PERPETUAL TESTING

University of Massachusetts

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. E097

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.
This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2003-197 has been reviewed and is approved for publication.

APPROVED: /s/ DEBORAH A. CERINO
Project Engineer

FOR THE DIRECTOR: /s/ JAMES A. COLLINS, Acting Chief
Information Technology Division
Information Directorate
**REPORT DOCUMENTATION PAGE**

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

<table>
<thead>
<tr>
<th>1. AGENCY USE ONLY (Leave blank)</th>
<th>2. REPORT DATE</th>
<th>3. REPORT TYPE AND DATES COVERED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUGUST 2003</td>
<td>Final Feb 97 – Dec 01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. TITLE AND SUBTITLE</th>
<th>5. FUNDING NUMBERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERPETUAL TESTING</td>
<td>C - F30602-97-Z-0032</td>
</tr>
<tr>
<td></td>
<td>PE - 63760E</td>
</tr>
<tr>
<td></td>
<td>PR - E097</td>
</tr>
<tr>
<td></td>
<td>TA - 01</td>
</tr>
<tr>
<td></td>
<td>WU - 01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6. AUTHOR(S)</th>
<th>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leon J. Osterwell and Lori A. Clarke</td>
<td>University of Massachusetts</td>
</tr>
<tr>
<td></td>
<td>140 Governors Drive</td>
</tr>
<tr>
<td></td>
<td>Amherst Massachusetts 01003-4610</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8. PERFORMING ORGANIZATION REPORT NUMBER</th>
<th>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>Air Force Research Laboratory/IFTB</td>
</tr>
<tr>
<td></td>
<td>525 Brooks Road</td>
</tr>
<tr>
<td></td>
<td>Rome New York 13441-4505</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>10. SPONSORING / MONITORING AGENCY REPORT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRL-IF-RS-TR-2003-197</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>11. SUPPLEMENTARY NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRL Project Engineer: Deborah A. Cerino/IFTB/(315) 330-1445/ <a href="mailto:Deborah.Cerino@rl.af.mil">Deborah.Cerino@rl.af.mil</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>12a. DISTRIBUTION / AVAILABILITY STATEMENT</th>
<th>12b. DISTRIBUTION CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13. ABSTRACT (Maximum 200 Words)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The University of Massachusetts participated with Purdue University and the University of California at Irvine (under separate contractual efforts) in a teamed research effort to advance the state-of-the-art in software testing. The University of Massachusetts has collaborated with its research partners on developing improved data flow analysis capabilities and on the development of integrated testing strategies that synergistically combine static and dynamic approaches. This effort is part of the DARPA Evolutionary Design of Complex Software (EDCS) Program.</td>
</tr>
<tr>
<td>The EDCS project made important progress along numerous fronts. Research activity was centered on: Finite State Verification Software Process Technology Self-Adaptive Software Distributed Object Technology</td>
</tr>
<tr>
<td>In addition, there was considerable activity aimed at dissemination, publication, and student training.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>14. SUBJECT TERMS</th>
<th>15. NUMBER OF PAGES</th>
<th>16. PRICE CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Testing, Static and Dynamic Analysis, Software Process Language</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>17. SECURITY CLASSIFICATION OF REPORT</th>
<th>18. SECURITY CLASSIFICATION OF THIS PAGE</th>
<th>19. SECURITY CLASSIFICATION OF ABSTRACT</th>
<th>20. LIMITATION OF ABSTRACT</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNCLASSIFIED</td>
<td>UNCLASSIFIED</td>
<td>UNCLASSIFIED</td>
<td>UL</td>
</tr>
</tbody>
</table>

NSN 7540-01-280-5500
Table of Contents

RESEARCH ACTIVITIES: .................................................................................................................. 1
  Finite state verification: .............................................................................................................. 1
  Software Process Technology: .................................................................................................. 2
  Language Features ..................................................................................................................... 3
  Self Adaptive Software: .......................................................................................................... 4
  Distributed Object Technology: ................................................................................................ 4
DISSEMINATION, PUBLICATION, AND STUDENT TRAINING: ............................................. 4
Publications: ............................................................................................................................... 4
The University of Massachusetts participated with Purdue University and the University of California at Irvine (under separate contractual efforts) in a teamed research effort to advance the state-of-the-art in software testing. The University of Massachusetts has collaborated with its research partners on developing improved data flow analysis capabilities and on the development of integrated testing strategies that synergistically combine static and dynamic approaches. This effort is part of the DARPA Evolutionary Design of Complex Software (EDCS) Program.

