Reviewing Inconsistent Behavioral Properties

Jennifer Brings and Marian Daun

paluno – The Ruhr Institute for Software Technology, University of Duisburg-Essen, Germany
{jennifer.brings, marian.daun}@paluno.uni-due.de

Context & motivation. During model-based engineering, numerous diagrams are created, some of which document the same behavioral properties of the system from different perspectives (cf. [1]). In industrial practice these diagrams are often developed or revised independently of each other. In consequence, inconsistencies will almost inevitably arise during the development. This situation may, for example, occur in function-centered engineering when just one of the two main artifacts, the behavioral requirements or the functional design, is updated with new or changed stakeholder intentions. While it is possible to automatically detect these inconsistencies, it is impossible to resolve them automatically if it is unclear whether the requirements are up-to-date. Hence, manual reviews are needed to determine the current stakeholder intentions and to ensure their correct representation in requirements and later on in design artifacts.
Question/Problem. Manual reviews, however, tend to be error prone and time consuming (cf. e.g., [2, 3]). To aid manual reviews of model-based specifications and in particular, the review of behavioral requirements and functional design, we proposed the use of a dedicated review model in [4], which represents both artifacts within one model. The review model supports the manual review of both artifacts, in that it transforms the functional design into a notation format that is more amenable to manual review (ITU-Message sequence charts [5]) and in that it identifies consistent parts, thus also reducing the size of the specification to be reviewed. Experiments have shown that the use of the review model increases effectiveness, efficiency, user-confidence, and supportiveness compared to the review of the two original artifacts [4]. However, so far the distinction between consistencies and inconsistencies has only been made for entire diagrams in the review model. This may lead to the review model containing two very similar diagrams, such as those depicted by the separate representation in Fig. 1 .
Principle ideas/result. To further investigate whether an integrated representation of two inconsistent behavioral properties in one single diagram can improve manual reviews even more, we extended our approach to automatically merge two inconsistent basic message sequence charts [5] (see integrated representation in Fig. 1). In addition to a first prototypical implementation, we evaluated this extension in close collaboration with our industry partners using a realistic specification [6]. Furthermore, we repeated the experiment reported in [4] for the two representation formats. While we could not find any significant difference between the two representation formats regarding effectiveness, user confidence, and supportiveness, the integrated representation format (M=.65, σ =.39) proved highly significantly more efficient than the separate representation format (M=1.14, σ =.88), $t(34)$=3.08, p<.01, d=.52, in cases where the

number of inconsistencies was small. For diagrams with a higher number of inconsistent modeling elements, however, the separate representation (M=.65, σ=.27) seems to be more useful than the integrated representation (M=.71, σ=.30), albeit not statistically significantly $t(35)=1.46$, $p>.05$, $d=.25$.

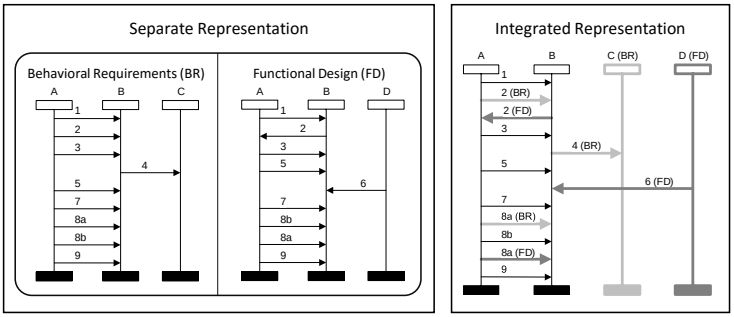

Fig. 1. Separate and Integrated Representation of Inconsistent Behavioral Properties

Outlook & future work. Preliminary results of our investigation show that different representation formats address the review of different degrees of inconsistencies. Hence, some degrees of inconsistencies can most appropriately be reviewed in a single diagram while others should preferably be reviewed in distinct diagrams. However, the distinction between small and large inconsistencies was based on common but trivial guidelines regarding cognitive load for differentiation (i.e. small inconsistencies contained fewer than five differing model elements and large inconsistencies more than nine). In the future we plan a more thorough investigation to establish which representation format is more advantageous in which cases. This will also include the investigation of other factors such as classes of inconsistencies, inconsistency patterns, and other metrics (e.g., difference between two diagrams, complexity of a diagram under review) to establish efficient review concepts for each case.

References

1. Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L., Goedicke, M.: Viewpoints: A Framework for Integrating Multiple Perspectives in System Development. Int. J. Software Eng. Knowl. Eng. 2, 31–57 (1992)
2. Johnson, P.M., Tjahjono, D.: Assessing software review meetings. In: ICSE, pp. 118–127 (1997)
3. Sims, S., Cleaveland, R., Butts, K., Ranville, S.: Automated validation of software models. In: ASE, pp. 91–96 (2001)
4. Daun, M., Weyer, T., Pohl, K.: Detecting and Correcting Outdated Requirements in Function-Centered Engineering of Embedded Systems. In: REFSQ, pp. 65–80 (2015)
5. International Telecommunication Union (ITU): Message Sequence Chart (MSC) (2011)
6. Föcker, F., Houdeck, F., Daun, M., Weyer, T.: Model-Based Engineering of an Automotive Adaptive Exterior Lighting System. ICB Research Report 64 (2015)