Proceedings of the IASTED International Conference
Internet & Multimedia Systems & Applications and Visual Communications
March 17-19, 2008 Innsbruck, Austria
ISBN Hardcopy: 978-0-88986-727-7 / CD: 978-0-88986-728-4

AUTOMATIC TEST CASE GENERATION THROUGH A COLLABORATIVE WEB APPLICATION

Alessandro Oliveira Arantes[1], Nandamudi L. Vijaykumar[2], Valdivino Alexandre Santiago[3], Adenilson Roberto Carvalho[2]

[1]Institute for Advanced Space Studies (IEAv), Aerospace Technological Center (CTA),
P. O. Box 6044 – 12228-970 – São José dos Campos – SP – Brazil

[2]Associated Laboratory of Computing and Applied Mathematics (LAC)

[3]Department of Atmospheric and Space Sciences (CEA)

National Institute for Space Research (INPE),
P. O. Box 515 – 12245-970 – São José dos Campos – SP – Brazil

alessandro.arantes@ieav.cta.br,{vijay,adenilson}@lac.inpe.br,
valdivino@das.inpe.br

ABSTRACT
In order to develop a software product where teams spread over a country or even over the world can work together, the distributed development of software is a natural approach. Therefore, web appears as a valuable resource enabling the cooperative development. Collaborative work joins efforts of several members of a team to coordinate their tasks with an objective of reaching a specific goal. In particular, for software, collaborative web applications are powerful resources that can help different teams to cooperatively address process activities related to the software development life cycle, especially those related to testing. This paper presents a web-based tool, WEB-PerformCharts, to enable test designers to generate black-box test sequences, remotely via Internet, based on Statecharts representation of the specification. WEB-PerformCharts tool enables a reactive system (software specification) to be modeled in Statecharts, and generates test sequences according to a test case generation method using the Web. The paper describes how a model specified in Statecharts can be used to generate test sequences (PerformCharts tool), besides showing the necessary implementations in the tool to develop a collaborative web application. A case study is presented in order to demonstrate the usability of the tool. The paper has a major contribution in providing support to test processes in a distributed environment besides discussing the use of a high-level technique, Statecharts, to model the specification.

KEY WORDS
Interface, Collaborative Systems, Cooperative Work, Web Application, Statecharts, Software Testing.

1. Introduction

In order to reduce development costs many software companies are using computer-supported collaborative tools to overcome the distance for geographically distributed settings in software development.
Collaborative work is nothing more than joint efforts to develop a project, and web-based systems offer an effective possibility among members of a team to coordinate their tasks in order to reach a major goal. Collaborative web applications are a powerful resource and can help different teams to cooperatively address process activities related to the software development life cycle.
Verification and Validation [1] is one of the key issues within software development process, especially when dealing with complex software applications such as space applications and nuclear plants monitoring. The validation process depends heavily on the quality of test sets applied to the product, which leads to the necessity of generating proper test sequences. It would be interesting if such a generation must rely entirely on a scientific basis in order to avoid their (test sequences) inadequacy in revealing errors. So, testing activities are essential, and depending on the software's complexity, it can involve many scattered teams working together.
A test sequence can be defined as taking a software being tested from a given state (or configuration when parallel activities are considered) to another state or configuration that is reachable. A technique to derive test sequences from the specification in design phase is one approach to generate them since the major advantage is their availability in the very early phases of the software development life cycle. Therefore, if software specification is modeled, methods can be applied to the model in order to generate test cases. For example, if the technique to model a software specification were a Finite State Machine (FSM), some methods that can generate

test sequences are: T, UIO, DS, W and Switch Cover [2], [3], [4] and [5]. However, features usually present in complex software, like parallel activities and encapsulation, are hard to represent using FSMs. Hence, higher-level techniques that support such features should be investigated.

This paper presents a tool, WEB-PerformCharts, that incorporates two main features: a web-based tool so that test designers can generate and obtain test sequences remotely via Internet, addressing distributed software development situations; and, an alternative to FSM by using Statecharts to generate test sequences.

