Topos - Manager for Computing in Distributed Computer Environment

Sergey S. Kosyakov
Landau Institute
Chernogolovka,Moscow region
142432 Russia
+7 (095) 913 2317
ks@chg.ru

Lev N. Shchur
Landau Institute
Chernogolovka, Moscow region
142432 Russia
+7 (095) 913 2317
shchur@chg.ru

ABSTRACT

The sophisticated management system for the large-scale simulations in computational physics is 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 the system is demonstrated by a real computer experiment on measurement of the probabilities of incipient spanning clusters.

Keywords

Distributed computing, CORBA, WWW.

1. INTRODUCTION

Real computer experiment contains many stages, beginning from an idea of experiment and ending with the report publication (Fig.1)

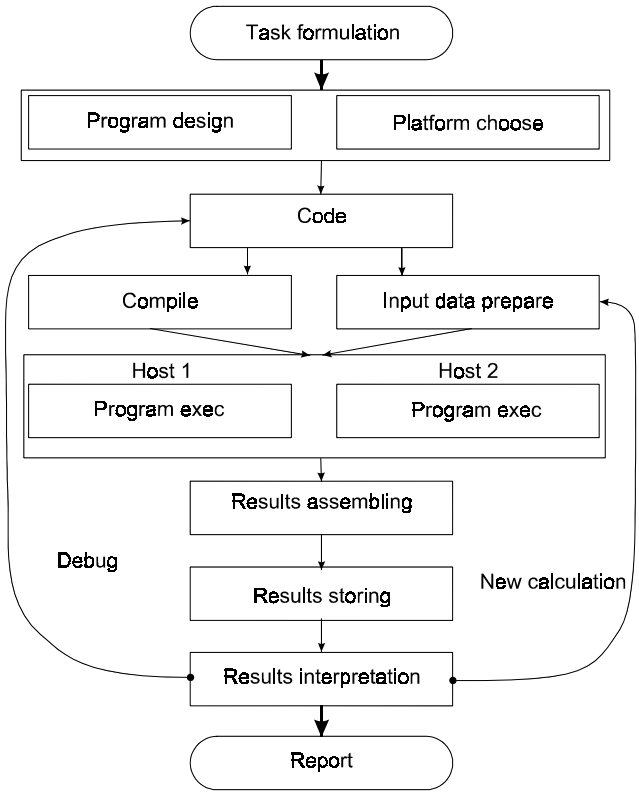

Figure 1. Stages of the computer experiment.

Some stages, like task formulation, program design, coding, etc. - are the human-specific ones, and cannot be automated, at least at the present time. Nevertheless, there are still enough room in Fig.1 to we offer a system, which dramatically decrease the volume of hard work at the stages of computations, results assembling and storing. Especially in the case of the cluster computations become more and more popular nowadays.

Computational scientists (physicists, chemists, biologists, geologists, etc.) are usually operates with thousands of files, figures, tables, and texts, using the number of computers, disks, printers and other peripheral equipment. The hand management of such system is a terrible task even for handyman. We present here our approach realized in a system Topos, designed for the management of large-scale computations.

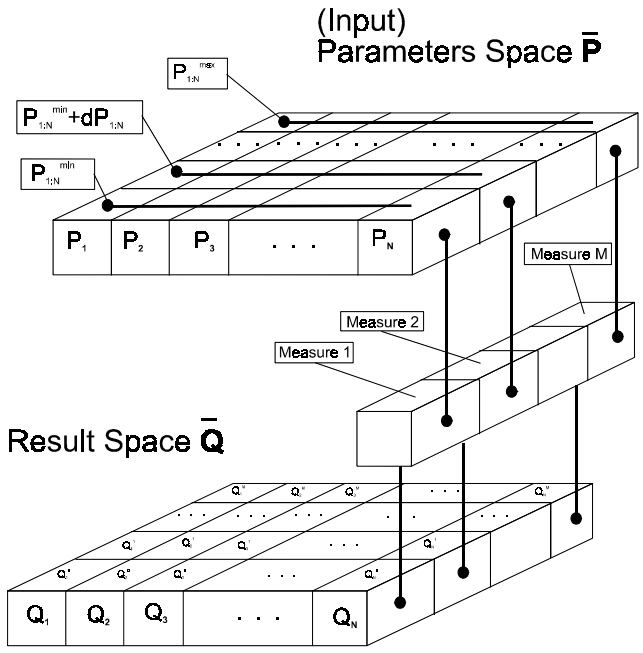

Figure 2. Computer experiment as mapping.

Computational physicists usually use the term 'computer experiment' mentioning large-scale simulations, and this term is

most suitable one for the problem considered here. Let us analyze now the stages of such experiment.

