Topos - Manager for Distributed Computing Media

Sergey S. Kosyakov Serge A. Krashakov Lev N. Shchur
Landau Institute for Theoretical Physics
142432 Chernogolovka, Russia
tel/fax: +7 (095) 913 2317
e-mail: ks@itp.ac.ru (Sergey Kosyakov, contact person)

Abstract *The sophisticated management system for the large-scale simulations in computational physics are presented. Our main goal is in the design and realization of the middle-ware software, which solve problems of the interoperability and the software portability. An efficiency of system is demonstrated in a real computer experiment on the probability of incipient spanning clusters measurement, done by us recently.*

Keywords: distributed computing, ILU, CORBA, Java, WWW

1 Introduction

An efficient and one-touch-one-move-one-click user interface for a file-task-item-resource-job manipulating system is an old dream of scientists, working in a different areas of computational physics, chemistry, biology, etc.

Computational scientists (physisist, chemist, biologist, geologist, etc. - choose the one you like) are usually operates with thousands of files, figures, tables, and texts, using a number of computers, disks, printers and other peripheral equipment. The hand management of a such system is terrible task even for handyman.

Here, we present a system Topos, designed for the management of large-scale computations.

Computational physicists are use usually a term 'computer experiment' mentioning large-scale simulations, and this term is most suitable one for the problem considered.

How we shall plan the experiment? First, we have in hand a number of external parameters $P1, P2, ..., PN$ (examples are magnetic field, loading current, temperature, etc.). All this parameters should be varied in some ranges $P1_{min} < P1 < P1_{max}$ with step dP. So, we should manipulate with the rather huge table of input parameters and their values.

Second, we expect to measure some quantities $Q1, Q2, ..., QN$ as a functions of values of input parameters $P1, ..., PN$. In practice, we needs to have some correspondence between input and output parameters, and vice versa.

Third, we build up a number of apparatus to measure this quantities. (The apparatus corresponds to the procedure in program language).

Fourth, we should connect all apparatus with some data acquisition equipment. (Compare with a database).

Fifth and final, after success with the experiment we need some more or less sophisticated tools for the data analysis and presentation (e.g., an automated table generation).

This formal steps are exactly the same for large-scale computations ($\equiv$ experiment) as well.

Difficulties were arose from several facts: i) the set of computed data is huge (growth combinatorially with the number of parameters and its values), ii) one needs to manage many computers available, and iii) this computers are usually based on different platforms.

So, spreading of a hundreds of processes with accurate in/out data holding on a number of computers by hand gives the definite headache to researcher.

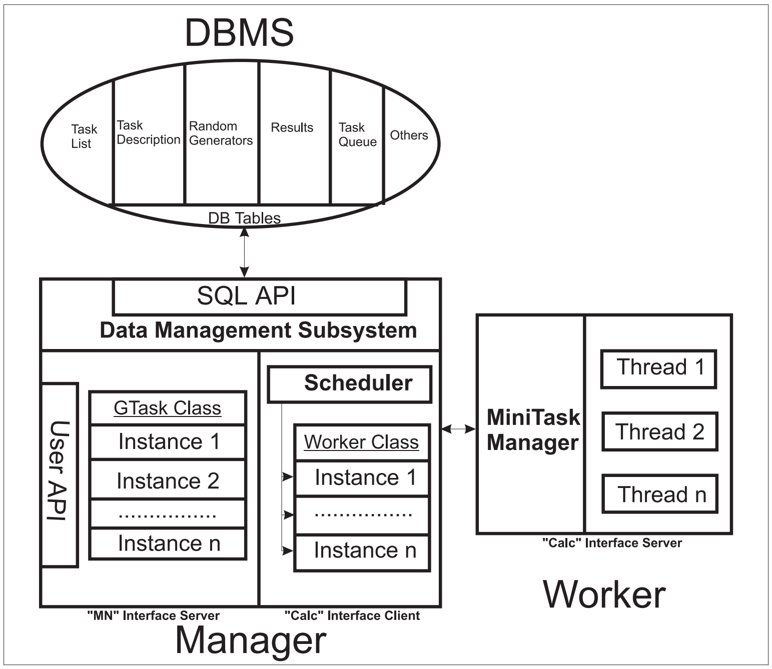

Figure 1: Topos detailed structure

In the next section we formulate our vision of the way to solve problem. In third section, main goals of the Topos system are emphasized. Topos objects are introduced in section 4. Development platforms are described in section 5. The first example of the computer experiment down by the Topos manager is briefly presented in section 6. Our concluding remarks are given in section 7.

2 Problem formulation

In fact, we formulate in above the problem of the management of the heterogeneous computing media, arised in the large-scale computations. Clearly, one needs the automated distributed system for the Tasks Management as well as for the data acquisition, the data analysis and the data presentation. Second, such system should be portable and easily reachable. Third, a presence of open interfaces on all levels of system is very crucial point.

It should be noted here, that our approach is quite universal, and that the main elements of our system could be used in the construction of information systems in a wide range of particular problems.