This paper is organized as follows: Section 2 discusses an introduction about collaborative systems and their applications. Section 3 discusses the importance of testing critical systems and also presents PerformCharts by explaining how a model represented in Statecharts is converted into a FSM from which test sequences are generated. Section 4 presents the WEB-PerformCharts tool. Section 5 presents a case study with results. Finally Section 6 concludes the paper.

2. Collaborative Systems

The main idea of collaborative systems is helping people involved in a common task supporting communication, coordination and cooperation.

A large distribution of great corporations added to the fast growth of internet has increased the need for cooperative applications [6]. These applications have means to promote any internet user and allows a large cost saving, time saving, and increasing teamwork and efficiency, since all manipulated data by one user can be immediately perceived by all other users at remote locations [6].

Web-based applications have become a common practice since they offer a low cost solution; therefore, in this architecture the client can use any operational system and it requires no other proprietary software. Also, another advantage in a web-based architecture is the fact that, nowadays, many people have easy internet access. Whenever updates are necessary, this is conducted only in the server where the tool is hosted without any necessity for the users to reinstall any kind of software. So, collaborative web-based systems (also known as E-collaboration) is a reality adopted for many companies to develop their applications.

There are several categories of collaborative tools such as Group Document Handling, Real-time conferencing, Non real-time conferencing, Electronic Meeting Systems (EMS) and Electronic Workspace. WEB-PerformCharts tool belongs to category of Electronic Workspace since its primary idea is to provide a common space to teams to coordinate and organize their work with documents and files centralized in an on-line server [7].

Some features that are relevant for collaborative systems are:

- E-mail notifications: to inform tasks, changes or new activities;
- Project management: to manage the access level of users and delegate tasks to members of a group;
- File and document sharing: all sort of documents, including software requirements specifications and documents related to the test process are available to a group of people to access them.

File and document sharing is the most common feature in such tools, and at the same time most needed collaboration service [7].

Space research organizations like National Institute for Space Research (INPE) demand high software quality, for instance, embedded into satellite on-board computers. Many a time, the teams involved in satellite missions are not exactly in one place due to joint collaboration among space agencies to develop space applications. In this scenario, an on-line cooperative tool is important to aid the software testing activities. An example of such a scenario is shown in Figure 1.

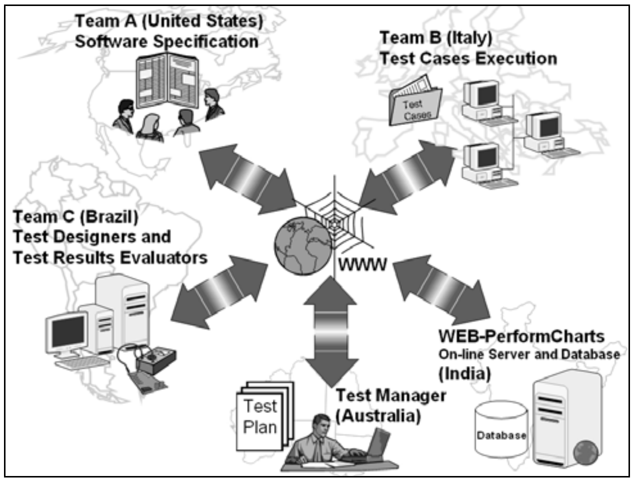


Figure 1. Example of cooperative work

3. Testing Critical Systems

Occurrence of faults in many systems controlled by software may cause inconvenience, but not serious damages. However, there are certain categories of systems where software faults can result in significant economic losses, physical damages or threats to human life. These systems, in general, are classified as critical systems and space software falls into this category. Space software systems demand high quality and high cost technologies to execute complex tasks and testing such systems is essential to guarantee their reliability.

The work described here uses a formal approach to represent specifications in order to enable automated test case generation. INPE has been developing data collecting, remote sensing and scientific satellites since 1979. Satellites usually have complex subsystems with embedded software that are reactive by nature responding to events. An adequate method for modeling reactive systems is using FSMs as they are a natural choice.

However, a complex software with several states requiring explicit representation of parallel activities in order to understand its behavior is hard to be represented by an FSM. A technique that could overcome these drawbacks are Statecharts.