The EDCS project made important progress along numerous fronts. Research activity was centered on work on:
• Finite state verification
• Software process technology
• Self adaptive software
• Distributed object technology

In addition, there was considerable activity aimed at dissemination, publication, and student training.

We first briefly summarize the research accomplishments, and then list the many dissemination activities wherein can be found details of our accomplishments.

**RESEARCH ACTIVITIES:**

**Finite state verification:**
On this project we made important progress in our work with finite state verification. Although we continued our work on applying our Flow Analysis for Verification of Systems (FLAVERS) data flow analysis system to Ada software, towards the end of the project, our emphasis and focus shifted to applying FLAVERS to Java software.

FLAVERS is a flexible, powerful system for automatically guaranteeing the absence or detecting the presence of a wide range of user-specified properties or behaviors in both sequential and concurrent systems. FLAVERS complements traditional testing approaches, which only demonstrate the presence or absence of errors for the specific test cases that have been executed. It also complements formal verification methods, which employ more comprehensive analysis, but require extensive expertise on the part of the user. FLAVERS is built using the FLAVERS toolkit, a flexible architecture and component library that facilitates the development of FLAVERS analyzers. The toolkit allows FLAVERS to be extended with additional specialized data flow analyzers and to be applied to a wide range of languages and systems. FLAVERS has been applied to the analysis of Ada and C++ and Java is underway. It has also been used for the analysis of network protocols and architecture specifications.

Work on the Ada version of FLAVERS had been aimed at continuing to speed the system up and improve the system’s handling of more features of the language. A great deal of emphasis was also placed throughout the project upon usability. There was considerable effort devoted to developing a graphical user interface. It resulted in the development of a user interface that
makes FLAVERS far more accessible to wider classes of non-expert users. This work was intended to facilitate the transition of this technology to foster adoption and the consequent ability to evaluate this technology more effectively and thoroughly.

During the latter stages of the project we also developed a process-driven user interface to FLAVERS. This work used the Little-JIL process programming language and the Juliette interpreter to create a system capable of guiding users through the use of FLAVERS by orchestrating the appearance of appropriate windows from the graphical user interface, and the graying out of buttons to prevent users from attempting to use FLAVERS in inappropriate ways. This work can be seen as a continuation of long-standing research in how to make integrated toolsets and environments more effective and better integrated. Thus, this EDCS project has made an important contribution to this area of research which is increasingly being understood and appreciated by the practitioner community in view of its growing interest in integrated programming environments.

In the final stages of this project, a great deal of effort was devoted to continuing the development of a Java version of FLAVERS. This work had been initiated earlier in the project, but continued on through prototype system development during the final year. Much activity involved learning about and interfacing to Java language front end services provided by our colleagues at Kansas State University and McGill University. In addition, we continued to develop our Java concurrency models, and created prototype implementations of analysis capabilities for Java. We also worked to converge the internal architectures of our Ada and Java analyzers, moving us strongly in the direction of being able to analyze systems of mixed Ada and Java code.

Software Process Technology:
Little-JIL is a graphical language developed by the University of Massachusetts for defining processes that coordinate the activities of autonomous agents and their use of resources during the performance of a task. Little-JIL programs are executable so that agents can be guided through a process while ensuring that their actions adhere to the process. Little-JIL programs are also statically analyzable to ensure that reliability requirements are satisfied for all executions of the process.

Flexible and adaptive, a Little-JIL process program defines a variety of ways of accomplishing tasks that can work with varying resource requirements and varying agent capabilities. Agents may be human or automated (software or robots, for example). The choice of particular techniques for a particular context can be made automatically based on resource availability or left up to intelligent agents. Thus, Little-JIL process programs need not tightly control the behavior of agents, but rather guide them through the maze of alternatives and facilitate their communication and resource sharing.

Semantically rich, the Little-JIL language provides features that allow proactive control flow as well as the ability to react to error situations and external events. Pre- and post-requisites are used to dynamically verify that the process is being applied correctly. Resources are defined
using a rich resource model and are reserved and locked during the execution of a process. An agenda manager provides communication with the agents using a graphical user interface for human agents and an API for automated agents.

