Consistent, global state transfer for message-passing parallel algorithms in Grid environments

Summary of Ph.D Theses

Written by

József Kovács
research fellow
MTA SZTAKI

Advisors:
Dr. Péter Kacsuk (MTA SZTAKI)
Dr. László Kovács (ME)

Leader of Doctoral School:
Dr. Tibor Tóth

Computer and Automation Research Institute,
Hungarian Academy of Sciences
(MTA SZTAKI)

„Hatvany József" Doctoral School of Information Science,
University of Miskolc
(ME)

Budapest, 2008

Members of critics committee

Chairman:

Dr. Péter Arató, Correspondent Member of HAS (BME-VIK)

Secretary:

Dr. László Czap, PhD (ME GÉK)

Members:

Dr. Péter Stefán, PhD (NIIF Institute)

Dr. Zoltán Juhász, PhD (Pannon University, Veszprém)

Dr. Ferenc Erdélyi, CSc (ME, retired associated professor)

Dr. Dezső Sima, DSc (BMF NJIK)

Official critics:

Dr. Szilveszter Kovács, PhD (ME GÉK)

Dr. Imre Szeberényi, PhD (BME-VIK)

I. Introduction

Science produces challenges requiring huge computational capacity. To perform the required computation a single, but powerful or even a supercomputer is simply not enough. In order to deliver more performance, more and more computers must be connected and their computational power must be aggregated and utilised. This is exactly the goal of a novel research area called Grid [A]. The goal of the Grid is to provide a dynamic collection of resources from which applications requiring very large computational power can select and use actually available resources. Building blocks of a Grid (i.e. resources) can be individual or cooperating (i.e. cluster) computers or even any hardware components that can provide some services. In the current context the ClusterGrid name is used for those Grid infrastructures, where clusters are used as basic building blocks for the Grid.

In order to utilise the huge computational power a Grid may deliver, distributed applications containing hundreds or thousands of parallel tasks are developed. While tasks are executed in parallel, they may exchange partial results typically via some form of communication. The most effective environment for executing a parallel, communication based application is the cluster (or ClusterGrid) where the high-speed connection among the nodes of the cluster enables the fast progress of the tasks performing intensive communication. A cluster typically contains a scheduler (e.g. Condor [B][C]) to perform resource utilisation and a communication library (e.g. PVM [D][E] or MPI [F][G]) to provide a communication service for the application.

In a Grid environment the applications or jobs claiming substantial amount of resources are assigned to the various resources or clusters by a so-called Grid broker. In case a running application cannot continue its execution due to some reasons (e.g. resource maintenance, overload or failure), the running application must be moved from one resource to another (migration). Usual applications start their execution from the beginning on the newly selected resource which may cause the loss of a huge amount of processing capacity. In order to avoid the loss, there are special techniques called rollback-recovery [H], which can help the application to continue its execution from the point where it was interrupted.

In the literature there are numerous methods and techniques for saving and restoring the internal state of a parallel and distributed application using communication libraries, and the most wide-spread technique is called checkpointing [I]. In the field of checkpointing a number of techniques and methods have been elaborated as a result of the numerous international research and development projects. The numerous techniques are developed for various needs and environments and they can be classified by various aspects like level of implementation, type of coordination, etc.

There is an unquestionable advantage of using checkpointing techniques in case of long-running parallel applications [J], however these techniques have not been widely used in Grid environments due to some reasons. A real obstacle in using the available tools is that they usually rely on some services or some middleware capabilities, which cannot be expected to exist on every cluster of the Grid. Since in a ClusterGrid each cluster might have different software, there is a need for a checkpointing technique that does not depend on any service (transparency).

During my research I have actively used and worked with the P-GRADE parallel program development environment developed by MTA SZTAKI. The P-GRADE

[1][K] integrated graphical environment gives support for developing message-passing applications, as well as executing them in various execution environments like clusters, supercomputers or Grid. P-GRADE makes reengineering of sequential programs much faster and easier based on a hierarchical, graphical design tool, a hybrid graphical net language called GRAPNEL [L], an integrated debugging tool [2], an on-line monitor and performance visualisation tool called PROVE [1].

