Design Guidelines for Improving User Experience in Industrial Domain-Specific Modelling Languages

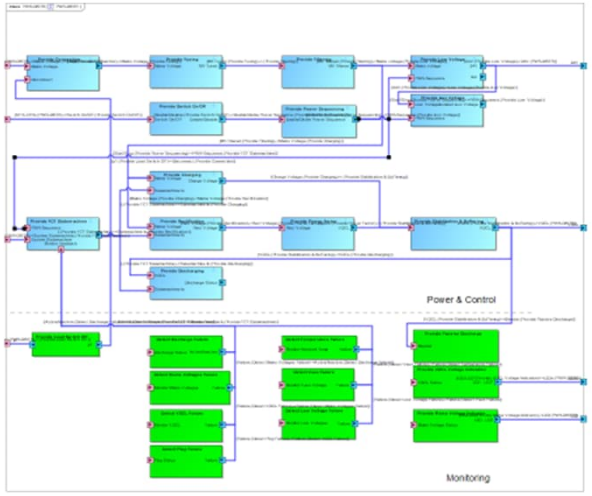
Rohit Gupta, Nico Jansen, Nikolaus Regnat, Bernhard Rumpe

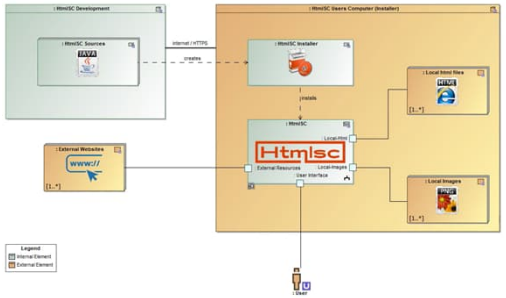
SE Software Engineering | RWTH AACHEN UNIVERSITY

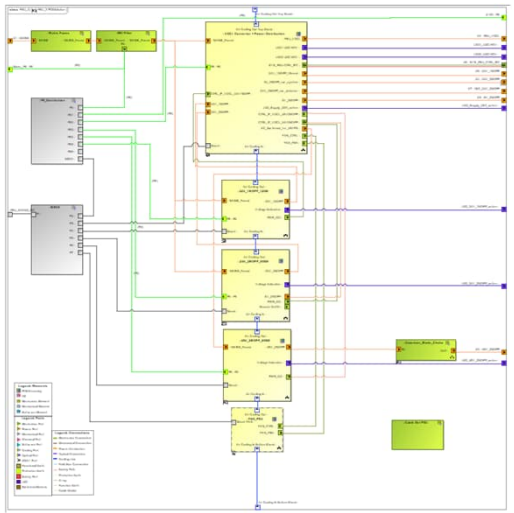
SIEMENS

Domain-Specific Modelling Languages

Domain-specific modelling languages help domain experts solve modelling challenges specific to a **domain** by providing a bridge from the problem space to the implementation space.





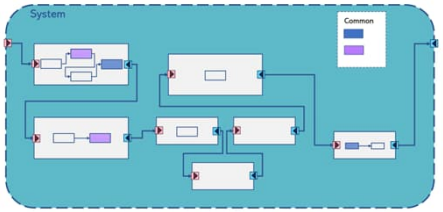


Motivation

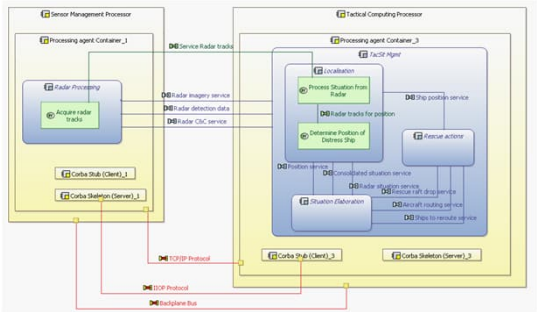
- Modelling involves key decision making in software and systems engineering, hence the need for conveying all relevant aspects of a domain.
- The usability of graphical DSMLs is still a challenge, as there is a lack of specific tooling guidelines for improving the user experience (UX) of practitioners.
- The overall topic of UX is vast and subjective, therefore general guidelines and definitions are either overly generic or tied to specific technological spaces or tools.
- To solve this challenge, we leverage existing design principles and standards of human-centred design and UX and propose definitions and guidelines for improving UX in graphical DSMLs.
- We categorize key aspects of user experience design (UXD) that language engineers should consider independent of graphical modelling tools and show this using an example of a Feature Model.

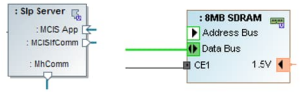
Modelling in MagicDraw

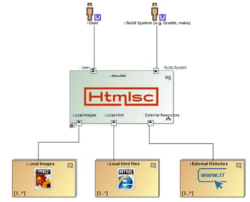
- Our tool of choice, MagicDraw, is based on UML and comes with extensions for SysML, providing a wide range of customization possibilities.