Statecharts can be used to specify reactive systems [8]. They are formal ([9] and [10]), can be handled computationally, and consists of the following elements: states, events, conditions, actions, and transitions [8], [9], [10] and [11]. When considering parallel components, one must use the global state of the specified system known as configuration in Statecharts. The definition of an initial configuration is mandatory. However, it can be bypassed through entry by history, in which the last visited state must be "remembered" while returning to the component. In several practical situations the state change can occur continuously at any instant of time. Statecharts classify events as external (explicitly stimulated) and internal (automatically sensed and stimulated) [12]. Statecharts already inlay in its formalism the following internal events: true[condition], false[condition], entered[state], exit[state]. Events can be guarded by using conditions. One such condition, very much used in Statecharts, is in(state s) which becomes true if state s is active. Composed conditions involving operators *and, not* and *or* can also be used. Action, another basic element of Statecharts, is considered as an internal event and it is stimulated automatically in other orthogonal components.

The general notation of a transition is: event[condition]/action, meaning once the event is enabled and the condition satisfied the transition is executed. After the transition has occurred the reaction continues by executing the action, which is another reaction. Statecharts where first used to evaluate performance of reactive systems by associating them to Markov chains, which are in fact FSM [12] with the exception that the events have to follow an exponential distribution. Thus a tool PerformCharts was created.

It is also possible to associate probability to a transition in situations where a same event takes a source state to more than one destination state considered as a conflict resulting in non-determinism. Therefore, the original notation was changed to: event[condition]{probability}/action in the PerformCharts tool [13].

PerformCharts tool has been adapted to use Statecharts to represent a reactive system with the objective of generating test sequences by converting the representation into a FSM.

A text-based interface PcML (PerformCharts Markup Language) [14], based on XML (Extensible Markup Language) [15], has been developed. PcML may be edited through any text editor and it is parsed by a code in Perl language that converts it into the main program of PerformCharts. This code, in C++, is compiled and linked with other classes to obtain the performance measures or an FSM also specified in XML format. This FSM is the basis to generate test sequences by associating with a test case generating method. The method being used at the moment is *switch cover* [5] implemented within yet another tool CONDADO [3].

4. WEB-PerformCharts

As already mentioned, WEB-PerformCharts tool has been developed in order to enable different teams working in software testing by creating, storing and sharing projects in a database through Internet access. It is a web application idealized to aid software testers that work in different places cooperating in a common project. The objective of WEB-PerformCharts is to approach teams combining their expertise and know-how in order to benefit software's quality.

In order to achieve this objective, PerformCharts tool has been adapted to run remotely through a web-based interface and to be completely functional hosted in a web server. An on-line database has been made available in order to allow testers to load and save testing data from anywhere, instead of using only local files to store data.

All involved technologies such as HTML, PHP, MySQL and Apache server software are free. Thus, the web-based system can be entirely free of costs when considering software packages. At the moment, PerformCharts can be installed in servers based on Windows platform, but a Linux version is under development.

When logged in, the user can create, edit or delete projects with their PcML specifications associated. Users can manipulate one project at a time and when a project is selected to be used (from a list with all available projects) it can be modified and run as many times as required, which is an interesting feature if the software was incorrectly modeled in Statecharts and requires changes in its specification. These changes can be seen by anyone who can access the same project. Projects created by any user can be shared between users and the implementation of workflow routines is under study acting like a "production line" of software testing. Also, communication between users can be done through the integration with their e-mail and the idea of split users into workgroups seems very useful and will be studied for implementation.

In theory there is no limit for the number of users to access WEB-PerformCharts; it just depends on the server capacity to support on-line workload as well as on the storage capacity. Since the number of users working can be huge, they must be organized according to their responsibilities (e.g. Administrator, User, Guest, Project Manager, General Manager, etc.) and this requires a secure and a robust management. The solution adopted for WEB-PerformCharts is an access level for different types of users. At the moment only two levels are set: Administrator that can work in any project and can create user accounts, and User that has authorization to work only in projects created by her or him. Through the web-based interface the test designer can manage her or his projects creating a new one, deleting, or modifying an existing project in order to obtain new test cases. All the test cases are stored in a database in the server to be accessed in the future by those who have the proper authorization. WEB-PerformCharts uploads a PcML specification to a web server using a web-based interface implemented in HTML and PHP. The upload interface is shown in Figure 2.

