USE OF HIGH PERFORMANCE COMPUTING TECHNOLOGIES AND SCRIPT RUN MEDIATOR MIDDLEWARE FOR EDUCATIONAL PROCESS IN LIEPAJA UNIVERSITY

Laimonis Zacs, Maksims Zigunovs, Anita Jansone
Liepaja University, Latvia
laimonis.zacs@liepu.lv, maksims.zigunovs@liepu.lv, anita.jansone@liepu.lv

Abstract. In this paper the authors describe a solution for implementation of the SRM (ScriptRunMediatorCSP) middleware to resolve trusting and secure connection problems to HPC (High Performance Computing) cluster resources in the Liepaja University. The main idea is to develop a specific middleware tool called SRM, which would allow connection to the parallel computing cluster system, namely to the main server *masternode*, to provide the easiest solution for using HPC technologies in educational processes. Cluster computing resources could be used for solution and analysis of various scientific problems applying mathematical methods and high performance technological solutions for learning and performing parallel computations in the educational process thus increasing the students' theoretical knowledge in practical applications.

Keywords: cluster, HPC technologies, Lazarus, SRM middleware, parallel calculations, parallel compilers.

Introduction

Rapid development of High Performance Computing (HPC) systems in the field of IT is determined by the necessity to solve society and science problems more effectively, for instance, to optimize gas and oil research and acquisition, to forecast climate changes, and also to introduce innovation products and services in industrial and financial fields. Therefore, it is important to provide the HPC technological environment, which is implemented with the help of parallel data processing and computation technologies, applications of cluster networks and information technologies, in order to promote involvement of students as young experts to real problem solution in the field of scientific research.

In the academic and research field, HPC utilizes multiple processors to perform concurrent tasks in parallel for solving large cutting-edge research computation problems. At this moment authors provide centralized high-performance computing resources and support to Liepaja University students and researchers in large-scale computing disciplines with the use of advanced computing cluster hardware infrastructure, software, tools and programming techniques.The goal of centralized supercomputing centres and conventional distributed computing systems, such as grids and clusters, and their support service is to enhance research capabilities and effectiveness.

The current efforts in the grid computing research focus on a design of the new grid schedulers, that can efficiently optimize the standard scheduling objectives, such as makespan, flowtime and resource utilization, but also can fulfil the security requirements of the grid users and can minimize the energy consumed by all of the system components. These schedulers should also capture the complexity of the whole system and provide meaningful measures for a wide range of grid applications and services. Therefore, energy-efficient and security-aware scheduling in CGs (Computational Grids) becomes a complex research and engineering endeavour mainly due to different priorities and preferences of the grid users and resource owners [1].

Several parallel computation clusters with various configurations have been created and tested, where the special middleware SRM (ScriptRunMediatorCSP (ClusterSecurityPackage)) was used in order to implement solutions of direct, indirect and optimisation problems using data parallel processing technologies – computation automation software. Within the research a Beowulf cluster is used, and the operating system adopted is Linux, which is widely available; software tools for message passing such as MPI [2; 3] and GNU compilers [4] are also a part of the Beowulf cluster. A Beowulf cluster is "A cluster of mass-market commodity off-the-shelf (M^2COTS) PCs interconnected by low cost LAN technology running an open source code Unix-like OS and executing parallel applications programmed with and industry standard message passing model and library" [5]. This is a very attractive proposal for our university, which has usually had limited resources. In this context, we began the construction of a cluster, which then had four nodes, and is continuously supplemented with modern and fast computing nodes. Parallel data processing is fully automated, therefore, a potential user of the software is offered ready command line solutions. At the moment SRM is used within the study course High Performance Computing Technologies and Parallel Programming at Liepaja University, in order to provide more effective, easier and safer access to cluster resources.

The special middleware SRM was created mostly for optimisation of parallel data processing: sending and execution of parallel computation tasks; acquisition of results and management of computation time; and also for students and researchers to perform parallel computations by using Intel Fortran 9 compiler of the experimental cluster, as well as the possibility to launch Intel Cluster Toolkit MPI [6] software and MathWorks MATLAB [7] software packages. The software had some improvements by integrating other compilers and parallel processing tools such as OpenMPI [8] and MPICH [9]. The developed middleware has all necessary documentation, it has been tested and corresponds to solution of all defined problems. The software documentation is created on the basis of the Latvian state standards regulating the software elaboration process documentation at various stages and performance of software quality provision events.

Materials and methods

At the initial stages of creation of the Beowulf cluster network an experimental cluster network was created, which is active at the moment.

1. 8 computational nodes – slavenodes (Intel® Pentium® 4 Duo 3.0Ghz, 2 GB RAM);
2. 1 main server and a node for computational tasks managing – masternode (Intel® Pentium® Xeon 2.8 Ghz, 4 GB ECC RAM).

The installed and configured solutions of data parallel processing technologies are the following.

