



1st Workshop on Assessment of Aspect-Oriented Technologies ASAT.07

- An Informal Workshop Report -

Contributors: Alessandro Garcia, Sergio Soares, Marc Eaddy, Marc Bartsch, Mehmet Aksit

This report is focused on describing the main points of discussion at the ASAT workshop. The people that joined (fully or partially) the workshop were: *Stefan Hanenberg, Marc Bartsch, Arjun Singh, Marc Eaddy, Sergio Soares, Alessandro Garcia, Mehmet Aksit, Ruzanna Chitchyan, Dimitri van Landuyt, Ana Moreira, Awais Rashid, Karl Libeherer, Claudio Santanna, and Americo Sampaio*. The report is organized in the following parts. First, it presents a summary of the workshop discussions during paper presentations. Second, it summarizes in a nutshell the main points that emerged from all the three paper sessions, and shaped the discussions in the afternoon. Third, it reports the main points debated in the discussion group.

1. Discussions in Paper Presentation Sessions

Session I – Empirical Studies

Chair: Sergio Soares (UPE, Brazil)

The 1st ASAT technical session had 3 interesting papers that addressed empirical studies and assessment mechanisms from different perspectives in the software lifecycle – requirements engineering, architecture design, and detailed design:

On the Modularity Assessment of Software Architectures: A Concern-Oriented Approach

Claudio Sant'Anna, Alessandro Garcia (Lancaster University -UK)

Discussant: Marc Eaddy (Columbia University, USA)

Comparing Aspect-Oriented Requirements Engineering Approaches: An Empirical Study

Americo Sampaio, Phil Greenwood, Alessandro Garcia, Awais Rashid (Lancaster University -UK)

Discussant: Dimitri van Landuyt (KULeuven – Belgium)

Comparing Aspect-Oriented and Component-Based Design: A Quantitative Study

Dimitri Van Landuyt, Steven Op de beeck, Johan Gregoire, Riccardo Scandariato, Wouter Joosen, Andrew Jackson, Siobhan Clarke (KULeuven - Belgium, Trinity College Dublin - Ireland)

Discussant: Americo Sampaio (Lancaster University -UK)

The first paper focused on the modularity assessment of software architectures. There were mainly two discussions topics. One is related to whether the traditional modularity definition is valid to AO systems. So there was a brief discussion if we should evaluate the need to enrich the modularity definition to the AO paradigm. Another topic was related to the challenges associated with empirical studies in software engineering, especially in fields

where there are not yet well-established AO techniques and definitions, such as in architecture design. If there is no strong consensus on what we mean by “architectural aspects”, how can somebody develop empirical studies to assess the benefits and drawbacks of AOSD in this field.

The next paper performed a comparison between requirements engineering approaches. Once again the concern of how designing/planning rigorous experiment in software engineering was raised. One of the comments was that the exercise of planning the experiment might also help the technique’s proponent to better understand his proposal. A common concern was how avoiding bias in such experiments. For instance, by just looking to the time to execute some requirements engineering tasks we might not be able to state if a requirement technique is better than other. A greater effort in such activities might decrease the effort in the next tasks/disciplines (such as trade-off analysis and architecture design), which in fact might be more productive. This, of course, requires a broader and systematic evaluation, taking into account the complete software development cycle. Another comment related to that is the need for empirical evidence that relies on deployed systems that had been changed over some time, therefore showing real development environments and realistic examples of requirements evolution requests.

In the last session paper once more the main topic was empirical studies. Several workshop participants raised the issue that empirical evidence based on industrial-strength applications is crucial to convince the industry to adopt new AO techniques, and that there is a lack of such experiments in the AOSD field.

Session II – Assessment Techniques

Chair: Marc Eaddy (Columbia University, USA)

The 2nd ASAT technical session had 3 thought-provoking papers that addressed respectively validation and development of complementary means to assess AO software artifacts and techniques, such as metrics, testbeds, and fuzzy-logic mechanisms:

Towards an Empirical Validation of Aspect-Oriented Coupling Measures

Marc Bartsch, Rachel Harrison (University of Reading - UK, Stratton Edge Consulting - UK)

Discussant: Marc Eaddy (Columbia University, USA)

Dealing with Imperfect Information in the Assessment of Aspect-Oriented Techniques

Joost Noppen, Pim van den Broek, Mehmet Aksit (University of Twente - The Netherlands)

Discussant: Marc Bartsch (University of Reading, UK)

On the Design of an End-to-End AOSD Testbed for Software Stability

Phil Greenwood, Alessandro Garcia, Thiago Bartolomei, Sergio Soares, Paulo Borba, Awais Rashid (Lancaster University - UK University of Waterloo - Canada Pernambuco State University - Brazil Federal University of Pernambuco - Brazil)