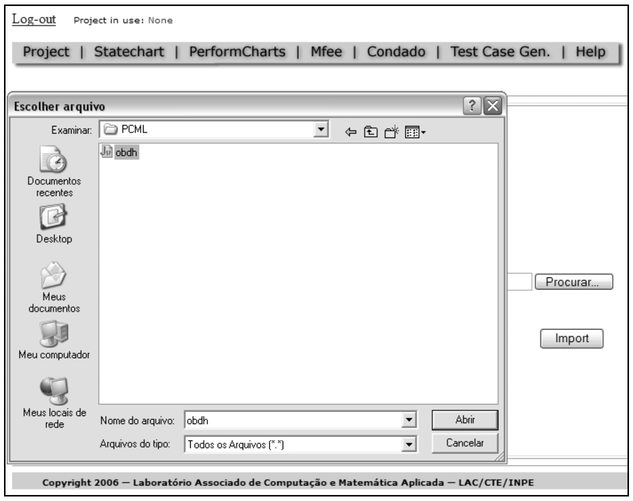

Log-out
Project in use: None
Project | Statechart | PerformCharts | Mfee | Condado | Test Case Gen. | Help
Escolher arquivo
Examinar: PCML
obdh
Documentos recentes
Desktop
Meus documentos
Meu computador
Meus locais de rede
Nome do arquivo: obdh
Arquivos do tipo: Todos os Arquivos (*.*)
Abrir
Cancelar
Procurar...
Import
Copyright 2006 – Laboratório Associado de Computação e Matemática Aplicada – LAC/CTE/INPE

Figure 2. Web server upload interface.

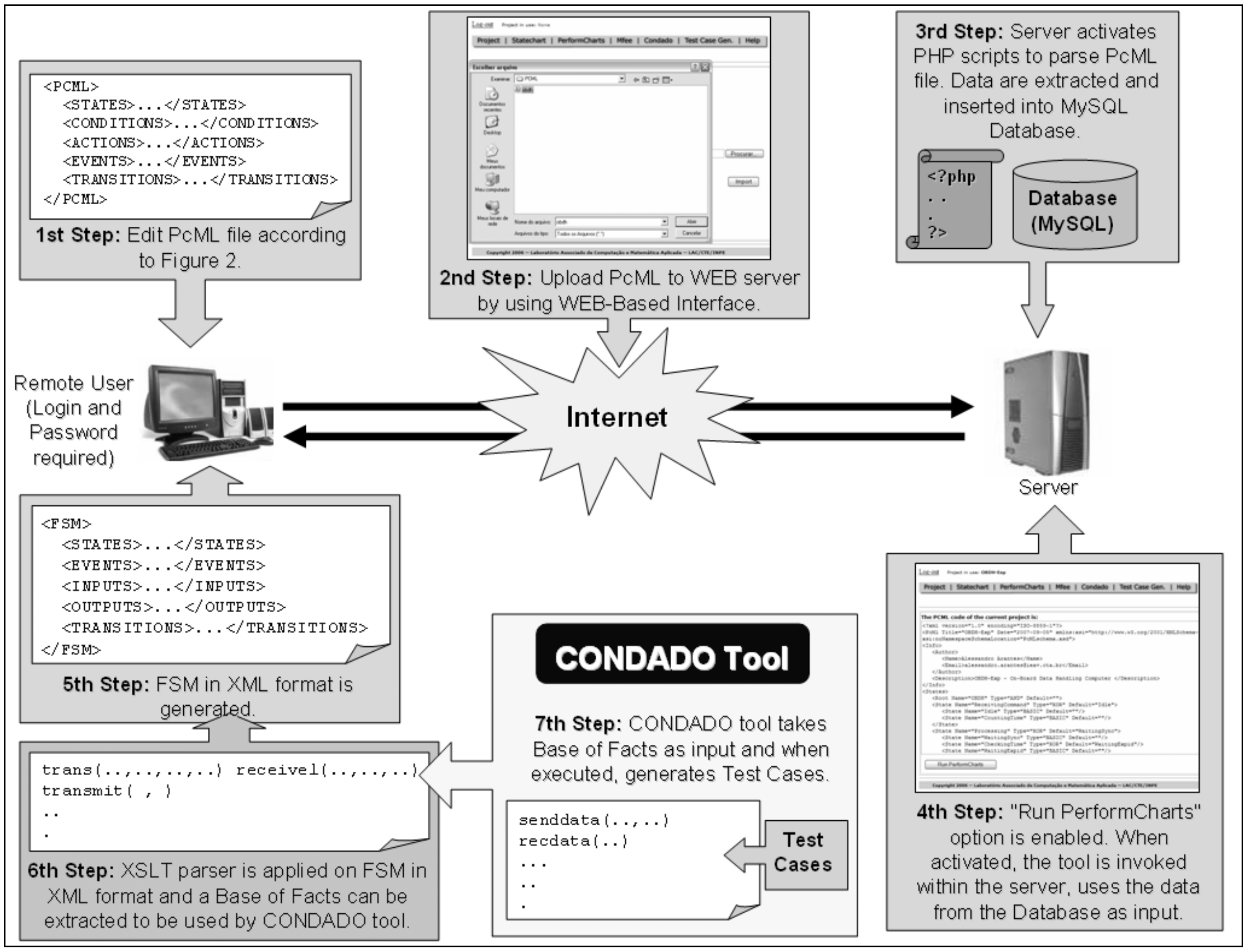
The server hosts PHP scripts which parse PcML file and extract all Statecharts data that is stored in a MySQL database from which the necessary data structures (to hold the encapsulation, states, events, conditions, parallel components and transitions) are created as well as the calls to appropriate methods to generate the FSM. When performance evaluation is required, a Markov chain is the result.
The output of the state-transition diagram is stored in the database and can be extracted in XML format for any other use. Steps to generate test sequences using WEB-PerformCharts are shown in Figure 3. The sequence diagram of WebPerformCharts is shown in Figure 4.
CONDADO tool accepts an FSM specified as a base of facts. Therefore, the FSM in XML format has to be converted into this base of facts. This conversion is achieved by using a parser written in XSLT.
Also, Transition Tour method (T-method) has already been implemented embedded in WEB-PerformCharts, allowing the complete test sequence generation on the web in a straight forward manner.