First, we have in hand a number of external parameters P_1 ,P_2, ..., P_N (among examples are the magnetic field, the loading current, the temperature, etc.). All this parameters are usually varied in some ranges $P_{1,min} < P_1 < P_{1,max}$ with the step dP. So, we have to manipulate rather huge table of the input parameters and their values.

Second, we have to measure some quantities Q_1, Q_2, ..., Q_N as a functions of values of input parameters P_1, ..., P_N. Practically, we have to determine some function, which maps the input parameter space onto the output parameter space, as shown schematically in Fig.2

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

Fourth, we have to plug all apparatus to some data acquisition equipment. (To be compared with the database system).

Fifth, and finally, we have to have some more or less sophisticated tools for the data analysis and for the data presentation (e.g., an automated table generation).

It is clear that these formal steps are in one-to-one correspondence in the real experiment and in the computer experiment.

What is the main difficulties aroused in the large-scale simulation? They comes from the following facts:

i) the set of the output data is huge enough and growth exponentially with the number of input parameters and its values,

ii) the huge number of computers in the network have to be downloaded, checked, and managed,

iii) these computers are usually from various vendors, they have different architecture and different operation systems.

So, it is not an easy task to spread the hundreds of processes with accurate in/out data holding on a number of computers by hand, and gives the definite headache to the researcher.

2. FORMULATION OF THE PROBLEM

We formulate in above the problem of the heterogeneous computing media management, raised in the state-of-the-art large-scale computations. Clearly, one needs the Task Management for the distributed computing, the data acquisition, the data analysis and the data presentation. Second, such system should be portable and easily reachable. Third, a presence of the open interfaces in the all levels of the system is very crucial point. Last, the robustness is the important feature of the complicated system.

It should be noted here, that we were developed the quite universal approach, and that the main elements of our system could be used in the construction of information systems for the wide range of applications.

The reason to use our approach firstly to build up namely the Tasks Manager is quite simple: we met a problem of mathematical physics, which required enormously large computations to be down. We successfully finished the first computer experiment using Topos system.

We choose the OMG CORBA technology because we find out it provides all requirements we needs, and Java applets which is suitable enough to build up of the open interfaces. The detailed analysis of the systems will be published elsewhere.

3. DESIGN GOALS

Main design goals of the Topos System:

1. The system are 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 object Task. The user 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 additional series and subseries.
4. Concurrent calculations on a number of computers with the different architectures, and with the automatic balance loading.
5. The portability, independence on low level layers (DBMS, transport, etc.) (Any available Workstation can be used). Such independence is reached thanks to some middleware between the core of the Topos and the DBMS.
6. The robustness: Task auto-resubmission after any host(s) breakdowns, which are extremely important in a huge distributed experiment. Some the experiment could lasts in months despite of the external influences like the occasional power shutdowns.
7. The Topos System is upgradable. This feature was reached using the native CORBA approach: the all Topos modules are described in terms of the CORBA-like interface specification language (ICL).

4. TOPOS OBJECTS

Three classes of CORBA objects are developed - a Worker, a Task and a Manager, as shown in Fig.3.

The "Task" class is the class of objects, which are held by the Data Management Subsystem. The last is the 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 these objects.

The "Worker" class is the class of objects, which provides actual calculations. Workers are spread over number of hosts in the Net. The Manager controls them. The Worker classes of objects are not visible by user. Jobs (in the usual sense) are distributed among a set of Workers in a manner like balance loading.

Each worker returns a minitask completion code by the manager request. If minitask performs an abnormal completion, it is marked as erroneous. In the case manager can not receive the completion code (e.g. due to worker host crash) it assume that minitask should be restarted on the first available Worker. Also,

Manager periodically (once in 20 seconds) checks all workers. If Manager crashes itself, it restore it's state completely from the state repository (DBMS object) and then continue to work with workers (without their restart).

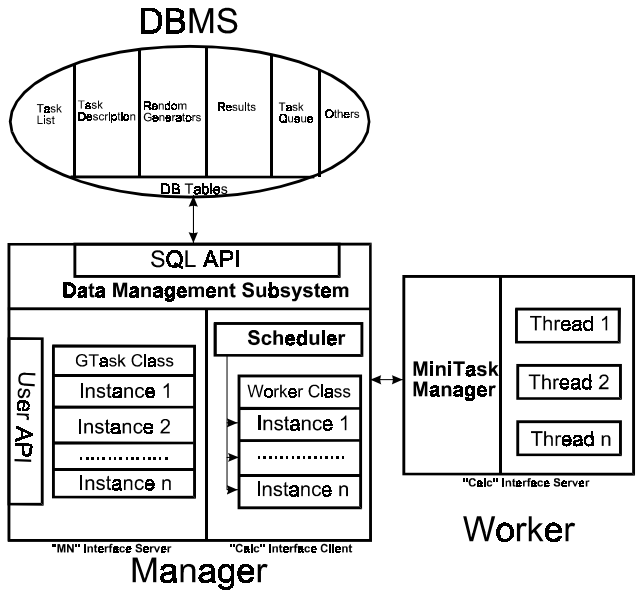

Figure 3. Detailed structure of Topos system.