1. Intel® Cluster Toolkit 3.2.1 - Intel® MPI Library 3.2 Update 1, Intel® Trace Analyser and Trace Collector 7.2, Intel® MPI Benchmarks 3.2;
2. Intel® Fortran Compiler Professional Edition - Intel® Math Kernel Library (Intel® MKL), Intel® Debugger, Intel® Fortran Compiler building applications that run on IA-32, Intel® 64 and IA-64 architecture systems running the Linux operating system;
3. MathWorks MATLAB R2009b software:
   - parallel Computing Toolbox (Distributed Computing Toolbox) is installed on the user workstation, where programming of files for parallel computation is done;
   - MATLAB Distributed Computing Server (MATLAB Distributed Computing Engine) is installed on each computation node in the cluster.

In 1991, the GNU/Linux OS was born. Nowadays, an array of Intel processors running Linux is perhaps the cheapest way to access the parallel computing world with the best speed/performance ratio (arrays of DEC/Alphas running Linux are also a popular choice) [5].

Scientific Linux (SL) is a Fermi National Accelerator Laboratory and the European Organization for Nuclear Research (CERN) Linux distributive SL, which is based on Red Hat Enterprise Linux version. In addition SL OS has a configured service Samba for sharing network resources between the server and slave nodes, SSH network protocol, which allows secure sending of commands from the server and sending of parallel computing commands to all nodes. In addition, SSH has a configured sudo option to execute commands in a security mode, which requires user root permissions by additionally generating SSH keys in order to provide automatic connection without confirmation of root password each time [10].

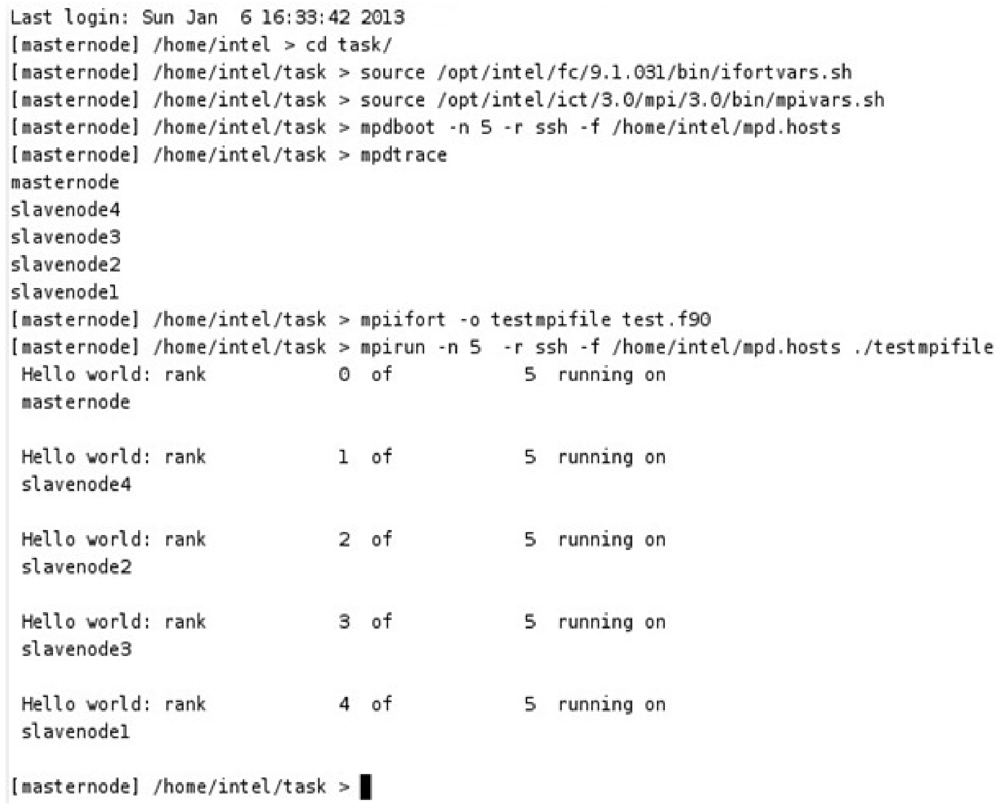
According to De Turck et al.: "Security is an important issue for the platform, especially when using external resources. Two aspects are important: the platform has to provide: (i) a secure communication environment and (ii) a secure execution environment [11].

In order to connect to the cluster, it is necessary to receive cluster access data, which are composed of an IP address, a user name and a password, taking into account the security rules in respect to the obtained information for protection of sensitive data. Until now the most problematic was the SSH connection mode to the cluster nodes, because there always was risk that some users could run some extra scripts in the command line console of the operation system in SSH mode, that would significantly increase the computation resource load or fully stop them for some time. In addition, in some cases problems were caused by a simultaneous connection of multiple users to the cluster with one user name, this caused termination of all parallel computation processes when someone stopped the parallel session. Moreover, there were cases when at the time of using SRM students shared their print screens in social portals thus showing the access information – IP address, login name and other secured information to other people that puts the cluster safety under the risk and is the reason for ineffective use of resources, or overload.

