Group: Verification Approaches for AO Programs

[1]Guillaume Pothier [2]Paulo Masiero [3]Roberta Coelho

[1]Pleiad Lab - Computer Science Department University of Chile gpothier@dcc.uchile.cl

[2] Computer Systems Department ICMC-USP masiero@icmc.usp.br

[3] Informatics Department PUC-Rio souzacoelho@gmail.com

Group Goals

Investigating the impact of verification tools and approaches on the reliability and verification time of AO programs; and proposing tools and approaches to support the verification process.

Keywords Testing, static analysis, debugging, fault model, exception handling.

1. Problem

The new constructs available in aspect-oriented languages represent new sources of faults, in particular when considering the impact of aspects on the exception flow of programs. Such faults bring new challenges to software testing, static analysis and debugging.

During the verification process of an AspectJ system, the developers usually apply ad hoc approaches to check the reliability of the program, such as: writing and executing functional tests, adding logging statements, or performing *smoke* testing (i.e., they execute the application and uses it until a failure occurs).

Some tools and approaches have been proposed that aim at detecting and diagnosing specific faults in AO programs, but no empirical study was conducted to asses their effectiveness.

2. Proposed Approach

Our approach to tackle this problem is twofold: (i) to conduct an experimental study to assess the effectiveness of a set of tools proposed by our group, and (ii) to evaluate the opportunities of tool integration.

The experimental study is going to be our first short/medium-term goal. This study will be conducted in the context of classes at the University of Sao Paulo/Sao Carlos and hopefully at other universities. It will involve a group of Masters and PhD students that will perform a set of verification tasks according to the experiment's plan.

In parallel we are going to evaluate the possibilities of integrating the three tools. Firstly, Roberta and Guillaume will work on integrating the SAFE tool into Eclipse, potentially reusing the data base structure and some GUI components of

TOD - it will avoid a bias caused by the non-user-friendly textual output of SAFE. Secondly, we are going to investigate the use of the set of tools (i.e., TOD, SAFE and JABUTi) through a web interface. A third step would be the integration of the data structures of the three tools, as a way of enabling a synergy between static analysis, structural testing and debugging. For instance, the developer could navigate through the data collected during structural tests and static analysis while debugging.

3. Goals

Short Term Goals - Develop an Eclipse plug-in for SAFE based on the infrastructure proposed by TOD.

Medium Term Goals - Conduct an experimental study to asses the effectiveness.

Long Term Goals - Integrate TOD, SAFE and JABUTi.

4. Plan

This section contains the set of tasks that will be performed by this group in the context of LatinAOSD project:

Experimental Study Tasks

S1 – Build SAFE plug-in.
S2 – Study an example of an experimental study.
S3 – Prepare the experimental study plan, which describes what will be done during the experiment.
S4 – Prepare the package, which contains all the artifacts (i.e., forms, tools, training material, instructions, *programs to be tested*, errors seeded) needed to conduct the experiment[1].
S5 – Conduct a pilot study to asses the experiment's package.
S6 – Update the package to address the problems discovered during the pilot study.
S7 – Conduct the experiment.
S8 – Analyze the results and write a report.
S9 – Write a paper about the study.

Tasks	Jun/08	Jul/08	Aug/08	Sept/08	Oct/08	Nov/08	Dez/08	Jan/09	Feb/09	Mar/09
S1	G	G	G							
S2	M	M	M							
S3		M	M							
S4			R	R						
S5					M					
S6					R					
S7						M	M			
S8								R	R	
S9										All of us

[1] We need to evaluate how JABUTi/AJ can be used to show whether an *exception path* (i.e., path in the call graph that links the method that signals the exceptions and the method that handles it) was exercised or not during an structural test.

Tool Integration Tasks

T1 – Study how to enable a light integration of the tools (e. g., through a unified process of verification that encompass the three tools, or a web interface that contains links to every tool).
T2 – Study how to widen the data model of TOD in order to integrate data from other sources (e.g., static analysis data computed by SAFE, or structural testing related information calculated by JABUTi). This task may include interactions between the developers of the three tools.
T3 – Define a detailed plan for the next tasks.

Tasks	Jun/08	Jul/08	Aug/08	Sept/08	Oct/08	Nov/08	Dez/08	Jan/09	Feb/09	Mar/09
T1				G						
T2					G	G	G			
T3						G	G			

5. People Involved

- Marcos L. Chain – He is a professor at USP who works with debugging. He could interact with Guillaume on experiments and improvements of TOD debugging tool.
- Otavio Lemos – PhD student at USP/Sao Carlos.
- Rodrigo Gondim – Master student at USP/Sao Carlos.
- Vânia de O. Neves – Master student at USP/Sao Carlos that will extend JABUTi tool to support integration testing of AO programs.
- Paulo Masiero – Professor at USP/Sao Carlos.
- Guillaume – PhD student at the University of Chile.
- Roberta – PhD student at PUC-Rio.

6. Requests for Students Interchanging

It would be interesting to have some participants of the group meeting next semester. Otavio Lemos a PhD student at USP/Sao Carlos, one of the main developers of JABUTi/AJ, could interact with Guillaume to asses the difficulties of integrating the two tools, and prepare an integration plan for doing it – in case they conclude such task would be feasible in a reasonable amount of time. Moreover, Guillaume could go to Brazil and spend some time working with Roberta and/or Otavio.

7. References

[1] G. Pothier and É. Tanter. Extending Omniscient Debugging to Support Aspect-Oriented Programming. In Proceedings of the 23rd ACM Symposium on Applied Computing (SAC 2008), volume 1, pages 266--270, Fortaleza, Ceará, Brazil, Mar. 2008.

[2] G. Pothier, É. Tanter, and J. Piquer. Scalable Omniscient Debugging. In Proceedings of the 22nd ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA 2007), pages 535--552, Montreal, Canada, Oct. 2007. ACM Press. ACM SIGPLAN Notices , 42(10).
[3] Coelho, R. et al; Assessing the Impact of Aspects on Exception Flows: An Exploratory Study , European Conference on Object Oriented Programming (ECOOP 2008).
[4] Lemos, O. A. L. et al. Control and Data Flow Structural Testing Criteria for Aspect-Oriented Programs. Journal of Systems and Software, doi:10.1016/j.jss.2006.08.02, v. 80, p. 862-882, 2006.
[5] Wohlin, C. et al., Experimentation in Software Engineering - An Introduction. Kluwer, 2000.
[6] Ernst, M. D. Static and dynamic analysis: Synergy and duality. WODA 2003: ICSE Workshop on Dynamic Analysis, Portland, OR, 2003, 24-27
[7] Chaim, M.L.; Maldonado, J.C; Jino, M. A Debugging Strategy Based on Requirements Testing, Journal of Software Maintenance and Evolution: Research and Practice. Vol 00, 2004.
[8] Chaim, M.L. et all. Simulating the Execution of Instrumented Programs, Submitted to SBES, 2008.