5. Results

In order to show the use of WEB-PerformCharts, a protocol specification [16] developed for the communication between a scientific experiment and the On-Board Data Handling Computer (OBDH) of a Brazilian scientific satellite has been chosen. The Implementation Under Test (IUT) is the command recognition component of the on-board software for an astrophysical experiment [17]. The FSM generated from WEB-PerformCharts can be converted by an integrated XSLT parser into the base of facts required by CONDADO which generates all input combinations based on *switch cover* method or, in other words, generates test cases for the system initially specified in PcML. If T-method is chosen, instead of *switch cover* from CONDADO, test sequences are generated directly in the web-based tool and can be extracted in XML format to be applied into the test execution tool.
Major advantage observed in this implementation is the importing and exporting of XML data. This has provided a huge benefit by opening doors to consider integrating other test case generation methods in the future since XML is a standard language and relatively easy to learn.

6. Conclusion

Sharing information in a collaborative work is, without any doubt, a necessity. The widely dispersed and decentralized mobile work is a very common trend and these features can result in great time and cost savings decreasing travel and infrastructure requirements for huge, centralized and expensive buildings.
As the name suggests, WEB-PerformCharts offers a multi-user web-based interface. The tool uses an on-line database system to allow test designers to generate test sequences from Statechart-based software specification.
WEB-PerformCharts has several advantages when compared to a conventional local system. It can be accessed in real time conditions from any place in the world at anytime. Its use is not restricted to a single computer and can be used from anywhere with a computer or laptop, an internet connection and a web browser. In addition, the tool is based on a high-level specificaton technique, Statecharts.
It has been shown that depending on the complexity of the reactive system, it is worth investing techniques that cater explicit representation of hierarchy and parallel activities in modeling software specifications for test sequence generation. In this respect, Statecharts come into picture.
However, one must bear in mind that dealing with higher-level techniques leads to more complexity in developing an automated environment and hence more computational effort. The paper did not discuss some aspects on test case explosion as this might occur depending on the number of states and the number of arcs of the generated FSM.
Some test sequence generating methods may really cause this explosion. Therefore, a simple pruning of a selected arc right in the Statecharts specification has been implemented and this enables to generate a partial FSM. On one hand, this has a drawback that a complete machine cannot be tested. On the other hand, it enables test case generation methods to deal with the machine.



```
<PCML>
  <STATES>...</STATES>
  <CONDITIONS>...</CONDITIONS>
  <ACTIONS>...</ACTIONS>
  <EVENTS>...</EVENTS>
  <TRANSITIONS>...</TRANSITIONS>
</PCML>
```

1st Step: Edit PcML file according to Figure 2.