According to De Turck et al.: "A secure execution environment is more difficult to implement. Making the execution environment secure for the users of the workstations can be achieved by certifying agents and code fragments with a PGP signature, so that the authenticity and integrity of the code fragment can be checked. This assures that users only execute code from a trusted source. However, it is next to impossible to make the execution environment safe from the clients' point of view. Preventing users of tampering with the code and/or binaries or to develop trojan horses which act as real agents but return bogus results is very difficult. The platform as described in this paper relies on the goodwill of the users and employs a user registration mechanism as dissuasion technique" [11].

Until now, in the Windows operation system it was necessary to use WinSCP[12] program to get connected to the cluster and upload or download source code files as well as the executed result data files. It was also necessary to use PUTTY [13] software, in order to execute certain commands and parallel computation commands in Linux console, for example (Fig.1):

1. `#source ***/fc/9.1.031/bin/ifortvars.sh`
2. `#source ***/ict/3.0/mpi/3.0/bin/mpivars.sh`
3. `#mpdtrace` (checks an mpd session and shows the names of active computation nodes);
4. `#mpdboot -n 5 -r ssh -f ***/mpd.hosts` (creates a new MPI engine parallel session for 5 computation nodes including the main server or masternode);
5. `#mpiifort -o test test.f90` (compiles the programmed source code file "test.f90", using a compiler Intel Fortran 9, and gives the name "test" to the executed file);
6. `#mpirun -n 5 -r ssh -f ***/mpd.hosts ./test` (starts execution of parallel computations of "test" file on 5 nodes, using MPI software opportunities to start the parallel computation);
7. `#mpdallexit` (closes the MPI engine parallel session.)



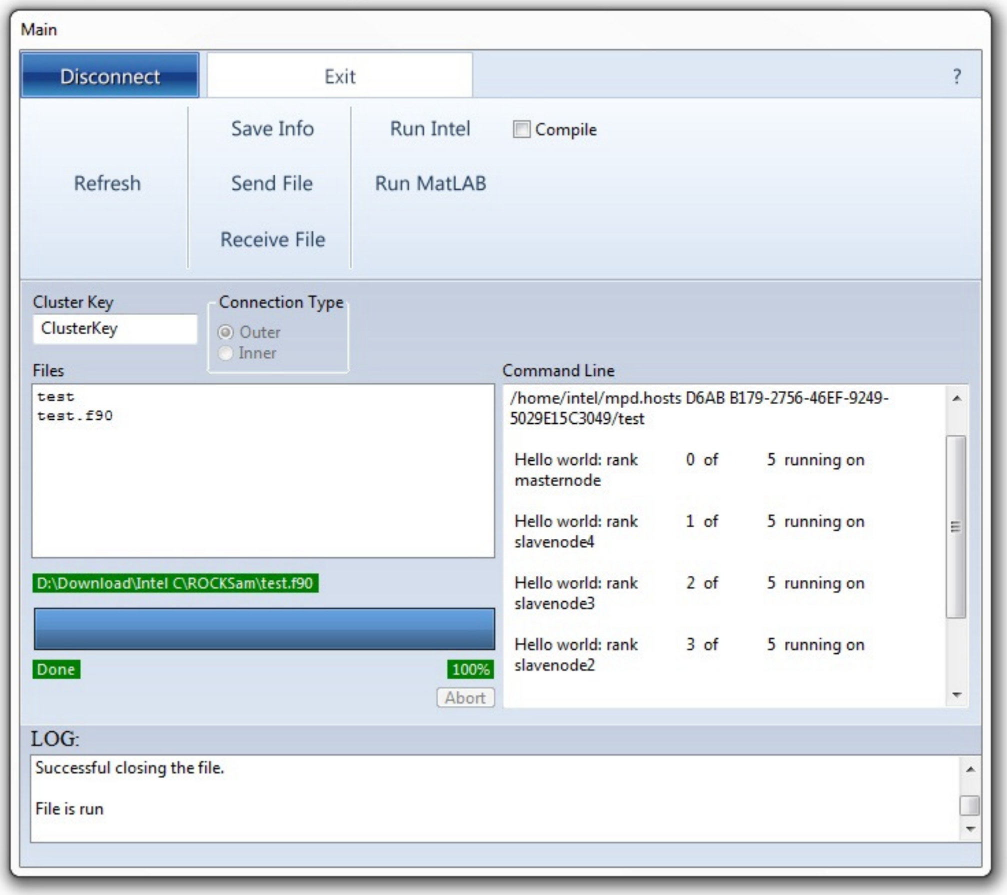
```
Last login: Sun Jan  6 16:33:42 2013
[masternode] /home/intel > cd task/
[masternode] /home/intel/task > source /opt/intel/fc/9.1.031/bin/ifortvars.sh
[masternode] /home/intel/task > source /opt/intel/ict/3.0/mpi/3.0/bin/mpivars.sh
[masternode] /home/intel/task > mpdboot -n 5 -r ssh -f /home/intel/mpd.hosts
[masternode] /home/intel/task > mpdtrace
masternode
slavenode4
slavenode3
slavenode2
slavenode1
[masternode] /home/intel/task > mpiifort -o testmpifile test.f90
[masternode] /home/intel/task > mpirun -n 5  -r ssh -f /home/intel/mpd.hosts ./testmpifile
 Hello world: rank           0  of           5  running on
 masternode

 Hello world: rank           1  of           5  running on
 slavenode4

 Hello world: rank           2  of           5  running on
 slavenode2

 Hello world: rank           3  of           5  running on
 slavenode3

 Hello world: rank           4  of           5  running on
 slavenode1

[masternode] /home/intel/task > █
```