- A systematic engineering process of developing industrial DSLs includes:
  - Modular reusable DSL Building Blocks consisting of language components, a method, and a UXD part.
  - Iterative developmental approach, where all stakeholders are involved in the project.







- Stereotypes and customizations allow for the creation and definition of the language profile, consisting of language component artefacts.

- Our methodology has been used in a variety of domains: Healthcare, Energy, IT, Digital Industry, as well as public funded MBSE projects.

SE Software Engineering | RWTH AACHEN UNIVERSITY SIEMENS

Definitions – UX and UXD

- **User Experience (UX)**

An instantaneous intuitive feeling (positive or negative) of a user (modeller) while interacting with the defined constructs of the graphical modelling language and the graphical modelling tool.

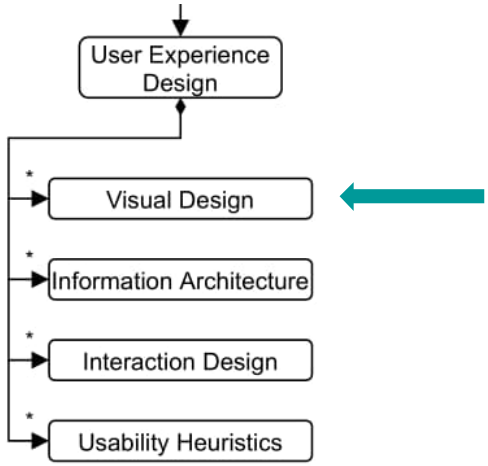
  - Good UX: Satisfies the modelling expectations of users in easy, positive, simple terms with minimal interactions.
  - Bad UX: Tends to invoke negative feelings that leaves users dissatisfied and introduces incomprehensibility between the stakeholders.

- **User Experience Design (UXD)**

Any design decision taken by a language engineer during the development of a graphical DSML, that ultimately fosters a good UX for a user.

  - Categorized based on the principles of human-centred design and supported with rationales by collecting feedback from users and domain-experts.

Categorization of UXD

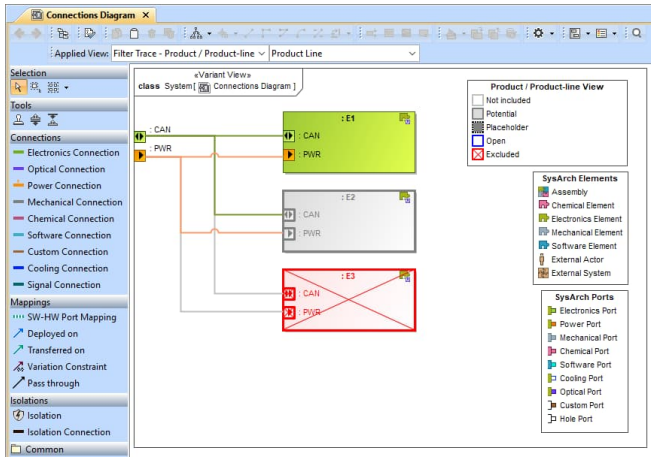


Visual Design

Represents the aesthetics (look and feel) of models and model elements.

- Configured using various icons, colours, appearances, dialogs along with their properties such as shape, size, and opacity.

- **Icon**: an extra graphical element displayed upon selection of a model element.
- **Colour**: enhances the appearance of a model element with a specific colour.
- **Modal Dialog**: a graphical control element with information for users on making relevant modelling decisions.
- **Custom View**: visual representation of the textual information using matrices, tables, or diagrams.
- **Dynamic View Plugin**: a GPL based plugin for enabling dynamic filtering and/or displaying specific model information on diagrams or custom views.