The P-GRADE environment has been widely used in universities for teaching the topic of parallel algorithms and systems. In addition, the introduction of the prototype on the software market is one of the goals for a currently establishing English Hungarian company.

The work introduced in this thesis attempts to overstep the aforementioned limitations by introducing a transparent checkpointing and migration technique, by defining an abstract model to give its background theory and by showing concrete applications.

II. Applied methods

During the research my first goal was to elaborate checkpointing methods for ClusterGrid environment. To do that as a first step I have defined the basic operational and architectural requirements of ClusterGrid based on evaluating the related literature and based on the analysis of some Grid systems. After the analysis of Grid infrastructures, I have explored the literature to overview the existing checkpointing techniques, methods and tools [H][I][J], then I have done similar with the most relevant tools in the area [M][N][O][P][Q][R][S]. I have analysed and classified the tools from different aspects [3].

After identifying the requirements towards the checkpointing techniques, my second goal was to create a formal model framework. To create the model I have chosen the method called ASM [T][U][V] (Abstract State Machine). The ASM is able to express and handle the notion of the model maker on an arbitrary abstract level. Therefore, the models I have elaborated are expressed as ASM notion and syntax. The models later have been refined based on the model refinement method of ASM and the correctness of model refinements have also been proven using ASM.

The ASM formal framework has been widely accepted and applied in industry and scientific projects to design and analyse different systems. It has been successfully applied for verifying the correctness of Prolog [W], Occam [X] compilers and of the Java Virtual Machine [Y] execution mechanism. Even Microsoft has used it for designing and analysing its software systems [Z].

After elaborating a checkpointing procedure, I have analysed the P-GRADE environment and adapted my solution for it. This work has been carried out using the literature and getting information on the internal operation. I have followed similar way when designed and developed the TotalCheckpoint [4] checkpointing tool. I have also studied the well-known message-passing system called PVM. To create the architectural and operational plan of the checkpointing tool, the aforementioned model creation principles of the ASM framework was followed.

To introduce the new checkpointing and migration procedure in real systems, the Condor [AA] job scheduler was used. Condor is developed in the Computer Science department of University of Wisconsin (Madison, USA). The impact of Condor is very high since approximately 100 thousands of computers belonging to more than 800 clusters are supervised in plenty of institutes all over the world by nowadays.

In order to study the internal operation and architecture of Condor I got direct help from the Condor developers in the framework of cooperation in a common project and several mutual visits. Finally, I elaborated a process and application migration procedure after I got a detailed view of the Condor system.

III. New scientific results

1 New checkpointing method in ClusterGrid environment

Checkpointing of message-passing based parallel algorithms or applications can be realised using various techniques and methods. A concrete solution must always face the requirements imposed by the Grid middleware. The goal of the first group of thesis is to identify the requirements and conditions towards the checkpoint and migration techniques imposed by the ClusterGrid environment and to elaborate a new (abstract) technique which fulfils the already identified requirements.

As a preparation of thesis 1.1, I have defined the main characteristics of the ClusterGrid environments and several desirable use cases for state migration mechanism. As a next step, I have identified the cluster components which may influence the internal operation of a checkpointing system and then I determined the requirements in 4+1 points. By using the ASM formal framework for modelling I have developed the model of the cluster components and their most relevant characteristics, and then based on this model I have elaborated a formal description of the requirements and conditions for checkpointing techniques. Finally, based on these criteria I have analysed, evaluated and classified the existing solutions and stated thesis 1.1.

