A Formal Methods Tool Suite for Education

June 13, 2001

Abstract

The project aims at systematically supporting the trend of modern education in computer science to involve software tools for illustrating the power and practical impact of the introduced concepts. Based on an innovative educational platform, students are meant to experience various tools, their differences, strengths and weaknesses, and therefore the impact of design decisions, modelling techniques, data structures and algorithms. The web-based design of this experimental platform will not only guarantee a world wide access to the provided functionality, but also an international cooperation already on the student level.

1 Scientific Objectives

Modern education in computer science increasingly involves software tools for illustrating the power and practical impact of the introduced concepts. This trend is particularly true for formal methods' education, which due to its mathematical and foundational character easily appears "dry," and far removed from relevant practice. Integrating the use of tools into the educational activities of our students teaches them the impact of making various choices among different models and methods; exposes them to interesting problems and case studies; and eases their passage through foundational concepts underlying formal methods. Optimally, students should be confronted with various tools of different types to experience the sensitivity of certain design decisions, and to develop a feeling of where to apply which kind of modeling (in particular on which level of abstraction) tool or method.

The cooperation between Professor Jeannette Wing (CMU) and Professor Bernhard Steffen (University of Dortmund) aims at providing a powerful environment built as a web-based application service for formal methods-based tools. This environment allows students to **experiment** easily with the integrated tools and functionalities, by

1. *running the* (stand-alone or integrated) *tools* on libraries of examples, case studies, and benchmarks;
2. *testing and running single tool functionalities*, capturing specific features offered by the integrated tools, on the same examples, from within a uniform graphical user interface; and
3. *constructing their own application-specific heterogenous tools* through combination of single functionalities coming from different tools.

This possibility of direct, *hands-on, experience* with the wealth of available formal methods tools and functionalities, will be complemented by a community service, where users (students, tutors and lecturers) may ask questions, propose case studies, and report on their experiences.

The realization of this platform will technically build upon the *Electronic Tool Integration* platform (ETI) developed in Dortmund (Section 2.2). An ETI extends service to the community by supporting

1. the distribution of course assignments as well as the collection of the corresponding results;
2. team work, by providing a version control-based platform, where teams can cooperatively develop software, case studies etc.; and
3. the evaluation of practical work: tutors and lecturers may log in as group members for assistance, as well as correction and testing purposes.

In the heart of the proposed project lies the adequate instantiation and adaption of the ETI platform for formal methods education, which requires the *selection* of appropriate tools, their *integration*, the preparation of their *interoperability*, as well as the design of an adequate *uniform graphical user interface* which works browser-independently over the net. These are important and non-trivial software engineering tasks, where the partners plan to cooperate. In fact, an easy handling of the platform, both for the integration of new tools as well as the later use by students will be critical for a wider acceptance.

We also envisage the opportunity of another use of formal methods for education, namely as a technology to elaborate on course work evaluation (item 3 above): Using the inherent power of formal methods-based analysis techniques we want to generalize the classical paradigm of multiple choice examinations. Based on the projected experimental platform, this goal seems particularly worthwhile in order to evaluate practical work. These are very new ideas of high potential and importance. One need only keep in mind that Dortmund expects 1200 beginners this fall!

2 Current Status

This section briefly summarizes the status quo relevant for the project, in three steps:

- a brief discussion of the preparatory work done at CMU and at the University of Dortmund;
- a more detailed project-oriented discussion of the ETI platform, which lies in the center of the application. Without this platform, which reflects a development effort of many man years, the proposed project would not be realistic;
- a discussion of related work focussing on the technological aspects of the exerimentation platform.

2.1 Current Status at the Partner Institutions

CMU is a world's leading institution in formal methods-based tools research. Not only have many important tools their origin at CMU, but also their evaluation and use for software engineering is exceptional here. In particular, Jeannette Wing is well-known for her activities in bringing formal methods tools into the area of software engineering, both in teaching and in the research community. In fact, it is very much her contribution that CMU's educational program more and more focusses on formal methods tools. She is also on the advisory board of the ITiCSE Working Group on Formal Methods Education, coordinated by Dr. Vicki Almstrum of the University of Texas, Austin.