2nd Step: Upload PcML to WEB server by using WEB-Based Interface.

3rd Step: Server activates PHP scripts to parse PcML file. Data are extracted and inserted into MySQL Database.

```
<?php
. .
.
?>
```

4th Step: "Run PerformCharts" option is enabled. When activated, the tool is invoked within the server, uses the data from the Database as input.

```
<FSM>
  <STATES>...</STATES>
  <EVENTS>...</EVENTS>
  <INPUTS>...</INPUTS>
  <OUTPUTS>...</OUTPUTS>
  <TRANSITIONS>...</TRANSITIONS>
</FSM>
```

5th Step: FSM in XML format is generated.

```
trans(..,..,..,..) receive1(..,..,..)
transmit( , )
..
.
```

6th Step: XSLT parser is applied on FSM in XML format and a Base of Facts can be extracted to be used by CONDADO tool.

7th Step: CONDADO tool takes Base of Facts as input and when executed, generates Test Cases.

```
senddata(..,..)
recdata(..)
...
..
.
```

Figure 3. WEB-PerformCharts architecture.

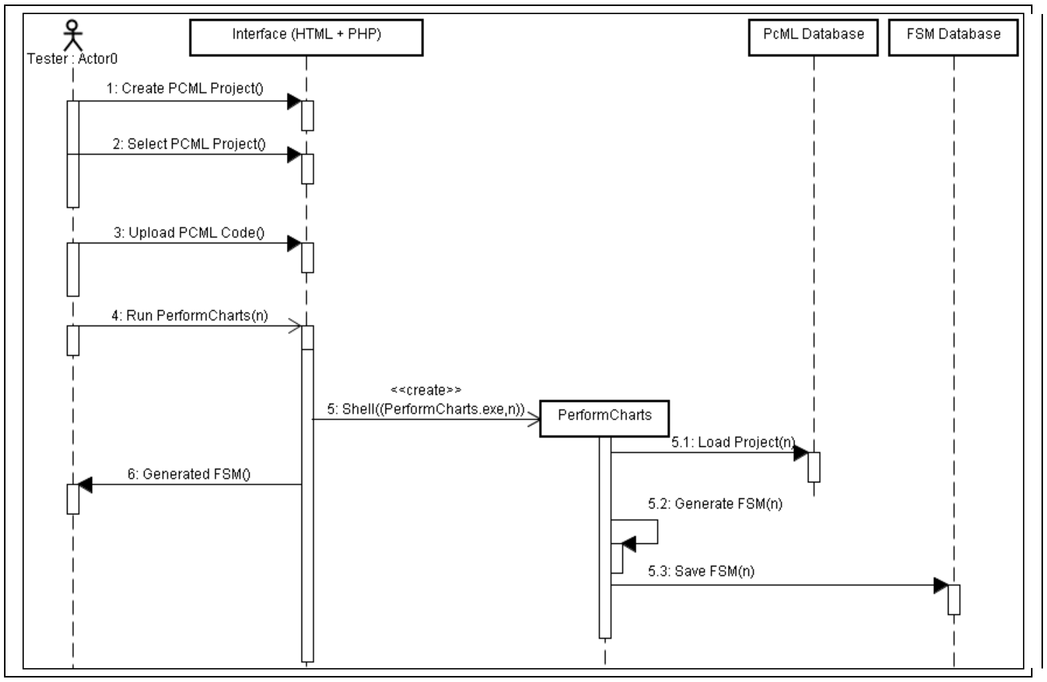


Figure 4. WEB-PerformCharts sequence diagram.