Thesis 1.1: *In a cluster environment a formal framework of requirements can be defined for message-passing parallel algorithms, which enables transparent checkpointing of the algorithms for the scenarios defined in the dissertation. In addition, currently there are no checkpointing and migration facilities fulfilling the defined requirements in a cluster environment for the defined scenarios.*

Related chapter in dissertation: 2.1

Related publications: [3][4][17]

The list of requirements introduced in thesis 1.1 defines several criteria for transparent checkpointing. The various solutions proposed by the different checkpointing techniques form a design space. In order to designate my proposed solution I have defined the operational and architectural details of the desired transparent solution. As a result, the definition of the proposed solution is summarised by a seven point list and the definition has been suited into the ASM formal model (CP_{ground}) introduced previously in thesis 1.1. In addition, I have elaborated the necessary ASM rules to describe the internal operation mechanism. Finally, I have proved thesis 1.2 through introduction of abstract events.

Thesis 1.2: *The newly elaborated checkpointing method defined by the ASM model called CP_{ground} implements transparent checkpointing both for the programmer and for the middleware at the same time.*

Related chapter in dissertation: 2.2

Related publications: [3][4][5][17]

The solution defined by the CP_{ground} abstract model enables the saving and restoration of the consistent, global state of a message-passing based parallel application or algorithm in a transparent way.

2 Application of checkpointing method for message-passing parallel algorithms

The theoretical background introduced in the first group of theses – which resulted in a checkpointing solution defined by an abstract model – forms an appropriate basis for developing a concrete tool. To utilise the theoretical results I have chosen the P-GRADE graphical parallel program development environment developed by MTA SZTAKI. The goal of the second group of theses is to apply the abstract model for message-passing PVM applications or algorithms created by the P-GRADE environment.

As a preparation of thesis 2.1 I have designed and elaborated a checkpointing technique for PVM applications created by the P-GRADE development environment based on the abstract method defined in thesis 1.2. I have studied the architectural design of the GRAPNEL application. I have defined the required modifications on the architecture and redesigned the communication primitives in a way that the checkpointing operation can be activated at any time during the execution. In addition I have elaborated an abstract model ($CP_{grapnel}$) that fits to the solution introduced in P-GRADE. Finally, I have proven the correctness of refinement between the CP_{ground} and $CP_{grapnel}$ models. Based on the results thesis 2.1 is stated.

Thesis 2.1: *The checkpointing technique integrated in GRAPNEL applications – following a static process model and developed by P-GRADE – realises transparent checkpointing and its corresponding $CP_{grapnel}$ ASM model is a correct refinement of the original model called CP_{ground}.*

Related chapter in dissertation: 3.1

Related publications: [1][3][13][15][16]

The aforementioned solution enables transparent checkpointing operation – both for the programmer and for the middleware – for parallel applications which follow a static process model and developed by the P-GRADE environment.

As a preparation of thesis 2.2 I have designed and elaborated a checkpointing technique for native PVM applications based on the abstract checkpointing method defined in thesis 1.2 and as a generalisation of the method developed in thesis 2.1. I have studied PVM applications and services. Afterwards, I have defined the structure of the PVM application and solved the problem of interruption for communication primitives. I have elaborated an abstract model (CP_{tckpt}) that fits to the introduced solution. Finally, I have elaborated a model refinement procedure to prove that the model CP_{tckpt} – implemented by the TotalCheckpoint (TCKPT) tool – is a correct refinement of the $CP_{grapnel}$ model. Based on the results thesis 2.2 has been stated.

Thesis 2.2: *The checkpointing technique for native PVM applications – following a dynamic process model and realised by the TotalCheckpoint tool – performs transparent checkpointing and its corresponding CP_{tckpt} model is a correct refinement of the $CP_{grapnel}$ model and of the CP_{ground} model.*

Related chapter in dissertation: 3.2

Related publications: [4][5][8][9]

The solution introduced in thesis 2.2 gives transparent checkpointing operation – both for the programmer and for the middleware – for native PVM applications which follow static or dynamic process model.

3 State migration for message-passing parallel algorithms

After the theoretical background elaborated in the first group of theses and the concrete checkpointing solution developed in the second group of theses, the goal of the third group of theses is to justify that transparent migration – based on the previous solutions – can be realised in ClusterGrid environment. There are two options: migration of certain processes of the application among the resources of the hosting cluster and migration of the entire application among clusters. To demonstrate the elaborated techniques the well-known job scheduler called Condor has been selected.

