ASE 2008 Review

Reviewer	
Last update	Mon Jun 2 17:53:12 2008 +0200 CEST
Paper Number	34
Title	
Authors	

1 - Classification as long paper: C

2 - Classification as short paper: *[Not filled in]*

3 - Reviewer's Expertise: X

4 - Relevance: Significantly Relevant

5 - Comments for the PC (incl. reason for classification as short paper):

[Not filled in]

6 - Summary:

The paper presents a formalization of an approach to the specification of automatic, on-site, and bidirectional synchronizers that can be used for (re)establishing consistency among related software artifacts without exporting and importing them to and from an intermediate representation.

7 - Evaluation, including points in favour and against, and comments for improvement:

Overview

The concept and approach are very interesting and seem like a promising approach. However, I have no way of judging the generality of the approach and its properties since the evaluation is inadequate. The evaluation focuses only on performance and ignores important issues outlined in general comments below. The approach is implemented and applied to a single toy example (and the implementation seems to be correct because the experiments were executed; however, authors do not say whether the artifacts were correctly synchronized).

The paper introduces a lot of notations that are used in the formalization making it very difficult to read and understand. I cannot determine that the formalization is correct, since many things are not properly explained and I think there are some errors in the formalization.

The idea of the synchronizer graphs is interesting.

To summarize, the presented ideas are interesting and promising; however, the evaluation is inadequate which prevents me from accepting the paper with full confidence.

Is the paper in-scope of ASE?
Yes, it proposes an approach to automatic synchronization of software artifacts, which addresses an important problem of consistency management among related software artifacts in software engineering.

Contributions

A compositional approach to the specification of bidirectional synchronizers. Formalization of the approach.

General comments
My biggest problem with this paper is that the rationale is not properly presented. Understanding the formulas is difficult due to the notation and it is even more difficult because the reader does not understand why certain definitions are the way they are. The proper intuitions are not (always) provided. I also think that introducing the entire notation upfront makes the paper more difficult to read. In my opinion the notation should be introduced gradually, before it is first used. This way, the reader can gradually build up an understanding of both the notation and the definitions.

Another problem is that the limitations of the approach are not discussed at all. The performance evaluation, although supporting the claim that incremental on-site synchr. is faster, does not address the most important questions. What kinds of modifications on artifacts are supported? What happens in the case of conflicting modifications (conflict resolution?)? What are the limitations? Is keeping the state of the synchronizer a problem (memory-wise)? Are the synchronizers presented in the paper all that you have? Is the EJB example the only example the approach has been applied to (if so, does it only have the three types?!?).
It is hard to judge the generality of the approach. What are the relations among artifact restricted to (e.g., only bijections or something more?)?

Therefore I conclude that the evaluation of the approach is inadequate.

About Theorem 1. You say that the proof is omitted due to space reasons. I noticed you published a technical report out of this submission on an author's web page. One would expect the tech report to have more detail since it has no space limitations, however, it does not contain the proof either. Do you have the proof or not? If not, you should not state Theorem 1 as a result since it is not confirmed!!! I realize that the formal proof is not a trivial task and maybe a material for a separate paper. However, if the proof was not done it should be mentioned as a future work. I am puzzled.

Overally, the paper is extremely difficult to follow. I think using formal notation should be limited to places where it actually provides benefits. The paper indeed looks more like a technical report rather than a conference paper. As a result, extracting ideas and intuitions from the paper is very difficult. The paper should be first readable entirely without relying on the formal definitions. Interested readers should however be able to delve into the details of formalization. Right now, the paper can only be 'very deeply understood' or 'not understood at all'. And this is not due to the inherent complexity of the presented concepts. I suggest raising the level of abstraction.

Detailed comments
p. 4. why do you introduce {}t here when it is first used in Algorithm 2, ln. 8? (see general comments about notation).

p. 4. propagation? What is s.data? Why? give the rationale. It is not understandable.

p. 4. "initially the synchronizer is in state s.v and

assume all artifacts are empty" - does that mean that existing artifacts cannot be synchronized?

p.5. synchronizer id. "if m1 and m2 are not distinct it returns error". So for example if element a was modified with value v1 in one artifact and with a value v2 in the other artifact (which is a conflict) then a cannot be synchronized. What happens next? What does an error mean? Does the synchronization terminate or can the remaining elements be synchronized? I suppose this is a limitation you should discuss in evaluation.

synchronizer remove. The explanation is inadequate. It does not replace the artifact, but any modification of the artifact into del. I don't understand the first condition: shouldn't that empty modification be replaced with del too?

synchronizer equal. How can you make an entire artifact equal to a primitive value? Why do you pass 'error' as a state to id? I'd say that sync cannot actually 'force' an artifact to be equal to a value because id cannot 'override' operations which are not distinct.

p.5. dget. "State records current values of the three artifacts". What? Is a primitive value (k) also an artifact? What exactly is an artifact for you? Is it any member of U? This is an important assumption that was not stated anywhere.

merge is not understandable. What is the rationale? Why? What does it actually do?

p. 7. "the more choices the synchronizer has to ..." What choices? What is a choice? Options? This part is not understandable. What is a "more determined synchronizer"? Explain.

You say a synchronizer may be invoked again if its variables changed. Are there any guarantees on the termination of the algorithm? Is it possible to build a synchronizer that does not terminate?

p. 7. Dictionary map combinator. R_k is a bijection. Does that imply that the relation between any two artifacts has to be a bijection? Can a single element in one artifact correspond to multiple elements in the other artifact? This limitation should be discussed in evaluation.

p. 8. So resync can be used to synchronize independently created (existing) artifacts by computing the initial state of the synchronizer. Should say so explicitly in evaluation that resync is required for that.

Related work Benjamin et al. -> Pierce et al. Yes, but the common artifact must be the view/abstraction of both artifacts (contain the shared information).

It is not so simple to define a synchronizer for concatenating strings, unless the strings have some special structure.

View consistency. Actually, Harmony was designed for this problem.

Points in favour and against:
+ promising idea, relevant to ASE
- hard to read and understand
- inadequate evaluation, major rewrite required