Fig. 1. **Sample interface of PUTTY software for using Beowulf cluster resources**

Results and discussion

The middleware SRM has been created within the study course High Performance Computing at Liepaja University, which makes it possible to use the experimental Beowulf cluster, in order to send the source code files and compile them and run them within the MATLAB, Intel Cluster Toolkit, OpenMPI and MPICH tools. The main computation devices (masternode or frontend) have Intel Cluster mpdboot mode launched as well as MatlabJobManager service. The computation nodes are used to launch certain workers, and the number of such workers depends on the number of cores of each node processor. The software users are mostly the students of Liepaja University and researchers, who use data parallel processing technologies. Opportunities of secure cluster software SRM are the following (Fig. 2).

1. Uploading of files to the experimental cluster.
2. Downloading of files from the experimental cluster to the user's computer.
3. Compilation of a source code file, using an Intel Fortran 9 compiler; use of data parallel processing and computation opportunities for the source code, compiled by an Intel Fortran 9 compiler, using Intel Cluster Toolkit MPI parallel computation software packages.
4. Compilation of a source code file, using an Intel Fortran compiler and GCC (GNU Compiler Collection [4]); use of data parallel processing and computation opportunities for the source code, compiled by an Intel Fortran compiler and GCC, using open source OpenMPI parallel computation software packages.
5. Compilation of a source code file, using an Intel Fortran compiler and GCC; use of data parallel processing and computation opportunities for the source code, compiled by an Intel Fortran compiler and GCC, using open source MPICH parallel computation software packages.
6. Use of data parallel processing and computation opportunities for a source code programmed by using MathWorks MATLAB software.



Cluster Key: ClusterKey

Connection Type: [✓ Outer] Inner

Files:

```
test
test.f90
```

D:\Download\Intel C\ROCKSam\test.f90

Done 100%

Command Line:

```
/home/intel/mpd.hosts D6AB B179-2756-46EF-9249-
5029E15C3049/test

Hello world: rank     0  of      5  running on
masternode

Hello world: rank     1  of      5  running on
slavenode4

Hello world: rank     2  of      5  running on
slavenode3

Hello world: rank     3  of      5  running on
slavenode2
```

LOG:

```
Successful closing the file.

File is run
```

Fig. 2. **Interface of the first version of the middleware ScriptRunMediatorCSP**

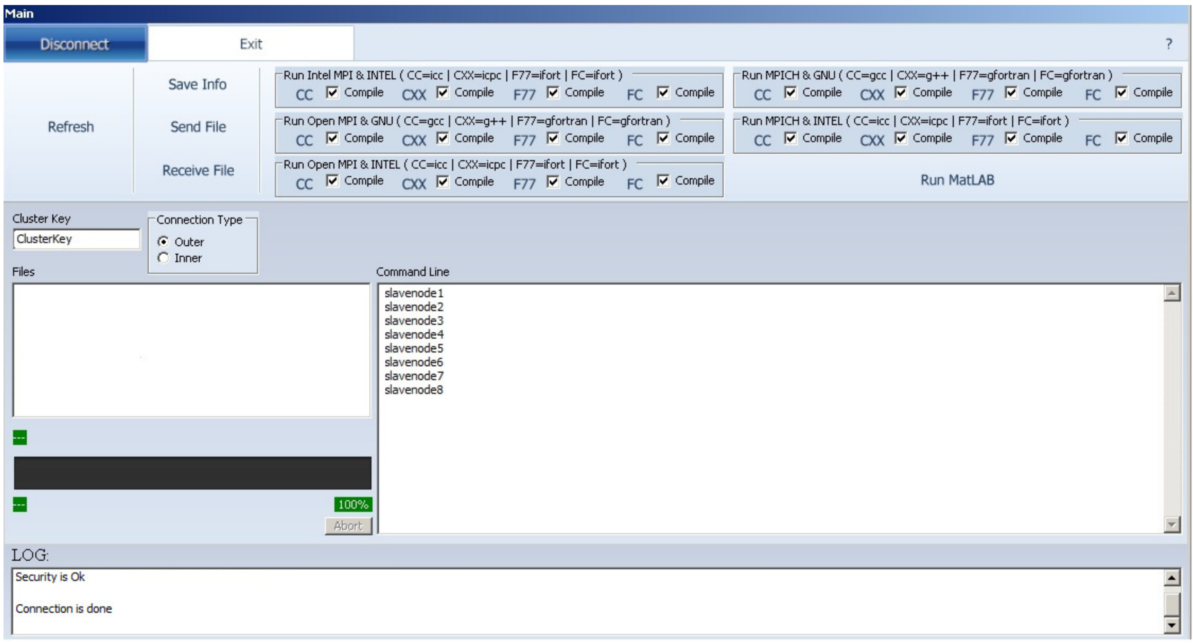
Improved middleware SRM uses several parallel computation tools and a compiler collection. These application functions such as file execution tool (using parallel computation tool) with specific configuration and run and/or compile (using compiler collection) with various compilers are summarized in Table 1.