Thesis 3.1 focuses on transparent process migration by analysing the interaction between the GRAPNEL application – integrating the transparent checkpointing facility developed in thesis 2.1 – and Condor maintaining the resources of a cluster. In this thesis after short overview of Condor, I define the basic conditions of operation, the components and I determine the steps of migration mechanism. The theory of this mechanism is justified by the analysis of state-transition diagram derived from the $CP_{grapnel}$ model and by mapping the flow of migration procedure into the state-transition diagram. Based on the introduced results I have stated thesis 3.1.

> **Thesis 3.1:** *The migration of processes of the GRAPNEL applications – developed in P-GRADE – is realised in a transparent way for the schedulers and the elaborated solution adapts itself to the internal rules of the $CP_{grapnel}$ ASM model.*
>
> Related chapter in dissertation: 4.1
>
> Related publications: [10][11][12][18]

In the introduced solution transparency is ensured from several aspects, since the migration does not require any modification in the (Condor) scheduler, nor in PVM, nor in any components of the operating system and nor in the source code of the application.

In the migration solution presented in thesis 3.1 the procedure is driven by a special coordinator process – integrated in the GRAPNEL application – where the termination of this process causes the entire application to be shutdown. However, this coordination process must be terminated during the application migration since link between the clusters in a ClusterGrid are usually not exists. To eliminate this problem I elaborated an application migration solution and then I mapped it to the $CP_{grapnel}$ model. Based on the results thesis 3.2 is stated.

> **Thesis 3.2:** *The GRAPNEL application implements a consistent, global state-space migration for message-passing parallel applications by saving and restoring the integrated coordination process. It enables the migration of the entire application among independent (not using each others' resources) clusters. In addition, the elaborated solution adapts itself to the internal rules of the $CP_{grapnel}$ ASM model.*
>
> Related chapter in dissertation: 4.2
>
> Related publications: [1][10][13][14]

IV. Application of new scientific results

Application of checkpointing techniques is more and more important nowadays, since fault-tolerant execution of a long-running application is indispensable. The rate of failures causing resource shutdown is not negligible in a Grid containing huge amount of resources. In case of long-running (weeks or months) applications there is a substantial likelihood of at least one resource failure which causes the execution to be interrupted [BB]. A resource failure may cause the already computed (sub)results to be perished. The avoid loosing computational capacity [J] provides a huge economical and technical advantage for the Grid. The results introduced in the theses can be utilised in any ClusterGrid environment, since they are independent from the actual infrastructure.

The P-GRADE programming environment has been introduced in numerous European universities and institutes for the purpose of teaching. In addition, several scientific projects were supported or inspired by the tool in the topic of parallel programming. P-GRADE was successfully applied in meteorology for parallelising an ultra-short range weather forecast application (Hungarian Meteorology Service) [CC], in scientific engineering area for urban traffic simulation (University of Westminster) and in chemistry for modelling reaction-diffuse systems (Eotvos Lorand University) [DD]. Since the checkpointing and migration tool is integrated into P-GRADE, all the applications developed by P-GRADE automatically completed with checkpointing and migration capabilities.

The results introduced in the theses have been published in numerous journals, conference and workshop papers and have been presented in various scientific exhibitions and forums. The aforementioned publications have inspired further research and the work forms the basis of some national and international activities. A significant example among the national activities is the one with the National Information Infrastructure Development Institute (NIIF) operating the Hungarian ClusterGrid infrastructure [EE] (~30 institutes, ~1800 machine, ~400GFlops performance). In this cooperation the TotalCheckpoint tool has been developed for NIIF. One of the most significant international activities is in the frame of the CoreGRID EU Network of Excellence project where the introduced results are about to be integrated into a Grid Checkpointing Architecture (GCA) [6][7][19][20][21] by the partners.