There are several other suggestions of minimizing the FSM from Graph Theory as well as from Model Verification and some of them will be considered for implementation within the tool. Future plans also include implementing other test sequence generation methods (for example Transition Tour) and these will be considered as cartridges to the system. With this, the user will be enabled with an option to apply the desired method to the resulting FSM. The main contribution of this paper is to enable PerformCharts for remote access with the objective of supporting test process in a distributed development environment. Another major contribution is the use of XML formatted documents which represent an important step in the standardization of the test data and distributed work. When other test case generation methods are implemented, the tool may be used to compare results from different methods. Also, in future, the WEB-PerformCharts is intended to be integrated to tools that perform automatic test execution allowing a further step towards the automation of the test process activities.

Acknowledgement

The authors would like to acknowledge the financial grant provided by CNPq/Edital Universal (Project N. 474031/2006-3)

References

[1] R.S. Pressman, *Software engineering - a practitioner's approach* (5th edition, McGraw-Hill International Editions, 2000).

[2] D. Lee, M. Yannakakis, Principles and Methods of Testing Finite State Machines – A Survey. In: *Proceedings of the IEEE*, 1996, 84(8).

[3] E. Martins, S.B. Sabião, A.M. Ambrósio, Condata: a tool for automating specification-based test case generation for communication systems". *33rd Hawaii International Conference on System Sciences*, 2000.

[4] G. Myers, *The art of software testing* (John Wiley & Sons, 1979).

[5] S. Pimont, J.C. Rault, An approach towards reliable Software. *Proceedings of the 4th International Conference on Software Engineering*, Munich, Germany, 1979, pp.220-230.

[6] G.Y. Tian, D. Taylor, Design and Implementation of a Web-based Distributed Collaborative Design Environment, *IEEE Fifth International Conference on Information Visualisation*, London, UK, 2001, pp. 703-707.

[7] G. Bafoutsou, G. Mentzas, A Comparative Analysis of Web-based Collaborative Systems, Database and Expert Systems Applications, *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*, 2001, pp. 496-500.

[8] D. Harel, Statecharts: a visual formalism for complex systems, *Science of Computer Programming*, Vol.8., 1987, pp. 237-274.

[9] D. Harel, A. Pnueli, J. Schmidt, R. Sherman, On the formal semantics of Statecharts, *IEEE Symposium on Logic in Computer Science*, Ithaca, USA, 1987.

[10] D. Harel, M. Politi, *Modeling Reactive Systems with Statecharts: the Statemate Approach* (McGraw-Hill, USA, 1998).

[11] D. Harel, A. Naamad, The STATEMATE Semantics of Statecharts, *ACM Transactions on Software Engineering*, 5(4), 1996, pp. 293-333.

[12] N.L. Vijaykumar, S.V. Carvalho, V. Abdurahiman, On proposing Statecharts to specify Performance Models, *International Transactions in Operational Research*, 9(3), 2002, pp. 321-336.

[13] N.L. Vijaykumar, S.V. Carvalho, V. Abdurahiman, Introducing probabilities in Statecharts to specify reactive systems for Performance Analysis. *Computers & Operation Research*, 33(8), 2006, pp. 2369-2386.

[14] A.S.M.S. Amaral, R.R. Veloso, N.L. Vijaykumar, C.R.L. Francês, E. Oliveira, On proposing a Markup Language for Statecharts to be used in Performance Evaluation, *International Journal of Computational Intelligence*, 1(3), 2004, pp. 260-265.

[15] W3C, Extensible Markup Language (XML), 2002. http://www.w3.org/XML/Activity.

[16] National Institute for Space Research (INPE), EXP-OBDH Communication Protocol Definition: a case study for PLAVIS. São José dos Campos: INPE/QSEE Project, 1998, p. 9 (INPE Internal Publication/QSEE Project).

[17] V. Santiago, A.S.M. Amaral, N.L. Vijaykumar, M.F. Mattiello-Francisco, E. Martins, O.C. Lopes, A Practical Approach for Automated Test Case Generation using Statecharts, *Second International Workshop on Testing and Quality Assurance for Component-Based Systems (TQACBS 2006) in the IEEE International Computer Software and Applications Conference (COMPSAC 2006)*, 17 – 21 de September/2006, Chicago, EUA– Vol. II, pp. 183- 188.