Discussant: Mehmet Aksit (University of Twente, The Netherlands)

First, Marc Bartsch presented the results of trying to correlate aspect-oriented coupling with software development effort. He introduced a suite of coupling metrics for aspect-oriented programs. He found a weak correlation between his metrics and effort. Indeed, module

size was more strongly correlated with effort. Marc believes the weak correlation may be due to his metrics not capturing all the coupling present, and is working on improving the metrics. It is significant that Marc was the first to try correlating concepts from aspect-oriented programming with external quality indicators such as effort. It is clear from his talk that a better understanding of aspect-oriented coupling is needed.

Mehmet Aksit presented the results of applying fuzzy logic to evaluating different software designs and modularization alternatives. Program designers must use imperfect information when making design decisions. By modeling the possible design alternatives using probability theory and fuzzy logic, we can estimate the quality of design decisions. Evaluating the quality of a design decision based on imperfect information appears to be similar to evaluating it based on "net option value," and a comparison of the two approaches may be useful.

A major obstacle to evaluating AOSD is the lack of available benchmarks with which researchers can compare modularization approaches, evaluation methodologies, and metrics, and verify that experimental results are repeatable. Garcia presented a roadmap for designing a testbed (set of benchmarks) for evaluating AOSD. The first benchmark created was for the HealthWatcher application. To allow comparison along multiple dimensions, the benchmark includes implementations in multiple languages (OO and AO), designs and architectures using multiple modeling languages (OO and AO), and multiple metrics both traditional and aspect-oriented systems.

Session III – Concern Metrics

Chair: Marc Bartsch (University of Reading, UK)

Towards Assessing the Impact of Crosscutting Concerns on Modularity

Marc Eaddy and Alfred Aho (Columbia University - USA)

Discussant: Marc Bartsch (University of Reading, UK)

Marc Eaddy presented a suite of concern-driven metrics which measure the distribution and separation of concerns in a software program. These metrics are based on the idea that all concerns have been identified and related to their specific code portions. Marc supports a formal requirements specification to determine all concerns. The discussion that followed the presentation included questions about the definition of “crosscutting”. The point was made that “crosscutting” may need to be defined by scattering and tangling and not only by scattering. This interesting discussion should be continued to further elaborate on fundamental concepts within the AOSD community.

2. Some Points for Further Reflection and Research

There were a number of issues that seemed to systematically pervade all the paper sessions. Then, a number of concerns shared by the workshop participants seem to require more careful attention on the assessment of AOSD techniques. In a nutshell, some findings and comments during the discussions after paper presentations were:

- the definition of modularity is not clear; each work seems to emphasize an important dimension of modularity, but it is currently difficult to give a precise definition and a big picture on what modularity means.
- there are some semantic dependencies or interferences that also need to be captured and well understood during an assessment process; they do not seem to be measured by the body of existing metrics nowadays. More fundamentally, are metrics the appropriate mechanisms to deal with such semantic dependencies?
- one of the difficulties in both developing empirical studies and proposing new metrics in AOSD arises from the fact that there is a mutual dependency that impedes us to evolve fast: (i) in order to develop well-accepted studies, it is required strongly-validated metrics which are not currently available, and (ii) in order to validate the metrics, we need to apply them to a huge number of “well-accepted studies” in order to produce statistically-relevant data.
- the design of rigorous empirical studies in AOSD is a challenge due to several reasons, including: (i) it is difficult to rely on industrial-strength case studies, (ii) we do not have accessible infrastructure for performing controlled experiments and validating assessment mechanisms (such as metrics), (iii) it is challenging to come up with convincing and validated metrics/indicators that support the comparison of AOSD and other paradigms, and (iv) while comparing AO and non-AO techniques, how to guarantee that AO and non-AO decompositions are good (if not the best) ones.
- Different software engineering phases in AOSD require specific metrics/indicators, which might not be ready for use, such as in requirements engineering. An alternative is to rely on (or extend) metrics/indicators being consistently used in that particular community (e.g. “precision” and “recall” in requirements engineering).

3. Discussion Group

Based on the paper presentation sessions in the morning, three generic topics were selected as theme candidates for the discussion groups in the afternoon:

- Which are generic steps on performing empirical studies concerned with comparisons of AO vs. non-AO techniques? Which lessons we could learn from the several studies presented today and/or available in the community?
- How to enhance the state-of-the-art on assessment of AOSD techniques?
- Are conventional metrics enough to assess AOSD artifacts?