V. Publications

Journal papers

[1] P. Kacsuk, G. Dózsa, **J. Kovács**, R. Lovas, N. Podhorszki, Z. Balaton and G. Gombás: “*P-GRADE: a Grid Programming Environment*”, Journal of Grid Computing Vol. 1. No. 2, pp. 171-197. 2004.

[2] **Kovács, J.**, Kacsuk, P.: “*The DIWIDE Distributed Debugger*”, Quality of Parallel and Distributed Programs and Systems, special issue of Journal of Parallel and Distributed Computing Practices, PDCP Vol.4, No. 4, Eds: P.Kacsuk, G.Kotsis, pp. 331-347, 2001

[3] **J. Kovacs**: "*Transparent Parallel Checkpointing and Migration in Clusters and ClusterGrids*", International Journal of Computational Science and Engineering, IJCSE, 2006, (to appear)

[4] **József Kovács**, Peter Kacsuk, Radoslaw Januszewski, Gracjan Jankowski: "*Application and Middleware Transparent Checkpointing with TCKPT on ClusterGrid*", Future Generation Computer Systems, selected papers of DAPSYS2006, (accepted)

Conference/workshop papers

[5] **J. Kovacs**, R. Mikolajczak, R. Januszewski, G. Jankowski: "*Application and middleware transparent checkpointing with TCKPT on Clustergrid*", Proceedings of 6th Austrian-Hungarian Workshop on Distributed And Parallel Systems, DAPSYS 2006, Innsbruck, Austria, September 21-23, 2006, pp. 179-189.

[6] G. Jankowski, **J. Kovacs**, R. Mikolajczak, R. Januszewski, N. Meyer: "*Towards Checkpointing Grid Architecture*", Parallel Processing and Applied Mathematics – Conference on Parallel Processing and Applied Mathematics, PPAM2005, Poznan, Poland, Lecture Notes in Computer Science, Vol. 3911/2006, pp. 659-666, Springer, 2006, ISBN 978-3-540-34141-3

[7] G. Jankowski, R. Januszewski, **J. Kovacs**, N. Meyer, R. Mikolajczak: "*Grid Checkpointing Architecture - a revised proposal*", Proc. of the 1st CoreGRID Integration Workshop, pp. 287-296, Pisa, 28-30, November, 2005

[8] **Kovács József**, Farkas Zoltán, Marosi Attila: "*Ellenőrzőpont támogatás PVM alkalmazások számára a magyar ClusterGriden*", Networkshop, Szeged, 2005

[9] **Jozsef Kovacs**: "*Making PVM applications checkpointable for the Grid*" Proc. of the Microcad 2005 Conference, Section N, Miskolc, 2005, pp. 223-228

[10] **József Kovács**: „*Process Migration in Clusters and Cluster Grids*", Distributed and Parallel Systems: Cluster and Grid Computing, Kluwer International Series in engineering and Computer Science, Vol. 777, Dapsys 2004, Budapest, Hungary, pp. 103-110.

[11] **József Kovács**, Péter Kacsuk: "*A migration framework for executing parallel programs in the Grid*", In: Grid Computing – Second European AcrossGrids Conference, AxGrids 2004, Nicosia, Cyprus, Lecture Notes in Computer Science, Vol. 3165, pp. 80-89, Springer-Verlag, 2004

[12] **József Kovács**, Péter Kacsuk: "*Improving fault-tolerant execution for parallel applications under Condo*r", microCAD International Scientific Conference, University of Miskolc, Miskolc, Hungary, March 18-19, 2004, pp. 251-256

[13] **József Kovács**, Péter Kacsuk: "*Párhuzamos programok vándorlása a Grid-en*", University of Miskolc, Doktoranduszok fóruma, Gépészmérnöki kar szekciókiadványa, 2003, pp. 158-164

[14] P. Kacsuk, R. Lovas, **J. Kovács**, G. Dózsa, N. Podhorszki: "*Metacomputing support by P-GRADE*", GGF8 Workshop on Grid Applications and Programming Tools, 2003