Table 1

Improved middleware ScriptRunMediatorCSPapplication functions

File execution tool with specific configuration	**Run and/or compile with C compiler** **<u>CC</u>button**	**Run and/or compile with C++ compiler** **<u>CXX</u>button**	**Run and/or compile with F77 compiler** **<u>F77</u>button**	**Run and/or compile with F90 compiler** **<u>FC</u>button**
1. Run Intel MPI	C compile	C++ compile	F77 compile	F90 compile
INTEL configuration	CC=icc	CXX=icpc	F77=ifort	FC=ifort
2. Run Open MPI	C compile	C++ compile	F77 compile	F90 compile
GCC configuration	CC=gcc	CXX=g++	F77=gfortran	FC=gfortran
3. Run Open MPI	C compile	C++ compile	F77 compile	F90 compile
INTEL configuration	CC=icc	CXX=icpc	F77=ifort	FC=ifort
4. Run MPICH	C compile	C++ compile	F77 compile	F90 compile
GCC configuration	CC=gcc	CXX=g++	F77=gfortran	FC=gfortran
5. Run MPICH	C compile	C++ compile	F77 compile	F90 compile
INTEL configuration	CC=icc	CXX=icpc	F77=ifort	FC=ifort
6. Run MATLAB				

The main module of the program looks like in Fig.3. Therefore, the main idea of the program algorithm is to make automation of the user actions. This means that the program makes several commands. SRM is programmed using a free software Lazarus [14].


Main
Disconnect
Exit
?
Refresh
Save Info
Send File
Receive File
Run Intel MPI & INTEL (CC=icc | CXX=icpc | F77=ifort | FC=ifort)
CC ☑ Compile CXX ☑ Compile F77 ☑ Compile FC ☑ Compile
Run Open MPI & GNU (CC=gcc | CXX=g++ | F77=gfortran | FC=gfortran)
CC ☑ Compile CXX ☑ Compile F77 ☑ Compile FC ☑ Compile
Run Open MPI & INTEL (CC=icc | CXX=icpc | F77=ifort | FC=ifort)
CC ☑ Compile CXX ☑ Compile F77 ☑ Compile FC ☑ Compile
Run MPICH & GNU (CC=gcc | CXX=g++ | F77=gfortran | FC=gfortran)
CC ☑ Compile CXX ☑ Compile F77 ☑ Compile FC ☑ Compile
Run MPICH & INTEL (CC=icc | CXX=icpc | F77=ifort | FC=ifort)
CC ☑ Compile CXX ☑ Compile F77 ☑ Compile FC ☑ Compile
Run MatLAB
Files

100%
Abort

Cluster Key: ClusterKey

Connection Type: [✓ Outer] Inner

Command Line

```
slavenode1
slavenode2
slavenode3
slavenode4
slavenode5
slavenode6
slavenode7
slavenode8
```

LOG:

```
Security is Ok

Connection is done
```

Fig. 3. **Interface of the improved middleware ScriptRunMediatorCSP**

The first automated command is connection to the cluster. For connecting to the cluster the user has to press the button “Connect”. In addition, for disconnecting from the cluster the user has to press the button “Disconnect”. For communications between the client program and the cluster the SSH connection technology is used.

For connection between the application and the server, it is modified “TTCPBlockSocket” data class of the “synapse39” package of Lazarus components provided by the Ararat Synapse project. The Ararat Synapse project makes it easier to operate with operating system sockets, therefore it is used for elaboration of this program.

When the program connects itself to the cluster using the preprogrammed cluster user name and a password, it reads the actual cluster access password from the textual file from the cluster user folder. When the program is connected to the cluster, the user needs to send some file to the cluster. It can be made by pressing the button “Send file” and selecting the file. When the file is sent to the cluster, the user needs to run the file by selecting the file in the file list and pressing the file running button on the top of the window. A use case diagram in Fig.4 represents the user’s interaction with the program.