Dortmund's major contribution and experience concerns the ETI platform. The group in Dortmund develops and uses this platform for many years. Special is the use as an electronic component for Springer's International Journal *Software Tools for Technology Transfer*. Since 1998 we have been organizing international ETI-days, typically co-located with the European Conference on Theory and Practice of Software. Since the start of this year, we are beginning also to instantiate ETI as a teaching environment.

The charm of the proposed project is that it combines two strong movements: the systematically prepared activities on changing the CMU-curriculum towards a modern tool-based education, and the platform developement effort in Dortmund, which it tailored to support the realization of such curricula.

2.2 The ETI Online Service

The functionalities offered by the ETI platform cover two distinct needs:

- features used to organize the ETI community like (tool-specific) discussion groups, newsletters, a developer's corner etc., and
- features that give direct access to the tools contained in the ETI site's tool repositories.

Together, these features allow a user to retrieve information on the available tools, execute single tool functionalities, combine tool functionalities, execute tool combinations, and exchange and discuss information and experiences with tool providers and other ETI users in a cooperative virtual space.

Logically, each ETI-site's platform is built using a four layered software architecture (see Fig. 1). The *persistency* (database) layer stores the tool repository as well as the discussion groups, mailing list archives, and user data. On top of this layer, the *feature* layer provides the real functionality offered by the ETI platform. This layer contains a Java-based Web application which implements the community services as well as a C++-based tool management application that gives access to the tool-related functionality. The *Internet access* layer makes the functionality offered by the platform accessible via the Internet. For its implementation we use a standard HTTP server (like Apache [1]) to make the community services available to the public. The functionality provided by the tool management software is encapsulated by the tool Web access server which can be connected via Java RMI [8]. The *presentation* layer contains software components that enable the site users to access the functionality provided by the feature layer via an intuitive graphical user interface. Here, a standard Web browser can be used to participate in the community infrastructure. In contrast, a Java client (called ToolZone client) is required to gain access to the tools contained in the tool repository. This ToolZone client can be launched from the Web browser using the Java Web Start technology [15].

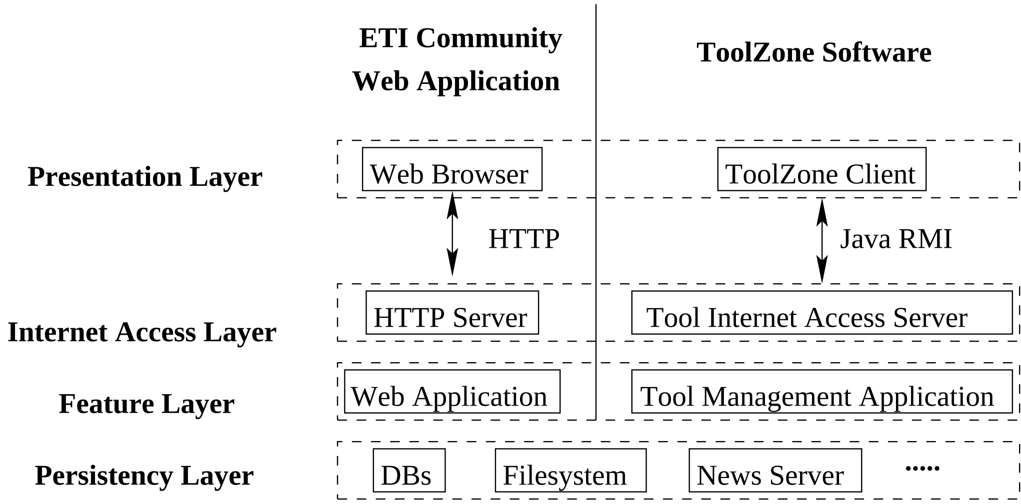

Figure 1: The Logical Platform Architecture

Usability, security, openness, and maintainability are the key requirements for a successful Internet service. In the ETI platform they are addressed as follows:

2.3 Usability

- Intuitive graphical user interface (GUI) for the client application: ETI provides an elaborate GUI with help functionality and hypertext support.
- Easy access to the features provided by the platform: The taxonomy-based retrieval mechanism together with the synthesis of coordination sequences from loose specifications enables a goal-oriented use of the platform without requiring any knowledge about the repositories' content and the interface-constraints between the integrated functionalities.
- Minimal set of requirements for client machines: The community features of the ETI platform can be used with a standard Web browser, and the access to the tool repository is provided by a Java-based software which requires a standard Java Runtime Environment to be installed on the client machine.
- Availability and performance: ETI distributes its functionality over a server farm. A load-balancing component ensures fast and reliable access to the provided functionality.

2.4 Security

The ETI platform offers protected home areas and a profile-based security concept to control the access to personal data and functionalities. In addition, sensitive parts of the community features are only accessible using Secure Socket Layer (SSL) based encryption.

2.5 Openness

A posteriori-integration is a key requirement for the ETI platform: each tool should be able to be integrated easily into the tool repository as an ETI-activity. In fact, ETI makes no assumptions on the design, the availability (e.g. communication via source code or binary version, operating system), and the accessibility (e.g. system calls, Java RMI [8], CORBA [21]) of the tool functionalities.

2.6 Maintainability

End users want easy, fast and reliable access to the functionality provided by the ETI platform. In particular, they want to

- browse through the tool repository looking for available activities;
- combine activities to coordination sequences and execute them using the ToolZone client;
- access tool-related information and discussion groups via a standard Web browser.

The corresponding maintainance of the ETI platform, which we support by means of the ETI community site, requires cooperation by people of different skills and profiles. We distinguish four groups of experts involved in different ways with the platform: tool providers, tool integrators, platform developers, and site managers.

Tool Providers

Tool providers are end users with special focus. Whereas the standard end user looks for candidate tools or a combination of tool functionalities to solve a certain problem, the tool provider is mainly interested in publishing his tool and getting valuable feedback by the end users. In addition to the tool itself, the tool provider supplies benchmarks, examples, and descriptions defining the tool's profile.

Tool Integrators

Tool integrators make new activities and types available to the ETI tool repository. They investigate the tools to be integrated, identify new activities, and establish connections between the new activities/types and already available ones. Thus they need a good understanding of the tools to be integrated and of the application domain modeled in the tool repository. In addition, they must be expert in C++, the language of the tool management application. We support the integration task by providing a tailored process together with some utility tools (see [3]).

Platform Developers

We distinguish two groups of platform developers. One group providing new building blocks, which can then be used to extend the ETI community application. An another group adding new functionality to the ToolZone software. Whereas building block developers are only required to have Java skills, ToolZone software developers must have Java and C++ knowledge. In addition, ToolZone software developers need a good understanding of the platform's architecture.

Site Managers

Site managers build up and maintain an ETI site. This includes setting up the servers, installing the required software and customizing the site-specific version of the community application. They need expertise about UNIX-based operating system (e.g. Linux or SUN's Solaris) for the installation process. The customization of the Web application, which requires some fundamental HTML knowledge in order to adapt the look and feel of the Web site, is done with our Service Definition Environment.

3 Related Approaches

We are not aware of any project which combines the features provided by the ETI platform. Roughly, the approaches and tools coming closest to our platform can be classified as follows:

3.1 Approaches Focusing on Tool Access

- **Link collections:** There are a lot of Web sites available which provide links to software tools in a special application domain. Examples are the Petri Nets Tool Database [22] or the Formal Methods Europe [9] database.

  The ETI platform covers this aspect of tool access by providing links to the home page of any tool available in the tool repository.

