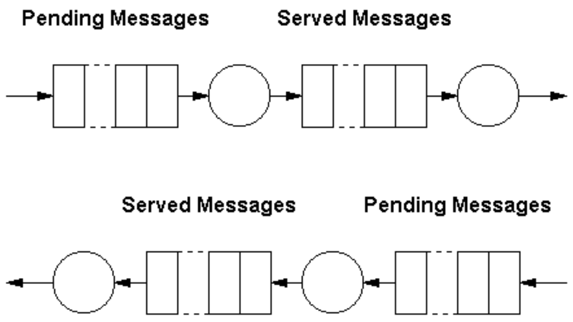

Fig. 4: Network model.

The transmission time to delivery a message is here computed according to the follow relation:

$$\text{Trasmission Time [s]} = \frac{\text{Message Dimension [MB]}}{\text{Throughput [MB/s]}}$$

2.6 Users

The users grid component is assumed as a jobs source with an exponential inter-arrival time distribution with rate _ and high variable job rate. The job is characterized by a high computational weight and data-intensive applications. Users and job models have been motivated by the computing model that describes the data-intensive networked analysis at regional centres for high energy physics experiment at CERN [(8)]. Table I shows the users and job features.

Tab. I: Users and Job features.

CHARACTERISTICS	MEASURED IN	Typical values are
Request Rate	[1/s]	$11.57\_10^{-6} \div 92.59\_10^{-6}$
Jobs Weight	[SI95·s/MB]	$102.4 \div 512.0$
Input Data Volume	[MB]	$1.049\_10^{3} \div 1.049\_10^{6}$
Output Data Volume	[MB]	$104.9 \div 104.9\_10^{3}$

3 IMPLEMENTATION AND VALIDATION RESULTS

The described model has been implemented with *multi-thread* technology. Every modelling process is described in a thread equipped with a queue that receives pending messages [(8)]. In this way it is obtained a direct correspondence among real entities, modelling processes and implementing threads. Moreover, centralised structures, like scheduler and event lists, are not used. In fact every thread possesses a queue with messages that others threads send it directly, and it accesses to own queue and assumes the necessary behaviour. In this way it is simpler adapt the Simulator to a distributed simulation environment, since various threads can be located on different nodes. The Simulator has been developed in Java because it is a pure objected-oriented language, it allows natively the realisation of multi-thread applications and pointers and memory are automatically managed by the Java Virtual Machine (JVM). Java is a real and total portable language. In the implementation of the model some of activities that entities complete, are really executed. Others, like data read, data write, jobs execute, transmission on network, are simulated rendering the implementing thread not available for necessary time to execution.

Validation phase, consists in determining how much is accurate the representation that it supplies of the simulated system 0. In order to realise this phase, it is necessary to dispose measures obtained on the real system to compare them with simulation results [(10)]. A systematic study to define a model for complex data and computing at the Centre European for Research in Nuclear Physics has been done by MONARC Project [(11)]. Moreover a real testbed in the small for data-intensive distributed system has been realised by MONARC members [(12)] with job and system characteristics shown in Tab. II. A simulation experiment has been realized according to the system developed in MONARC project. Execution times of 1, 2, 4, 8, 16, 32 concurrent identical jobs are been measured.

Tab. II: Job and System characteristics for testbed in the small.

Request Rate [1/s]	$11.57\_10^{-6} \div 92.59\_10^{-6}$
Input Data Volume [MB]	36.89
Job Weight [SI95·s/MB]	6.629
CE Power [SI95]	17.4
SE Speed Reading [MB/s]	31
Network	Ethernet

Measures obtained by testbed and those obtained through simulation with relative and absolute errors are compared in Tab. III. A satisfactory mean relative error less than 10% can be pointed out.

Tab. III: Measured Execution Time vs Simulated Execution Time.

Concurrent Jobs	Measured Execution Time [s]	Simulated Execution Time [s]	Relative Error [%]	Absolute Error [s]
1	22,08	18,29	17,19	3,80
2	23,43	20,95	10,58	2,48
4	30,63	32,25	5,30	1,62
8	42,40	43,18	1,83	0,78
16	77,50	83,75	8,06	6,25
32	151,59	166,72	9,98	15,13
Mean	-	-	8,82	5,01

4 CONCLUSION

This paper introduces the concepts, mechanisms and results of a Java-based, multi-thread toolset for a data-intensive grid simulator. It makes possible a rapid virtual prototyping by a natural description of a data-intensive grid based system. A good agreement between testbed in the small and simulate results has been obtained. Nowadays the toolset is running to evaluate scheduling algorithms for a simulated scenario in accord with a scientific data-intensive applications foreseen by a founded European data grid project.

5 REFERENCES

(1) I.Foster, J.Geisler, B.Nickless, W.Smith, S.Tuecke – **Software Infrastructure for the I-WAY High-Performance Distributed Computing Experiment** – *Proc. 5th IEEE Symposium on High Performance Distributed Computing, 1997*

(2) M.Baker, R.Buyya, D.Laforenza – **The Grid: International Efforts in Global Computing** – International Conference on Advances in Infrastructure for Electronic Business, Science and Education on the Internet, l'Aquila, ITALY, August 2000

(3) I.Foster, C.Kesselman – **The Grid: Blueprint for a Future Computing Infrastructure** – Morgan Kaufmann Publishers, July 1998

(4) J.Banks – **Introduction to Simulation** – *Proc. of the 2000 Winter Simulation Conference, Orlando, Florida, USA, December 2000*

(5) J.Davis, C.Hylands, B.Kienhuis, E.A.Lee, J.Liu, X.Liu, L.Muliadi, S.Neuendorffer, J.Tsay, B.Vogel, Y.Xiong – **Heterogeneous Concurrent Modeling and Design in Java** – *Tech. Mem. UCB/ERL M01/12, EECS, Univ. California, Berkeley, March 01*

(6) K.Aida, A.Takefusa, H.Nakada, S.Matsuoka, S.Sekiguchi, U.Nagashima – **Performance Evaluation Model for Scheduling in Global Computing Systems** – *The International Journal of High Performance Computing Applications, vol. 14, n. 3, pp. 268-279, 2000*

(7) L.Kleinrock – Queueing Systems – *John Wiley & Sons, Inc. 1975*

(8) MONARC Members – Models of Networked Analysis at Regional Centres for LHC Experimens (MONARC) – Phase 2 Report – *March 2000*

(9) P.Metz, J.O'Brien, W.Weber – **Message Queue Concept – An Implementation Pattern for Concurrent Objects** – *Object-Oriented Modeling of Embedded Realtime Systems, Herrsching, Ammersee, GERMANY, May 1999*

(10) T.J.Schriber, D.T.Brunner – **Inside Discrete-Event Simulation Software: How it Works and Why it Matters** – *Proc. of the 2000 Winter Simulation Conference, Orlando, Florida, USA, December 2000*

(11) J.P.C.Kleijnen – Validation of Models: Statistical Techniques and Data Availability – *Proc. of the 1999 Winter Simulation Conference, Phoenix, Arizona, USA, December 1999*

(12) I.C.Legrand – The MONARC Toolset for Simulating Large Network-Distributed Processing Systems – *Proc. of the 2000 Winter Simulation Conference, Orlando, Florida, USA, Dicembre 2000*

(13) Y.Morita – **Validation of the MONARC Simulation Tools** – Computing in High Energy Phisics, Padova, ITALY, February 2000.