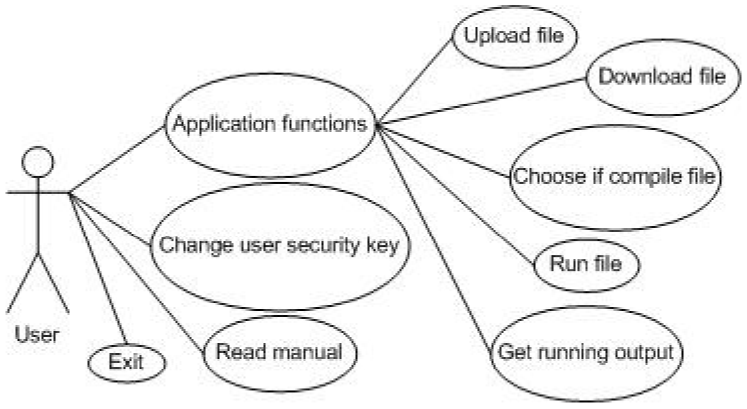


Fig. 4. **Use case diagram**

The application makes several automated actions to compile and run the compiled file represented in Fig. 5 with the sequence diagram.

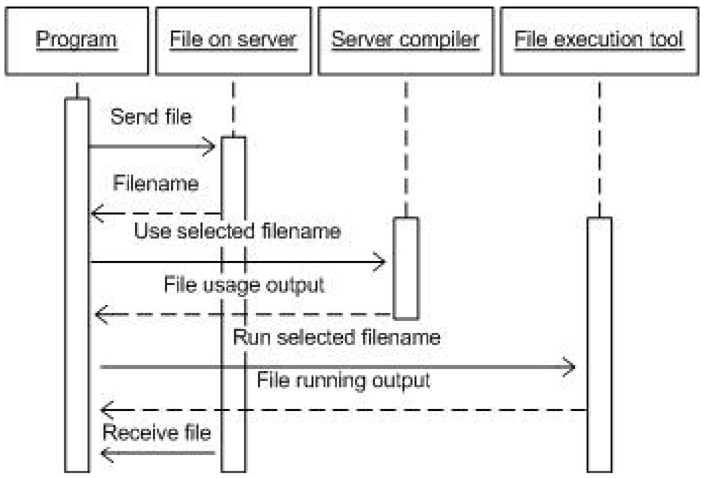


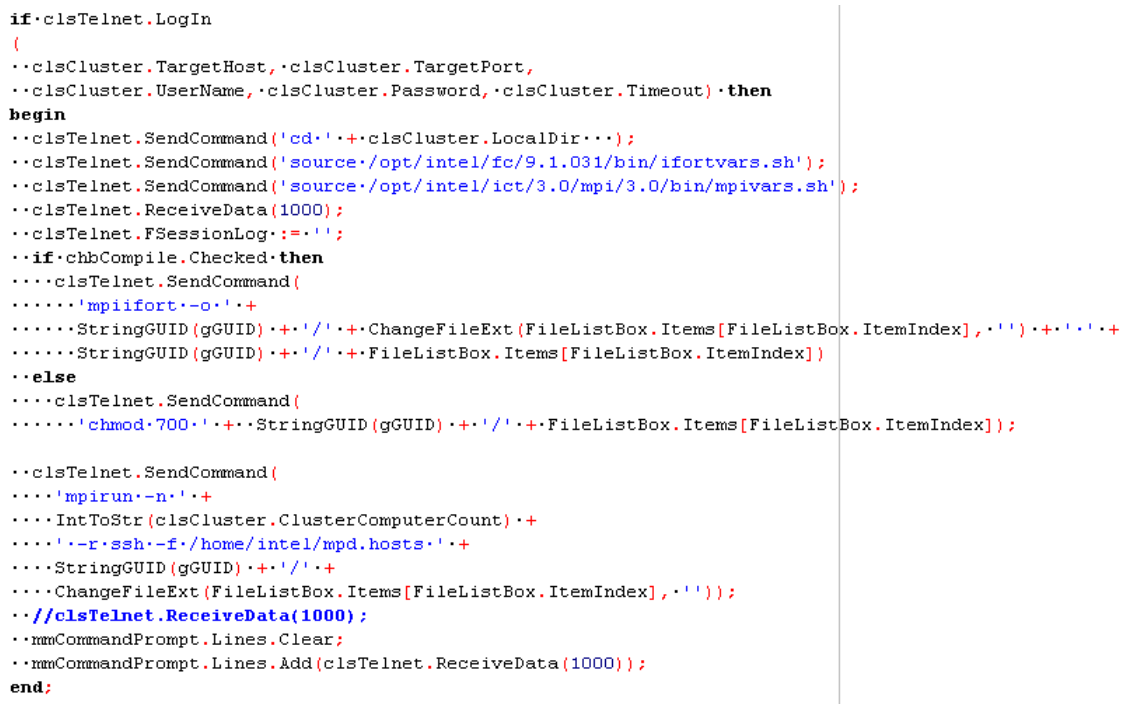
Fig. 5. **Sequence diagram**

Shown in Fig. 6, the part of the source code demonstrates the actions automated for the Intel Fortran compiler to compile and run the Fortran source code.