- **Software archives:** In contrast to software archives, like collections of Java Beans, ETI is not a distribution platform: tool providers retain all legal rights to, and responsibilities for their software. The user cannot download the software. He can only experience using the tools installed on one of the ETI application servers. To obtain a license of the chosen tool, the user can contact the tool provider via the information supplied by the platform.

- **Web sites providing execution facilities of a single tool:** Some Web sites like the HyTech home page [12] or the SMV guided tour [23] give access to tool functionality via an HTML form. In most cases simple CGI [4] scripts are used to access a single functionality of the tool and to deliver the results to the user in text form.

This kind of tool access is also available within the ETI platform. In addition to that, users can combine tool functionalities to sequential programs and then execute them.

3.2 Approaches Focusing on Tool-specific Communication

- **News Groups and Mailing Lists:**

  Mailing list like the Uppaal [17] user's list, or news groups provide a means for tool-specific communication, however, without any tool related functionality. They only address one aspect of the ETI platform, i.e., the communication functionality. Participation in any discussion assumes running both an email and a news client.

- **Community Sites:**

  Community sites like the Java Developer Connection [14] offer discussion groups, email notification, and information in a special application field. As in the approach above, these community sites only cover the community functionality, however, in a more integrated fashion. They do not provide Web-based experimentation support.

3.3 Approaches Focusing on Tool Integration and Coordination

- **PROSPER:**

  The PROSPER (Proof and Specification Assisted Design Environments) [7] toolkit provides an infrastructure where existing verification tools can be integrated into nearly any application (like CAD and CASE tools). This provides end users with applications enriched with new validation features, but it does not provide any user-level handle to coordinate verification tools. Coordination is exclusively is done by the PROSPER expert.

- **ToolBus:**

  Closest to ETI is ToolBus [2, 16], a very general environment for tool coordination. However, being based on a process algebra, its coordination specifications are much more general and complicated than in ETI. Thus the user community is rather restricted.

In contrast, the ETI project provides an open platform, where the tool coordination facility is designed to be available to a very wide community. To coordinate different tool features, experienced users may use ETI's coordination language HLL (High-Level Language). Unexperienced users are additionally supported by ETI's synthesis component. Here, the user can define *what* he wants to achieve, instead of specifying *how* he wants to achieve his goal. ETI's automated synthesis component then generates sequential programs that implement the intended task. Generated programs can be executed via the Internet.

4 Details and Realization

This section summarizes the schedule, the involved teams together with their respective contributions, a budget justification, and a summary of past interaction by the two PIs.

4.1 Planned Schedule

The project will roughly proceed in four phases, each corresponding to the four semesters of the two project years.

1. Set up the ETI infrastructure for the Dortmund/Carnegie Mellon cooperation. Analyse the requirements for an adequate ETI instantiation.

2. Instantiate the ETI platform according to the requirements collected in the first phase and optimize the graphical user interfaces for educational purposes. This will result in the first version of the (Dortmund/Carnegie Mellon) CMUD-Education Platform by the end of the first year.

3. Evaluate the CMUD-Education Platform in practice: here coursework both at Carnegie Mellon and at Dortmund is already planned. The instantiation and user optimatization process will continue during this phase.

4. Continue the evaluation process of the CMUD-Education Platform and write an experience report to be published e.g. in Springer's International Journal on *Software Tools for Technology Transfer*. The CMUD-Education Platform should be stable by the end of the second year.

Whereas the Dortmund team will focus on maintaining and adapting the software infrastructure, the Carnegie Mellon team will contribute as technology providers, tool integrators, and general software engineering experts giving direction and support in improving the CMUD-platform. Both groups are known for their knowledge about formal methods, which constitute the application domain as well as the base technology in this project.

4.2 The Teams

Comment: For these people CV's must be included in the application!