The reason to start design with the namely Tasks Manager is quite natural: we met a problem of mathematical physics which needs an enormous computations to be down. And we successfully make up the first computer experiment [3] using Topos.

We find out that the OMG CORBA technology provides all requirements we needs, and Java applets are suitable enough for the building of the open interface. The detailed analysis will be published elsewhere.

3 Design goals

Main design goals of the Topos System:

1. The system can be reachable by the Web browser or some other client supporting Java applets - this approach is clearly excludes problem of the user interface portability, and does "open" the system.

2. The automated "task environment" support - internal task parameters are hidden in a human, intuitively understandable

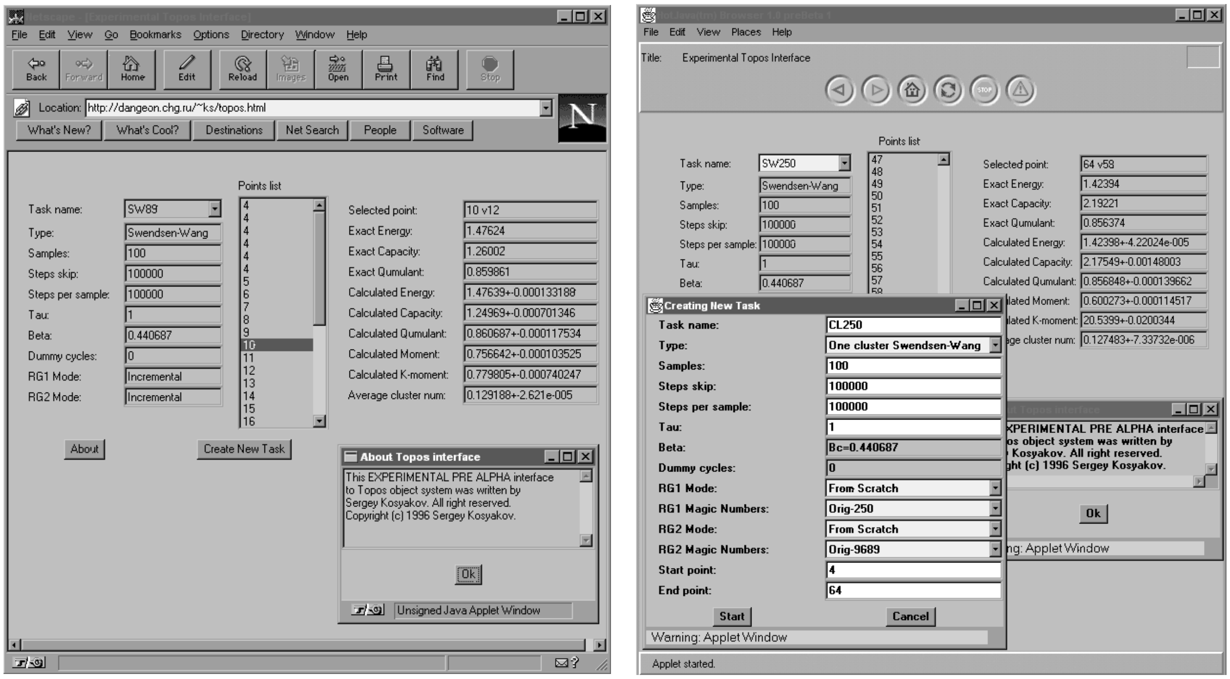

Figure 2: Topos user interface examples

object Task. The user is operates with descriptive (human level) task names, and not with the set of parameters.

3. The system is own: i) the results repository and the automatic track of parameters in desired region; ii) the possibility for recalculations and additional calculations for any series with the original environment.

4. Concurrent calculations on a number of computers with the different architecture, possibly with the automatic balance loading.

5. The portability, an independence on low level layers (DBMS, transport, etc.) (Any available Workstation could be used). Such independence is reached via using some middleware between the core of Topos and the DBMS.

6. The robustness: Auto-restart after any host(s) breakdowns, which is extremely important in a huge distributed experiment, could lasts in months.

7. The Topos System is the upgradable one. This feature is reached using the native CORBA approach: all Topos modules are described in terms of the CORBA like interface specification language.

4 Topos objects

Three classes of CORBA objects are used - a Worker, a Task and a Manager, as shown in Figure 1.

The "Task" class - this is the class of objects, which are held by the Data Management Subsystem (which is a middleware between DBMS and the rest of the world). Each Task object held the whole information about the particular user task as well as it held calculated results and dynamic elements like initial seeds for random number generators. User interface operates with tasks through the methods of this objects.

The "Worker" class - objects, which provides actual calculations. Workers are spreaded over number of hosts in the Net. They are controlled by the Manager and they are not visible to user. Jobs (the ones in the usual sence) are distributed among a set of Workers (in a manner like balance loading).

The "Manager" class contains the only object which controls the whole system. It controls Worker loading as well as decision on restarting of tasks after some OS failings, and it provides the job queue management (The Scheduler in Figure 1).