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 Fig. 3).

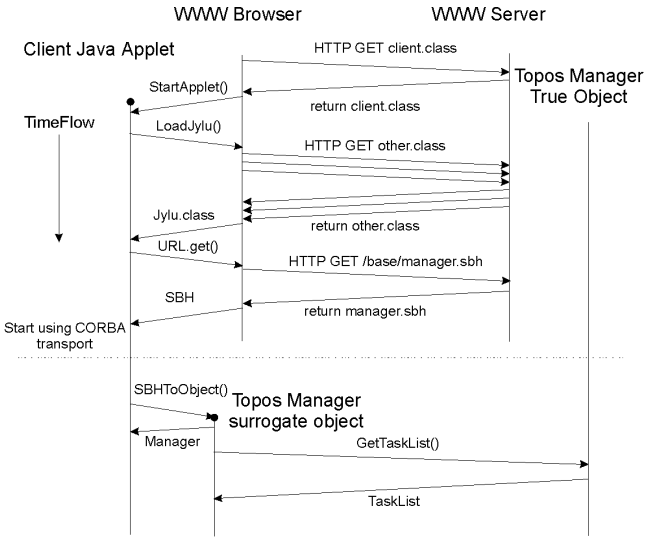

Figure 4. Time flow diagram of GUI applet loading and starting.

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

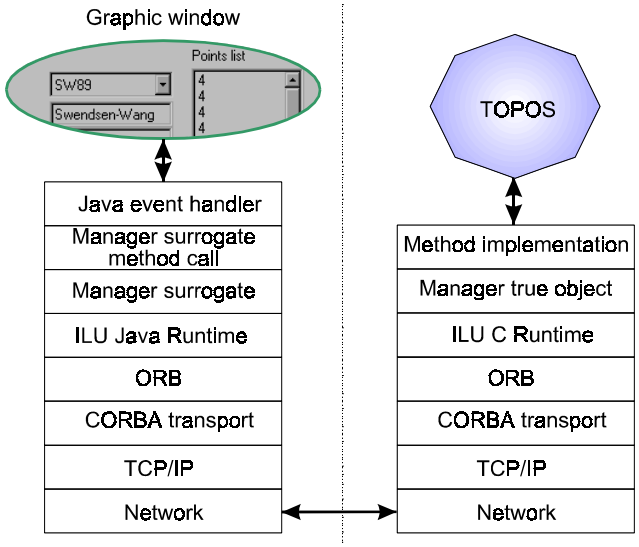

Figure 5. How user interacts with the Topos using GUI.

5. USER INTERFACE

The graphical user interface was made as Java applet. This applet can be loaded via WWW by any Java-capable WWW browser.

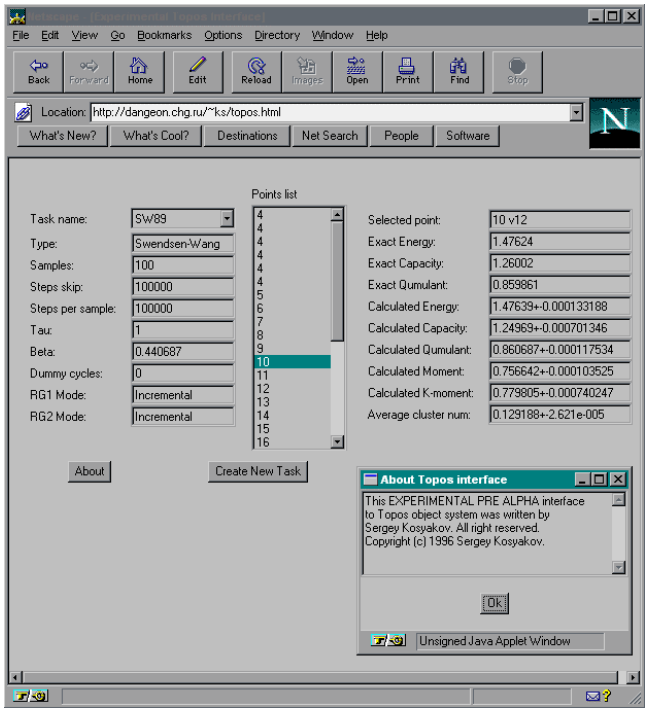

Figure 6. The example of Topos GUI

When started, GUI applet creates Topos Manager object surrogate, and after that moment interacts with Topos via CORBA transport. The process of user interaction with the system Topos is sketched in Fig.5. An example snapshot of user interface is shown in Fig. 6.