[15] **Jozsef Kovacs**, Peter Kacsuk: "*Server based migration of parallel applications*", 4th DAPSYS Conference, Linz, Austria, 29th September-2nd October 2002, pp: 30-37

[16] **József Kovács**: "*Párhuzamos programok checkpointolása és migrációja klasztereken*", Networkshop'2002, Eger, Eszterházy Károly Főiskola, 26th-28th March 2002

Conference talks and posters

[17] **Jozsef Kovacs**: "*Formal analysis of existing checkpointing systems and introduction of a novel approach*", CSCS 2006, Szeged, Hungary, June 2006 (honored with Best Talk Award)

[18] **Jozsef Kovacs**: "*PVM & Condor checkpointing*", Condor Week 2004, April 14-16, 2004, University of Wisconsin, Madison

Technical reports

[19] G. Jankowski, R. Januszewski, R. Mikolajczak, **J. Kovacs**: "*Scalable multilevel checkpointing for distributed applications - on the integration possibility of TCKPT and psncLibCkpt*", CoreGRID Technical Report, TR-0019, March 2006

[20] G. Jankowski, R. Januszewski, R. Mikolajczak, **J. Kovacs**: "*Scalable multilevel checkpointing for distributed applications - on the possibility of integrating Total Checkpoint and AltixC/R*", CoreGRID Technical Report, TR-0035, May 2006

[21] G. Jankowski, R. Januszewski, R. Mikolajczak, **J. Kovacs**: "*Grid Checkpointing Architecture - a revised proposal*", CoreGRID Technical Report, TR-0036, May 2006

VI. References

[A] I. Foster, C. Kesselman, S. Tuecke, “The Anatomy of the Grid. Enabling Scalable Virtual Organizations”, International Journal of Supercomputer Applications, 15(3), 2001

[B] D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid", in Fran Berman, Anthony J.G. Hey, Geoffrey Fox, editors, Grid Computing: Making The Global Infrastructure a Reality, John Wiley, 2003

[C] Condor homepage: http://www.cs.wisc.edu/condor

[D] V. S. Sunderam: ”PVM: A Framework for Parallel Distributed Computing”, Concurrency: Practice and Experience, 2, 4, pp.:315-339, December, 1990.

[E] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, B. Mancheck and V. Sunderam. PVM: Parallel Virtual Machine a User’s Guide and Tutorial for Networked Parallel Computing, MIT Press, Cambridge, MA, 1994.

[F] William Gropp, Ewing Lusk, and Anthony Skjellum: ”Using MPI”, 2nd Edition, MIT Press, ISBN 0-262-57132-3

[G] Al Geist, Ewing Lusk, William Gropp, William Saphir, Steve Huss-Lederman, Tony Skjellum, Andrew Lumsdaine, and Marc Snir.: ”MPI-2: Extending the Message-Passing Interface”, In EuroPar96, February 1996

[H] Elnozahy, E. N., Alvisi, L., Wang, Y., and Johnson, D. B. 2002. A survey of rollback-recovery protocols in message-passing systems. ACM Comput. Surv. 34, 3 (Sep. 2002), 375-408. DOI= http://doi.acm.org/10.1145/568522.568525

[I] S.Kalaiselvi and V.Rajaraman, A Survey of Checkpointing Algorithms for Parallel and Distributed Computers, Sadhana, Vol.25, Part 5, October 2000, pp.489-510

[J] M. Treaster. A Survey of Fault-Tolerance and Fault-Recovery Techniques in Parallel Systems. ACM Computing Research Repository (CoRR), (cs.DC/ 0501002), January 2005. http://citeseer.ist.psu.edu/treaster05survey.html

[K] P-GRADE Parallel Grid Application Development and Execution Environment, http://www.lpds.sztaki.hu/pgrade

[L] D. Drótos, G. Dózsa, and P. Kacsuk, “GRAPNEL to C Translation in the GRADE Environment”, Parallel Program Development for Cluster Comp. Methodology, Tools and Integrated Environments, Nova Science Publishers, Inc. pp. 249-263, 2001