We write out almost all components using C language, for the obvious reason of a good performance. Thereafter, the user interface were written as Java applets, and one could use any WWW browser with Java support to browse Topos, get results of interest, and even start new series of computations (a new task).

5 Development platforms

The entire system is based on the freely distributed CORBA 2.0 [5] implementation named as Inter Language Unification System (ILU) [6], developed by Xerox Corporation. Many thanks to Xerox Corporation.

The Data Management Subsystem based on Postgres95 (now PosgreSQL) [7] DBMS. It is free of charge, with the good functionality and is the portable one. Because of ILU properties, Topos can run on a wide range of platforms - from MS Windows (3.11, 95, NT) to the whole set of Unix'es (Digital Unix, SunOS, Solaris, AIX, HPUX, etc.).

Really, the test-bed was based on: three Digital Alpha's (two departmental servers and a workstation) and a number of FreeBSD boxes, spreaded over the Chg-NET (Network of Science Park in Chernogolovka).

The user interface part is based on the ILU release, written completely using Java - JYLU [8]. It was checked with Netscape Navigator v.3.0 (Netscape Communications) and Hot-Java (Sun Microsystems). We do not see any problems to run this applets on any JDK-1.01 supported platforms.

Newest versions of Netscape Communicator 4.0 supports VisiBroker for Java (Visigenic) which could work with ILU as well. We hope, new versions of Topos will work with Netscape Communicator.

6 Mathematical Physics Experiment

First version of the Topos system were used recently by two of us in the large-scale computer experiment [3]. We checked numerically a new statement on mathematical physics, formulated by Aizenman [2]: the number of Incipient Spanning Clusters (ISC) in two dimensional critical percolation can be larger than one, and that the probability of at least k separate clusters is bounded

$$A\ e^{-\alpha\ k^2} \leq P_L(k) \leq e^{-\alpha'\ k^2}. \qquad (1)$$

We investigate by Monte-Carlo the number of spanning clusters in the critical bond percolation model on two dimensional square lattices. We have determined the numerical values of the probabilities

$$P(k) = \lim_{L\to\infty}\ P_L(k) \qquad (2)$$

for $k = 1, 2$ and 3 and analysis of our data gives $P(k > 1) = 6.58 \cdot 10^{-3}$ with error $\approx 3 \cdot 10^{-5}$ and $P(k > 2) = 1.48 \cdot 10^{-6}$ with uncertainty $2.1 \cdot 10^{-7}$ for square bond percolation with Free Boundary Conditions (FBC). Those for Periodic Boundary Conditions (PBC) in one direction are $P(k > 1) \approx 2.0(4)\ 10^{-3}$ and $P(k > 2) \approx 1.4(5)\ 10^{-7}$ We got the data of high quality due to the use of lattices which are self-dual and deeply symmetric for any finite size L.

The experiment were performed using the Web browser under FreeBSD and Win95, the manager running on the DEC Alpha station, and simulations performed on a number of DEC Alpha stations and Intel Pentium's running FreeBSD, and distributed over Chg-NET.

7 Conclusions

Despite of alpha-status of Topos system (see Figure 2), we could definitely argue the CORBA technology is the right choice for us.

The system can be easily tuned to wide range of tasks. Topos is really portable system and does not depend on the choice of the specific DBMS, compilers, OS, CPU platform, etc.

Moreover, recently, we carry out the computer experiment on measuring of probabilities of Incipient Spanning Clusters in square bond critical percolation [3]. All computations were down using Topos. Our practice confirm completely our vision of Topos system as we claims on above in subsection 'Design goals'.

Our final message is that combination of the CORBA technics with Java and WWW opens new horizons for the development of information resources and of (really) open interfaces.

References

[1] S.S. Kosyakov, S.A. Krashakov, and L.N. Shchur, in *Proceedings of Second Workshop on IS and DB* eds. Yu.I. Zhuravlev, L.A. Kalinichenko, and Yu.E. Hohlov (RFBR, Moscow,1995).

[2] M. Aizenman, *Nucl. Phys. B [FS] i* ***485**, 551 (1997).*

[3] Lev N. Shchur and Sergey S. Kosyakov, Int.J.Mod.Phys..C: **8**, 473-481 (1997) and `http://www.wspc.com.sg/journals/ijmpc/83/ijmpc83.html`.

[4] S.S. Kosyakov, S.A. Krashakov, and L.N. Shchur, in *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications* (Las Vegas, 1997)

[5] *OMG CORBA v2.0 specification* http://www.omg.org/coriiop/corb2prf.htm

[6] *ILU - Inter Language Unification System* Xerox Corporation ftp://ftp.parc.xerox.com/pub/ilu/ilu.html

[7] *Postgres95 Database Management System* University of California at Berkeley ftp://s2k-ftp.cs.berkeley.edu/pub/postgres95/index.html

[8] *JYLU - an implementation of part of the ILU kernel and a Java run time purely in Java* http://www-db.stanford.edu/~hassan/Java/Jylu