6. DEVELOPMENT PLATFORM

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

The Data Management Subsystem based on Postgres95 (now PosgreSQL) DBMS. It is free of charge, with the good functionality and it is the portable one. Thanks to the ILU properties, Topos can run on a wide range of platforms – on MS Windows (3.11, 95, NT) and through 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 one workstation) and a number of FreeBSD boxes, spreaded over the Chg-NET (**Net**work of Science Park in **Ch**erno**g**olovka, Moscow region, Russia).

The user interface is based on the ILU release, written completely using Java - JYLU. It were checked with Netscape Navigator v.3.0 (Netscape Communications) and HotJava (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.x supports VisiBroker for Java (Visigenic) which can work with ILU as well. We hope new versions of Topos will work with Netscape Communicator as well.

7. MATHEMATICAL PHYSICS EXPERIMENT

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

$$exp(-\alpha k^2) < P_L(k) < exp(\alpha` k^2).$$

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 for $k=1,2$ and 3 and analysis of our data gives $P(k>1)=6.58 \cdot 10^{-3}$ with error $\approx 3 \cdot 10\text{-}5$ 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). The experiment was made on square lattices with side $L=4\text{-}128$, each measurement consists of 100 series of 10^6 samples each. It means, that for typical lattice with $L=32$ we should made cluster decomposition for $2 \cdot 10^{11}$ bonds, and generate the same number of random numbers.

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 (Fig.7).

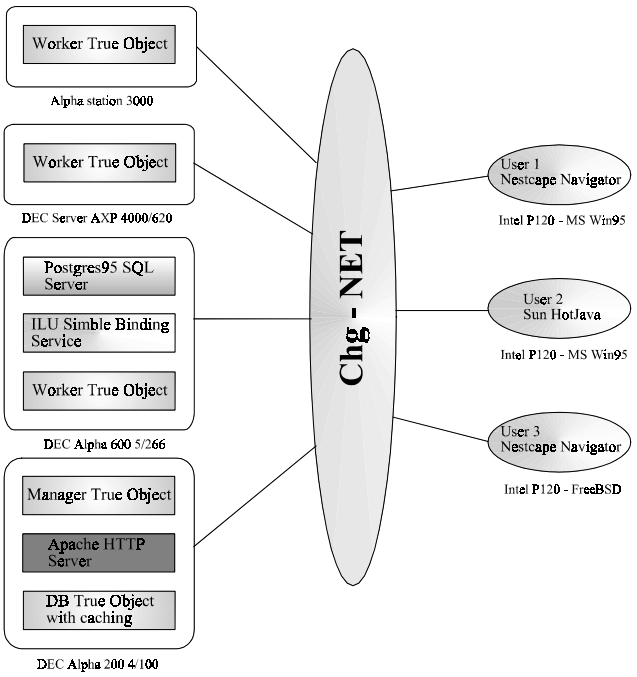

Figure 7. Network configuration for the first computer experiment under Topos management.

8. CONCLUSIONS

Despite the alpha-status of the Topos system we can definitely say that this technique is the right choice for us. The system can be easily tuned for the wide range of problems in large-scale computer experiments. Topos is the really portable system and does not depend on the choice of the specific DBMS, compilers, OS, CPU platform, etc.

Recently, we carry out the computer experiments on measuring of probabilities of Incipient Spanning Clusters in square bond critical percolation. All computations were down using Topos. Our practice confirms completely our vision of Topos system as we claims on above in subsection 'Design goals'. The robustness was successfully confirmed during the large-scale computer experiments.

Our final message is that combinations of the CORBA techniques with Java and WWW are open the new horizons for the development of information resources and have (really) open interfaces.

9. REFERENCES

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

[2] Lev N. Shchur and Sergey S. Kosyakov, Probability of Incipient Spanning Clusters in Critical Square Bond Percolation, Int.J.Mod.Phys.C: **8**, 473-481 (1997)

[3] S.S. Kosyakov, S.A. Krashakov, L.N. Shchur, Distributed Tasks Management System Topos, Proc. Int. Conf.

PDPTA'97, Las Vegas, Nevada, USA, June 30 - July 3, 1997, Vol.2, 1170-1173 (Ed. H. R. Arabnia, 1997)

[4] L.N. Shchur and S.S. Kosyakov, Probability of Incipient Spanning Cluster in Critical Two-Dimensional Percolation, Nuclear Physics B (Proc. Suppl.) 63A C (1998) 664-666

[5] S.S. Kosyakov, S.A. Krashakov, L.N. Shchur, Topos - Manager for Distributed Computing Media. Proc. Int Conf. ADBIS-97, St. Petersburg, 1997

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

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

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

[9] JYLU - an implementation of part of the ILU kernel and a Java run-time purely in Java, http://www-db.stanford.edu/$\sim$hassan/Java/Jylu

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