[M] George Stellner, ”Consistent Checkpoints of PVM Applications”, In Proc. 1st Euro. PVM Users Group Meeting, 1994

[N] J. Leon, A. L. Fisher, and P. Steenkiste, ”Fail-safe PVM: a portable package for distributed programming with transparent recovery”. CMU-CS-93-124. February, 1993

[O] Iskra, K. A., van der Linden, F., Hendrikse, Z. W., Overeinder, B. J., van Albada, G. D., and Sloot, P. M. 2000. The implementation of dynamite: an environment for migrating PVM tasks. SIGOPS Oper. Syst. Rev. 34, 3 (Jul. 2000), 40-55. DOI= http://doi.acm.org/10.1145/506117.506123

[P] J. Casas, D. Clark, R. Konuru, S. Otto, R. Prouty, and J. Walpole, ”MPVM: A Migration Transparent Version of PVM”, Technical Report CSE-95-002, 1, 1995

[Q] C.P.Tan, W.F. Wong, and C.K. Yuen, ”tmPVM - Task Migratable PVM”, In Proceedings of the 2nd Merged Symposium IPPS/SPDP, pp. 196-202, 1999.

[R] Pawel Czarnul: ”Programming, Tuning and Automatic Parallelization of Irregular Divide-and-Conquer Applications in DAMPVM/DAC” in International Journal of High Performance Computing Applications, 2003, Vol.17, No.1

[S] Dan Pei, Wang Dongsheng, Zhang Youhui, Shen Meiming, ”Quasi-asynchronous Migration: A Novel Migration Protocol for PVM Tasks.” ACM SIGOPS Operating Systems Review, 33(2): 5-15 (April 1999).

[T] E. Börger, “High level system design and analysis using abstract state machines”, ASM Workshop, Magdeburg, September 1998

[U] E. Börger, “Why use evolving algebras for hardware and software engineering?”, SOFSEM’95, LNCS, 1995

[V] Y.Gurevich, “Evolving algebras 1993: Lipari guide”, E.Börger, editor, Specification and Validation Methods, pages 9-36, Oxford University Press, 1995

[W] E. Börger and D. Rosenzweig, “The WAM – definition and compiler correctness”, Logic Programming: Formal Methods and Practical Applications, 1994

[X] E. Börger and I. Durdanovic, “Correctness of compiling occam to transputer code”, Computer Journal, 39(1):52-92, 1996

[Y] E. Börger and W. Schulte, “Programmer friendly modular definition of the semantics of java”, Formal Syntax and Semantics of Java, LNCS, Springer, 1998

[Z] Mike Barnett, Egon Boerger, Yuri Gurevich, Wolfram Schulte and Margus Veanes, “Using Abstract State Machines at Microsoft: A Case Study”, Proceedings of ASM'2000 in "Abstract State Machines: Theory and Applications" Eds. Y. Gurevich, P. Kutter, M. Odersky, and L. Thiele Springer Lecture Notes in Computer Science vol. 1912, pages 367--379, 2000

[AA] D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid", in Fran Berman, Anthony J.G. Hey, Geoffrey Fox, editors, Grid Computing: Making The Global Infrastructure a Reality, John Wiley, 2003

[BB] Ralston A, Reily E D 1993 Encyclopedia of computer science 3rd edn (New York: IEEE Press)

[CC] R. Lovas, P. Kacsuk, A. Horvath, A. Horanyi: Application of P-GRADE Development Environment in Meteorology, In: Distributed and Parallel Systems, Cluster and Grid Computing, pp. 109-116, Kluwer Academic Publishers, 2002

[DD] R. Lovas, P. Kacsuk, I. Lagzi, T. Turányi: „Unified development solution for cluster and grid computing and its application in chemistry”, In: Computational Science and Its Applications – ICCSA 2004: International Conference, Assisi, Italy, Lecture Notes in Computer Science, Vol. 3044, pp. 226-235, Springer-Verlag, 2004

[EE] NIIF ClusterGrid Project, http://www.clustergrid.niif.hu