Hands-on Workshop on Collaborative Modeling

Hands-on Workshop Protocol

Co-authored by the the Organizing Committee and
the Involved members of the Program Committee

Version: 0.2.
September 8, 2021.

1 Goals

The main goal of the hands-on sessions of the workshop is to infer a unified knowledge of the current state of tool support for collaborative modeling. This is achieved by answering the following research questions.

RQ1) What features are supported in current collaborative modeling tools?

RQ2) What is the level of user appreciation towards these features? Which features should be improved?

RQ3) What missing features would users like to see in collaborative modeling tools?

Typical takeaway of the workshop would read as "*Feature X is supported in tools A, B, C.*", "*Feature Y is not supported in any tool.*", "*Users find the tool support appropriate for carrying out task Z overall.*".

The workshop does **not** aim at comparing and establishing a competition between the tools. However, tool builders are encouraged to highlight as many features as possible during the workshop.

2 Tools

Based on the reviews, the following tools are invited to the workshop.

- MetaEdit+ (*Collaborative modeling and metamodeling with MetaEdit+*)
- MCMC (*MindMaps sans Frontières*)
- TGRL Live Share (*Towards Conflict-Free Collaborative Modelling using VS Code Extensions*)
- Parsafix (*Collaborative Modeling across Language Workbenches - a Case based on Jetbrains MPS and Eclipse Spoofax*)

3 Cases

Cases are developed by the *authors* in order to expose the participants to the collaborative features (Section 3.2) of the tool as much as possible. Cases are composed of a series of tasks (Section 3.3) covering a set of features. Participants experience the features through these tasks.

3.1 Common properties of cases

- The case can be carried out by a user with modeling experience but new to the tool in 20–30 minutes.
- No prior knowledge on the tool must be assumed, apart from the information given right before the experiment. Authors may ask participants to install or get familiar with their tool in advance, but that is not a requirement.
- Timing is as follows:

  00:00-00:05 Introduction to the case and the tool

  00:05-00:25 Execution of tasks

  00:25-00:30 Exit questionnaire

3.2 List of features

The list of features is assembled from the published empirical studies [1–3]. *Authors* will provide the level of support by their tool for each specific feature on a scale of {`full support`, `partial support`, `no support`}. Authors can also add a comment on each cell if they think more information is useful. This information is stored on a Google Drive[1].

PC members and *authors* of case papers are encouraged to suggest changes to this list and add more features.

[1] `https://docs.google.com/spreadsheets/d/1ciozNoC3pxoDM38iRrUkSwHZemrk18H02R0hqGU0TQ0/edit`

3.3 List of tasks

The list of tasks is to be developed by the *authors* of the case papers in accordance with the list of features.

- The list of tasks should fit the time frame of the hands-on session. Consider time for setting up and answering the questionnaire.
- Tasks can cover multiple features and features can be covered by multiple tasks.
- The mapping between tasks and features is recorded in a spreadsheet on Google Drive[2].

Guidelines to good scenarios: [4], pp. 173–177. (Find attached in the workspace.)

3.3.1 Example tasks

Some example tasks are provided below. Suppose the DSL is Petri nets, each user should perform the following tasks:

1. Create X instances of different metamodel elements (e.g., 9 places and transitions). Users should coordinate to decide who creates which elements (e.g., Alice creates 5 places and Bob creates 4 transitions).
2. Create X relations between the elements of your model (e.g., X transitions from/to places/transitions) that any user has created. Like the previous task, a coordination is required between users. However, conflicts may arise if constraints are not satisfied (max outgoing arcs from a place) or one user already created a relation.
3. Modify the values of elements (e.g., set the number of tokens in each place) that any user has created. Like the previous task, coordination is required and inconsistencies may arise due to a constraint violation on attributes (e.g., the number of tokens exceeds the capacity of the place).
4. Integrate models created by different users (e.g., the Petri net model of user 1 should have transitions from the model of user 2 and vice-versa). Having a variant of the task that triggers a conflict would be advisable.
5. Remove elements or change values so that the integrated model makes sense. Situations like in tasks 2–4 may arise but now it is on the integrated model.
6. Change an element in the metamodel and the modelers should correct their model accordingly. This showcases a scenario where the language engineer and the domain expert collaborate.

These can all be applied in a real-time or offline collaboration setting.

When defining tasks, please, adhere to the schema defined in the corresponding spreadsheet.

3.4 Questionnaire

After each session, a questionnaire will be completed by the participants. To ensure compatibility between the tools and experiments, we will use the *exact same* questionnaire for each tool.

The questionnaire is composed of the following questions.

- *Email of the participant* [Optional]
- *Tool used* [Options, one of the four tools, Mandatory]
- *Familiarity with the tool before the hands-on session* [None-Novice-Intermediate-Proficient-Expert, Mandatory]
- 10-15 questions about the tasks/features. [Likert scale, Mandatory]

[2] `https://docs.google.com/spreadsheets/d/1ttfxg-aOTZzuEnEybL3xjeymW6hKpo1hwy94MuLcKHk/edit`

  - Some of the questions are positive, some are negative.
  - The exact questions will be extracted from the final cases late September - early October.

- *(Collaborative) Features to improve* [Free text, Optional]
- *(Collaborative) Missing features* [Free text, Optional]
- *General remarks* [Free text, Optional]

References

[1] M. Franzago, D. Di Ruscio, I. Malavolta, and H. Muccini, "Collaborative model-driven software engineering: a classification framework and a research map," *IEEE Transactions on Software Engineering*, vol. 44, no. 12, pp. 1146–1175, 2017.

[2] I. David, K. Aslam, S. Faridmoayer, I. Malavolta, E. Syriani, and P. Lago, "Collaborative Model-Driven Software Engineering: A Systematic Update," in *Proceedings of the 24th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, 2021.

[3] C. Masson, J. Corley, and E. Syriani, "Feature model for collaborative modeling environments.," in *MODELS*, pp. 164–173, 2017.

[4] J. S. Dumas, J. S. Dumas, and J. Redish, *A practical guide to usability testing*. Intellect books, 1999.