As it can be seen from the source code, the program executes the same commands like the user has to execute every time he needs to compile and run the code. Moreover, this program makes it possible to have more than one client simultaneously. It creates randomly named folders for every newly connected user without making the program confrontations between the connected users.

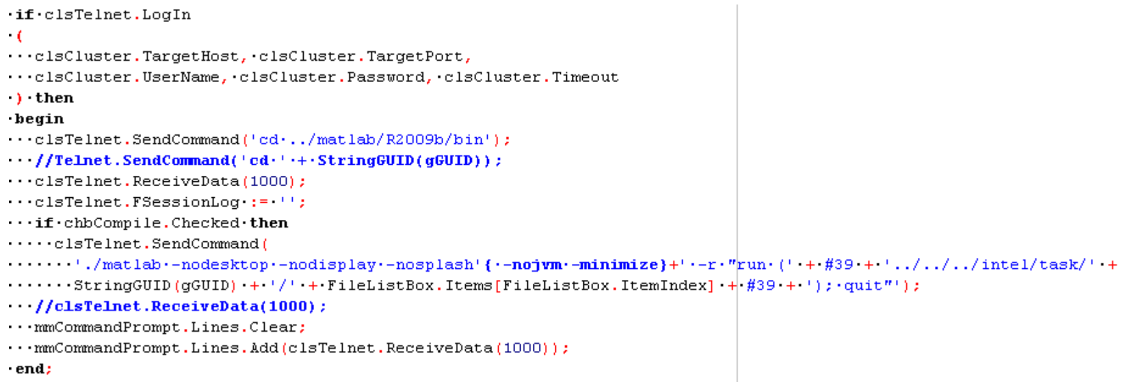
In case of using the MATLAB program within the cluster for making calculations the program executes exactly the MATLAB known commands to run the MATLAB file created by the user and uploaded to the cluster.

The actions made for executing the MATLAB file are shown in Fig. 7 of the source code.



```
if·clsTelnet.LogIn
(
··clsCluster.TargetHost,·clsCluster.TargetPort,
··clsCluster.UserName,·clsCluster.Password,·clsCluster.Timeout)·then
begin
··clsTelnet.SendCommand('cd·'·+·clsCluster.LocalDir···);
··clsTelnet.SendCommand('source·/opt/intel/fc/9.1.031/bin/ifortvars.sh');
··clsTelnet.SendCommand('source·/opt/intel/ict/3.0/mpi/3.0/bin/mpivars.sh');
··clsTelnet.ReceiveData(1000);
··clsTelnet.FSessionLog·:=·'';
··if·chbCompile.Checked·then
····clsTelnet.SendCommand(
······'mpiifort·-o·'·+
······StringGUID(gGUID)·+·'/'·+·ChangeFileExt(FileListBox.Items[FileListBox.ItemIndex],·'')·+·'·'·+
······StringGUID(gGUID)·+·'/'·+·FileListBox.Items[FileListBox.ItemIndex])
··else
····clsTelnet.SendCommand(
······'chmod·700·'·+··StringGUID(gGUID)·+·'/'·+·FileListBox.Items[FileListBox.ItemIndex]);

··clsTelnet.SendCommand(
····'mpirun·-n·'·+
····IntToStr(clsCluster.ClusterComputerCount)·+
····'·-r·ssh·-f·/home/intel/mpd.hosts·'·+
····StringGUID(gGUID)·+·'/'·+
····ChangeFileExt(FileListBox.Items[FileListBox.ItemIndex],·''));
··//clsTelnet.ReceiveData(1000);
··mmCommandPrompt.Lines.Clear;
··mmCommandPrompt.Lines.Add(clsTelnet.ReceiveData(1000));
end;
```

Fig. 6. **ScriptRunMediatorCSPactions automated for the Intel Fortran compiler to compile and run the Fortran source code**



```
·if·clsTelnet.LogIn
·(
···clsCluster.TargetHost,·clsCluster.TargetPort,
···clsCluster.UserName,·clsCluster.Password,·clsCluster.Timeout
·)·then
·begin
···clsTelnet.SendCommand('cd·../matlab/R2009b/bin');
···//Telnet.SendCommand('cd·'·+·StringGUID(gGUID));
···clsTelnet.ReceiveData(1000);
···clsTelnet.FSessionLog·:=·'';
···if·chbCompile.Checked·then
·····clsTelnet.SendCommand(
·······'./matlab·-nodesktop·-nodisplay·-nosplash'{·-nojvm·-minimize}+'·-r·"run·('·+·#39·+·'../../../intel/task/'·+
·······StringGUID(gGUID)·+·'/'·+·FileListBox.Items[FileListBox.ItemIndex]·+·#39·+·');·quit"');
···//clsTelnet.ReceiveData(1000);
···mmCommandPrompt.Lines.Clear;
···mmCommandPrompt.Lines.Add(clsTelnet.ReceiveData(1000));
·end;
```