Language Features

- Task-centered semantics that support multi-agent coordination
- High-level proactive control constructs allow scheduling and drive execution forward
- Reactions support event-driven processes
- Powerful exception handling for recovery from failures
- Pre and post-requisites help to detect and manage process deviations
- Resource modelling and management guides and constrains execution
- Information flow represents communication between tasks
- Visual notation facilitates understandibility and conciseness

On this project we spent considerable effort on improving the Little-JIL language and the Juliette interpreter for Little-JIL. These activities were strongly motivated by considerable experimentation with the application of the language to diverse types of process areas. We wrote Little-JIL process programs in such areas as electronic commerce, data mining, robotics, and intelligent vehicles. Our electronic commerce processes centered on processes for supervising auctions. It is particularly noteworthy that this work entailed collaboration with the finite state verification work, in that the FLAVERS data flow analysis system was used to analyze auction processes written in Little-JIL. This was an important demonstration of the way in which process programs can be analyzed for adherence to key process properties.

As a result of these experimental applications of Little-JIL we made some significant improvements to the language. We clarified our notion of “resource bounded iteration”, an idiom that had been in common use since the language was first conceptualized, adding small but important improvements to the specification of child steps, and clarifying the semantics of our exception management.

Our research on this project also emphasized investigation of the problem of specifying resources. We developed an initial prototype resource specification and management system. Our early experiences with it indicated several shortcomings of this system. As a consequence we designed and implemented an improved version of this system. This early research suggests that resource specification and management is a difficult problem, with great importance to process definition and execution, but it is a problem that has not received adequate attention from the research community.

On this project we also completed the implementation of an early version of the Juliette interpreter. We demonstrated this system at the International Conference on Software Engineering. This demonstration was centered on showing how Little-JIL and Juliette could be used to organize the FLAVERS toolset. This project was referred to in the previous section. From the perspective of the software process technology effort this project should be viewed as a demonstration of how software process technology has matured to the point where it can be used.
to implement a process-centered environment. This environment addresses a portion of the software verification phase of development.

**Self Adaptive Software:**
On this project we also worked on robot team coordination. Much of the work was aimed at understanding the notion of a containment unit. These software entities in this project have the important role of assuming responsibility for specific forms of failure resistance. We designed a number of containment units, and began implementation of some of them. These implementation efforts led to more specific understandings of the nature and role of containment units. This in turn improved our understanding of the nature of self-adaptation. One of the significant outcomes of this project has been the identification of the importance and difficulty of self-adaptation in software. It appears that our process and verification technologies can have considerable importance in studying this problem area.

**Distributed Object Technology:**
Another key accomplishment of this project was the development of our Grapevine agenda management system, concluding with the completion of a Ph.D. thesis on this topic. The system was developed as a prototype. Evaluation of this prototype was by means of its application to the generation of an agenda management system for the coordination of the components of the Juliette interpreter, and a system for coordination of a laboratory meeting scheduling. These evaluations strongly indicate that further development and refinement of our agenda management ideas is indicated.

**DISSEMINATION, PUBLICATION, AND STUDENT TRAINING:**
This section summarizes activities in these areas.

During this project our personnel wrote many papers and technical reports listed below. We are gratified that many of these appeared in the most prestigious publication venues, such as the International Conference on Software Engineering and the Foundations of Software Engineering Conference.

In addition, project personnel gave numerous presentations around the country and the world. A partial listing is given here.

Finally we note that PhD students, Eric McCall and Gleb Naumovich completed their Ph.D. degrees through support of this project, and that several Masters students also completed their degrees.

**Publications:**


An Evaluation of Object Management Systems Architectures for Software Engineering Applications, Jayavel Shanmugasundaram, Barbara Staudt Lerner and
Lori A. Clarke, Computer Science Department, University of Massachusetts, Amherst, MA, Technical Report 97-47.


Specifying Coordination in Processes Using Little-JIL, Leon Osterweil, Barbara Staudt Lerner, Stan Sutton, Sandy Wise, and Eric McCall, Computer Science Department, University of Massachusetts, Amherst, MA, Technical Report 98-38.


Modeling Resources for Activity Coordination and Scheduling, Rodion Podorozhny, Barbara Staudt Lerner and Leon J. Osterweil, Proceedings of International Conference on Coordination Models and Languages, April 1999, pp. 307-322. Springer Verlag Lecture Notes in Computer Science #1594.


Fast Generation of Dynamic and Multi-Resolution 360° Panorama from Video Sequences, Zhigang Zhu, Guangyou Xu, Edward M. Riseman and Allen R. Hanson,


Exploiting Hierarchy for Planning and Scheduling, Aaron G. Cass, Krithi Ramamritham and Leon J. Osterweil. Computer Science Department, University of Massachusetts, Amherst, MA, Technical Report 00-64, December 2000.