Dortmund

1. Professor Bernhard Steffen

2. Dr. Tiziana Margaria

3. Dipl.Inform. Volker Braun

4. Dipl.Inform. Claudia Gsottberger

Carnegie Mellon

1. Professor Jeannette M. Wing

2. Professor David Garlan

3. Doctoral candidate, Oleg Sheyner

4.3 Respective Contributions

Besides the discussion about and evaluation of the CMUD-Platform, which will be the driving force for the further development, and which will happen in strong cooperation between the two partners, we envisage the following division of labour:

Dortmund

Dortmund will provide the ETI platform, the basis for the CMUD-Educational Platform, together with a number of already integrated formal methods tools like the three real time verification tools, *Hytech, Kronos,* and *Uppaal*, as well as the CADP verification suite developed in Grenoble (France).

During the project, Dortmund will focus on the overall technical development and maintainance of the platform, as well as on the integration of new formal methods' tools into the platform. This is not restricted to the integration of individual tools, but also to methods supporting the integration process as such.

Similar to the previous events for ETI

- The STTT Special Section (as publication, and the editorial too).
- The STTT Day in July 1998 (colocated with ICALP).
- The ETI Day in ETAPS Berlin.
- The ETI Day at ETAPS Genova.
- The preparation of a cooperation with FMICS.

we are planning to organize international meetings concerning the CMUD-Educational Platform in order to obtain feedback and encourage cooperation. In this course, the range of possibilities will also be illustrated along Dortmund's activities to include tools supporting undergraduate studies.

Carnegie Mellon

The Carnegie Mellon team will provide formal methods tools developed in house, including model checkers (e.g., *SMV*), real-time model checkers (e.g., *Verus*), hybrid-systems model checkers (e.g., *Checkmate*), a combination model checker and theorem prover (e.g., *SyMP*), and application-specific model checkers (e.g., *Livingstone*, applied to space missions). We can also provide software architecture analysis tools such as *ABLE*. For educational and research purposes, the Carnegie Mellon team also makes heavy use of formal methods tools developed elsewhere and we would like to integrate them as well. These tools include theorem provers (e.g., *PVS*, developed at SRI), model checkers (e.g., *FDR*, developed in the United Kingdom), and design checkers (e.g., *Alloy*, developed at MIT). We also have plans to investigate the use of other probabilistic model checkers such as *Prism*.

We already use *SMV*, *FDR*, *Alloy*, and *Statemate* in the Master's of Software Engineering program at Carnegie Mellon, but they are not integrated. Two of the five required core courses in the MSE program use these tools: Models of Software Systems and Analysis of Software Artifacts. Both Professors Jeannette Wing and David Garlan have been instrumental in these two courses curricula and have taught these courses from their inception. This fall we also plan to include new tools such as a deduction engine (successfully used at Stanford for educational purposes) to complement our model checkers. It is on the CD-ROM that comes with the book *Language, Proof, and Logic* by Barwise et al.

We look forward to having the ETI platform give us an integrated approach toward the disparate set of tools we use. Moreover, for students, the user-oriented graphical interfaces will be critical for eventual adoption of not just the tools, but formal methods in general. Thus, we will work closely with the Dortmund team to gather feedback from the students to improve and adapt the ETI platform.

We will gather this feedback systematically, through questionnaires and interviews, per example problem or homework assignment.

While we expect initially to use the ETI platform of tools at Carnegie Mellon for the Master's of Software Engineering program, we plan also to use it at both the undergraduate level and doctoral level.

- Professor Edmund Clarke of Carnegie Mellon is creating a new undergraduate course this fall that will be able to use this integrated tool suite. Thus, by the second year of this project, he should be able to use our platform for his use in the second and subsequent times he teaches this new course.

- Professor Jeannette Wing, Professor David Garlan, and other faculty at Carnegie Mellon have been using tools in doctoral-level courses and seminars. Having an integrated tool suite of formal methods tools will enable them to expose doctoral students to a wider variety of tools and applications of formal methods. Many hardware companies such as Intel and software companies such as Microsoft recruit our doctoral students specifically for their expertise in model checking; we would like to broaden our students' expertise not only to other formal methods tools and techniques, but more importantly how they can be used to complement each other.