The workshop participants felt that it was a better choice to have only one group discussing the first topic above rather than split the attendees in two or more groups. The people that participated in the discussions were: Stefan Hanenberg, Mehmet Aksit, Marc Bartsch, Arjun Singh, Marc Eaddy, Sergio Soares, Ruzanna Chitchyan, Dimitri Van Landuyt, and Alessandro Garcia. In short, the discussions were around the following points:

- First, Mehmet Aksit reported his group’s experience on developing a case study involving the application of Compose* and AOP techniques in cooperation with an industrial partner. In such a project, they were interested in measuring time effort; reliability of measures especially plays an important role in this context. In order to

do that, they have selected small set of changes in particular components which could be accomplished in a few hours (3-6 hours), so that the time spent on such tasks could be more effectively controlled. In addition to measuring time, the number of remaining bugs was also observed. One of the study outcomes was that the industrial partner is going to adopt AOP only after Mehmet's group convince that they are able to deal properly with issues such as semantic concern dependencies (and interferences). The company is currently paying around 8 people (in addition to the people employed by the company) for carrying out new case studies encompassing Compose* and other AOP mechanisms, such as C weaver, Statechart integration, and Compose* in-ling.

- Marc Bartsch mentioned that the size of the tasks, under assessment/measurement in controlled experiments, can not be very complex but not very simple; identify the correct scale of complexity is not always trivial.
- It is difficult to develop rigorous studies involving new paradigms, such as aspect-orientation. In addition to issues discussed in the morning, participants reported that: (i) it is difficult to find data for aspect-oriented systems, and (ii) there is missing knowledge on how to perform statistics analysis even though AOSD is in an age now that requires meaningful statistically-relevant data in order to facilitate technology transfer. The creation of testbeds and benchmarks in the community are interesting initiatives to address this gap.
- How to minimize bias when comparing OO and AO solutions? Up to now, many of the empirical studies rely on refactoring Java implementations with AOP languages, which might be a problem. Dimitri suggested that try to not impose a base solution as starting point might be a better idea.
- Stefan Hanenberg pointed out that is really difficult to identify in the experimentation literature (even in social sciences books) effective guidelines on how to pre-select subjects for participation in experiments. Then, as far as selection of subjects is concerned, Bartsch also commented about the possibility of selecting subjects from specialized emailing lists, such as the AspectJ one, based on the topics they have posted. The well-know preferred strategy of random choices is practicable only if you have a huge amount of subjects at hand, which is uncommon.
- The issue of using students in experiments was also discussed. Is it possible to construct "valid" studies using undergraduate and postgraduate Computer Science students? Alessandro Garcia mentioned the outcomes of a paper from the Empirical Software Engineering literature ("A Survey of Controlled Experiments in Software Engineering" by Sjøberg et al, IEEE TSE, Sept 2005), where the authors have reported that from 113 experiments described in widely-recognized Software Engineering journals, around 5,400 subjects took part in the experiments, from which 87% were students and 9% were professionals. Then, Alessandro described his experience on developing two instances of the same empirical study at Lancaster - one with undergraduate students and other with Masters' students (during an AOSD module). The goal of this study was to assess and compare the accuracy of three suites of heuristic rules to identify bad smells in AO design. Not surprisingly, the data outcomes from the two study instances presented some significant

differences. However, the studies served to test some initial hypothesis, and as pilot studies for planning better other empirical studies in the future, in addition to be interesting a vehicle for educational purposes. The selection and distribution of the students in groups was based on: (i) the use of questionnaire previously distributed to the potential subjects, (ii) marks of students in previous years (e.g. Java Programming), and (iii) observations about their performance on the first days in the practical sessions in the lab.

- Alessandro mentioned that studies presented in the first paper session seem to indicate that one of the main bottlenecks for AO techniques is the complexity associated with “composition specifications”. For instance, Americo’s study was very incisive in that respect. Then, a question was formulated: is it influenced by the fact that people are not used to specify such AO compositions? or is it an intrinsic paradigm problem? There is definitely the need for more work in this area for enable us to understand this issue. However, Ruzanna Chitchyan mentioned that, from her perspective, AO makes dependencies more explicit, and such composition complexities are inherent part of the problem; for her, aspect-orientation provides interesting abstractions for express certain concepts. A lively discussion happened at this point. From the discussion, it seems that a final answer depends a lot on the level the developer define the compositions or specific aspect-oriented technologies we are using. Some participants mentioned that composition languages, such in AspectJ, do not seem to provide a good feeling that we have intuitive composition descriptions and that they accelerate the software development process. Pointcut specifications are a bottleneck and aspect interference is always a problem – presence of loops, for example.