Fig. 7. **ScriptRunMediatorCSP actions made for executing the MATLAB file**

Conclusions

1. Nowadays, the HPC technological environment is successfully implemented with data parallel processing and computing tools, cluster network and applications of information technologies, with the aim to implement the cluster for parallel computations by preparing the computation resources and developing middleware.
2. The cluster provides an opportunity for students, researchers and others to use the HPC data parallel processing technology and computation tools and also provides involvement of experts into scientific researches for solution of real problems within the frameworks of various projects.
3. SRM provides an easy, safe and controlled student connection to the cluster resources.
4. It is needed to enrol the cluster management platform with dedicate intelligent agent coordinator and load balancing server tool to the SRM program.
5. It is needed to implement NVIDIA GPGPU computing technology & CUDA [15] parallel computing platform opportunities in the cluster in order to implement and integrate it in the SRM program.

Acknowledgements

This text has been elaborated within the framework of the ESF funded project "Development of the Doctoral studies at Liepaja University" (Agreement No. 2009/0127/1DP/1.1.2.1.2./09/IPIA/ SEDA/018).

References

1. Kołodziej J., Khan S.U., Wang L., Kisiel-Dorohinicki M., Madani S.A, Niewiadomska-Szynkiewicz E., Zomaya A.Y., Xu C.Z. Security, energy, and performance-aware resource allocation mechanisms for computational grids. Future Generation Computer Systems. Volume 31, pp. 77-92 (February 2014). Special Section: Advances in Computer Supported Collaboration: Systems and Technologies. Edited by Anna Divoli, DomenicoPotena, Claudia Diamantini and Waleed W. Smari [online] [10.04.2015] Available at: http://www.sciencedirect.com/science/article/pii/S0167739X12001823
2. MPI forum.[online] [10.04.2015]. Available at: http://www.mpi-forum.org/docs/docs.html
3. MPI: A Message-Passing Interface Standard. Version 3.0. Message Passing Interface Forum. [online] [10.04.2015]. Available at: http://www.mpi-forum.org/docs/mpi-3.0/mpi30-report.pdf
4. GCC, the GNU Compiler Collection.[online] [10.04.2015]. Available at: https://gcc.gnu.org/
5. Sterling T.L., Salmon J., Becker D.J., Savarese D.F. How to build a Beowulf. MIT Press, Cambridge, MA, 1999.
6. Intel Corporation home page.Intel Cluster ToolkitCompiler EditionTutorial for Linux. [online] [10.04.2015]. Available at: http://www.physics.udel.edu/~bnikolic/QTTG/shared/docs/ICTCE_Linux_Tutorial.pdf
7. MATLAB doumentation home page MATLAB Tutorial. [online] [10.04.2015]. Available at: http://www.mathworks.se/academia/student_center/tutorials/launchpad.html
8. Open MPI: Open Source High Performance Computing home page. [online] [10.04.2015]. Available at: http://www.open-mpi.org/
9. MPICH High-performance and Portable MPI home page. [online] [10.04.2015]. Available at: http://www.mpich.org/
10. Jansone A., Zacs L., Jakimov K. "An Approach to Information Technologies for Solving Mathematical Physics Problems". Book of Emerging Trends in Computing, Informatics, Systems Sciences, and Engineering, Vol. 151, 2013, SobhTarek, Elleithy Khaled (Eds.), Springer New York, pp. 751-757
11. De Turck F., Vanhastel S., Volckaert B., Demeester P. A generic middleware-based platform for scalable cluster computing. Future Generation Computer Systems. Volume 18, Issue 4, Pages EX1-EX2, 435-572 (March 2002). Best papers from Symp. on Cluster Computing and the Grid (CCGrid2001). Brisbane, Australia. 15-18 May 2001 Available at: http://www.sciencedirect.com/science/article/pii/S0167739X01000784#BIB6
12. WinSCPhome page. WinSCPFree SFTP, SCP and FTP client for Windows. [online] [10.04.2015].Available at: http://winscp.net/eng/index.php
13. PuTTYhome page.[online] [10.04.2015].Available at:http://www.putty.org/
14. Lazarus – The professional Free Pascal RAD IDE.[online] [10.04.2015]. Available at: http://www.lazarus-ide.org/
15. NVIDIA home page, High Performance Computing>What is GPU Computing?[online] [25.04.2015]. Available at: http://www.nvidia.com/object/what-is-gpu-computing.html