In short, this plan to integrate formal methods tools fits in with curriculum changes in Carnegie Mellon's educational programs already underway. Having an integrated tool suite will greatly enhance all these planned educational activities.

Finally, many of these tools are being developed as part of *The Rare Glitch Project*, a research project funded by the Army Research Office, focused on the verification of embedded systems. One of the ARO program manager's goals is to integrate our tools—a goal consistent with this proposal's plans.

4.4 Budget Justification

The hardware and software required for the project are available both in Dortmund and at Carnegie Mellon. Thus only travel funding for the mutual visits is required. These meetings are necessary, in the beginning to detail the project plan, later to discuss the development of the platform, and finally to write the experience report about the project. Each meeting will last from one week to ten days, depending on the purpose of the visit.

We plan minimally for the following trips:

Year One

1. Spring 2002: The whole project team will meet in Dortmund, where the current status of the ETI platform will be demonstrated and where the project schedule will be fixed. This trip requires funding for three Carnegie Mellon team members to travel to Dortmund.
2. Summer 2002: One trip by three project members from Dortmund to Carnegie Mellon. This trip will be shortly after the initial meeting, in order to install the platform for immediate use by Carnegie Mellon.
3. Fall 2002: One trip by two members from the Carnegie Mellon team to Dortmund and vice versa. The purpose of these two trips is to get feedback from the Carnegie Mellon team on how well the technical integration of the platform played out in the MSE courses. (Both the Methods and Analysis courses are offered during the fall term.) Collection feedback from the students will begin.

Year Two

1. Spring 2003: One trip by two members from the Carnegie Mellon team to Dortmund and vice versa. The purpose of these two trips is to obtain complete student and faculty feedback from the Carnegie Mellon users and for the Dortmund team to adapt their ETI platform to this feedback.
2. Summer 2003: Another pair of exchange visits, to stabilize the platform and to prepare for the final project meeting.
3. Fall 2003: The whole project team will meet in Pittsburgh for evaluation and for initiating the experience report. This will require travel money for the Dortmund team members.

Collaboration

The project leaders Professor Jeannette Wing (CMU) and Professor Bernhard Steffen (University of Dortmund) have known each other for more than a decade. They cooperated during various events, in particular, during the ACM Workshop on *Strategic Directions in Computing Research* in Boston (MA). Professor Wing was then responsible for the Working Group on Formal Methods, which resulted in a common position paper published in the December 1996 issue of the *ACM Computing Surveys*.

Another tight cooperation concerns the international journal of *Software Tools for Technology Transfer*, published by Springer-Verlag, where Professor Steffen is co-Editor-in-Chief and Professor Wing a member the editorial board. Finally, both partners have been cooperating in the editorial board of TOPLAS, *ACM Transactions of Programming Languages and Systems*, and are active in the 21st Century Initiative on formal methods education, where it is also planned to establish an ETI-based platform.

References

[1] *The Apache Project*, http://www.apache.org.

[2] J. A. Bergstra, P. Klint. *The ToolBus Coordination Architecture*, Proc. int. Conf. on Coordination Models and Languages, LNCS 1061, pp. 75-88, Springer Verlag, 1996.

[3] V. Braun, T. Margaria, C. Weise: *Integrating Tools in the ETI Platform*, [30], pp. 31-48.

[4] *CGI: Common Gateway Interface*, http://www.w3.org/CGI/.

[5] *Cold Fusion, a cross-platform Web Application Server*, http://www.allaire.com/products/coldfusion/.

[6] T. David, *Server Side Includes (SSI) Tutorial*, http://www.tdscripts.com/ssi.html.

[7] L. A. Dennis, G. Collins, M. Norrish, R. Boulton, K. Slind, G. Robinson, M. Gordon, T. Melham: *The PROSPER Toolkit*, Proc. TACAS'2000, Int. Workshop on Tools and Algorithms for the Construction and Analysis of Systems, LNCS 1785, pp. 78-92, Springer Verlag.

[8] T. Downing: *Java RMI, Remote Methond Invocation*, IDG Books Worldwide Inc., 1998.

[9] *Formal Methids Europe*, http://www.fmeurope.org.

[10] J. Hunter, W. Crawford: *Java Servlet Programming*, O'Reilly, 1998.

[11] *International Journal on Software Tools for Technology Transfer*, http://sttt.cs.uni-dortmund.de.

[12] *HyTech Home Page*, http://www-cad.EECS.Berkeley.EDU/ tah/HyTech/.

[13] *JavaBeans: The only Component Architecture for Java Technology*, http://java.sun.com/products/javabeans/.

[14] *Java Developer Connection*, http://developer.ja.va.sun.com/developer/.

[15] *Java Web Start*, http://java.sun.com/products/javawebstart/.

[16] P. Klint, P. Olivier. *The TOOLBUS Coordination Architecture: A Demonstration*, Proc. of 5th Int. Conference on Algebraic Methodology and Software Technology (AMAST'96), LNCS 1101, pp. 575-578, Springer Verlag, 1996.

[17] K. G. Larsen, P. Petterson, W. Yi: *Uppaal in a nutshell*, Int. Journal on *Software Tools for Technology Transfer*, Vol. 1, pp. 134-152, Springer Verlag, 1997.

[18] M. Lutz: *Programming Python*, O'Reilly, 2001.

[19] T. Margaria, V. Braun, J. Kreileder: *Interacting with ETI: A User Session*, [30], pp. 49-63.

[20] T. Margaria, B. Steffen: *Coarse-grain Component Based Software Development: The* METAFrame *Approach*, Proc. STJA'97, "Smalltalk und Java in Industrie und Ausbildung", 10.-11. September 1997, Erfurt (D), ISBN 3-00-001828-X, pp. 29-34.

[21] OMG. *The Common Object Request Broker: Architecture and Specification Revision 2.0*, Technical Document ptc/96-08-04.

[22] *Petri Nets Tool Database*, http://www.daimi.aau.dk/PetriNets/tools/db.html.

[23] *SMV guided tour*, http://www.cs.cmu.edu/ modelcheck/tour.html.

[24] B. Steffen, T. Margaria: *METAFrame in Practice: Design of Intelligent Network Services*, in LNCS State-of-the-Art Survey *Correct System Design – Recent Insights and Advances*, Lecture Notes in Computer Science 1710, pp. 390 - 415, 1999.

[25] B. Steffen, T. Margaria, V. Braun: *The Electronic Tool Integration platform: concepts and design*, [30], pp. 9-30.

[26] B. Steffen, T. Margaria, A. Claßen, V. Braun: *The* METAFrame*'95 Environment*, Proc. CAV'96, Int. Conf. on Computer-Aided Verification - Juli-Aug. 1996, New Brunswick, NJ, USA, LNCS 1102, pp. 450-453, Springer Verlag.

[27] B. Steffen, T. Margaria, A. Claßen, V. Braun, N. Kalt: *Hierarchical Service Definition*, Annual Review of Communications, Int. Engineering Consortium (IEC), pp. 847-856, 1997.

[28] B. Steffen, T. Margaria, A. Claßen, V. Braun, M. Reitenspieß, M. Wendler: *Service Creation: Formal Verification and Abstract Views*, 4th Int. Conf. on Intelligent Networks (ICIN'96), pp. 96-101, 1996.

[29] B. Steffen, T. Margaria, B. Freitag: *Module Configuration by Minimal Model Construction*, Techn. Rep. MIP-9313, Fak. für Mathematik und Informatik, Universität Passau (Germany), 1993.

[30] *Special section on the Electronic Tool Integration Platform*, Int. Journal on *Software Tools for Technology Transfer*, Vol. 1, Springer Verlag, November 1997

[31] T. Thai: *Learning DCOM*, O'Reilly, 1999.

[32] L. Wall, T. Christiansen, J. Orwant: *Programming Perl*, 3rd Edition, O'Reilly, 2000.