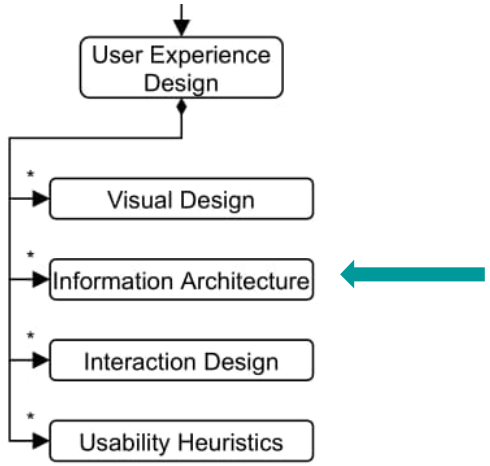


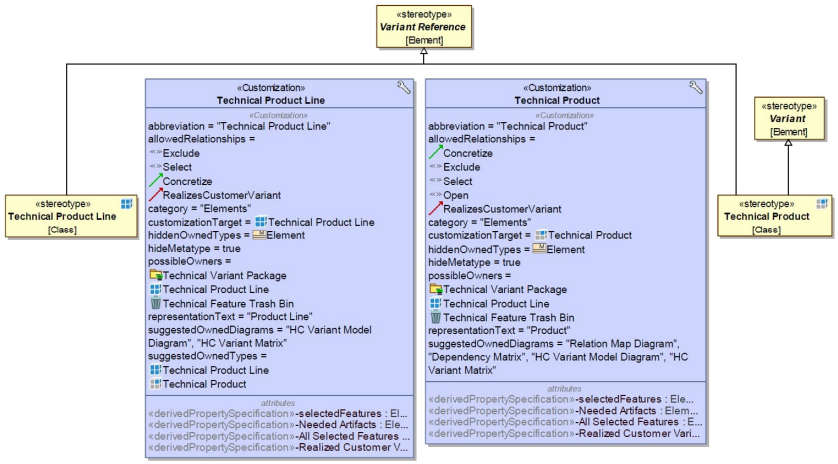
Categorization of UXD

Information Architecture

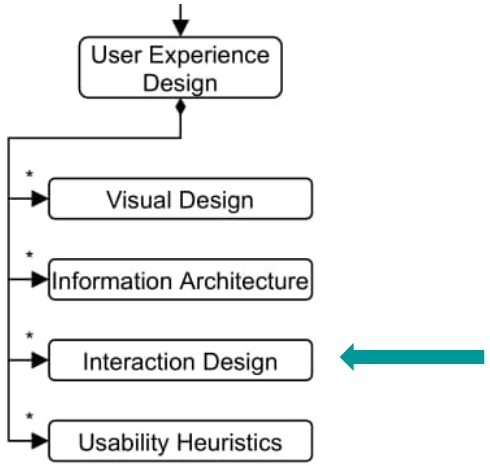
Structuring and organizing the constructs of the graphical DSML in a way that they are easy to find and use.

- **Layout**: determination of the position of model elements on custom diagrams based on the context of use.
- **Model Browser**: a visual representation of the hierarchy of model elements. It is a hierarchical navigation tool for managing the model data.
- **Perspective**: displaying a fixed set of modelling language constructs or tool functionalities based on the kind of user, novice or advanced.
- **Creation View**: an additional pane or window that shows the logical grouping of different language elements, standard UML diagrams and custom views during model element creation on the model browser.





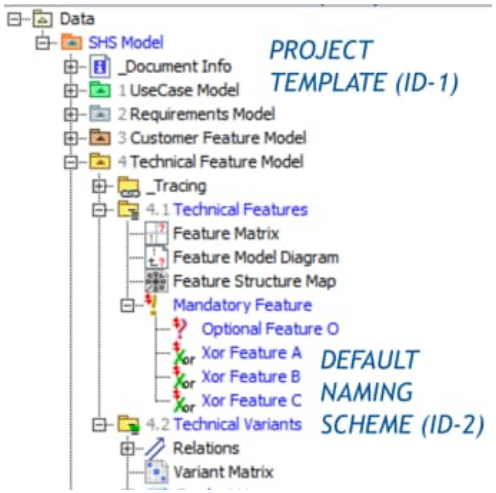
Categorization of UXD

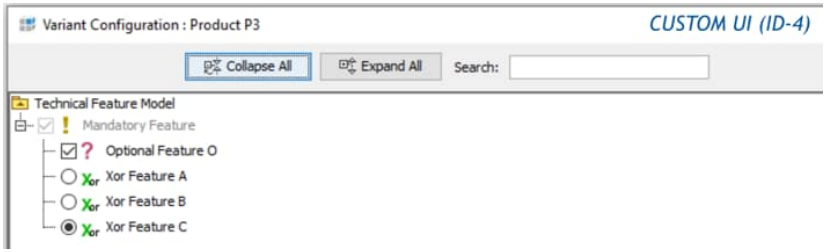


Interaction Design

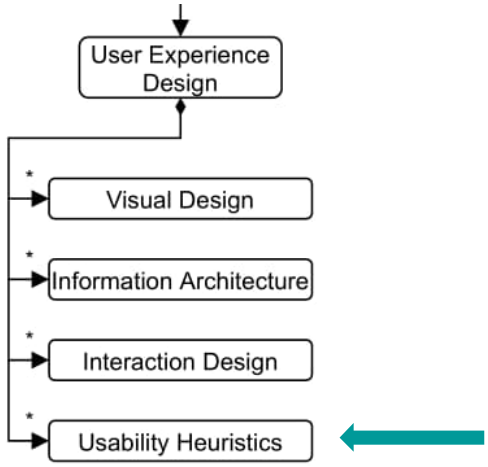
Help users interact effectively with the constructs of the graphical DSML while focussing on the cognitive dimensions.

- **Project Template**: a customized project pattern that serves as a starting point for creating a project in a predefined format.
- **Default Naming Scheme**: a naming scheme automatically assigning default names or numbers to model elements.
- **Model Transformation**: a transformation of a model into another formalism.
- **Custom User Interface**: a custom user interface (UI) programmed using frontend frameworks (e.g., Java Swing) to access and/or edit specific model and model elements based on the DSML requirements.





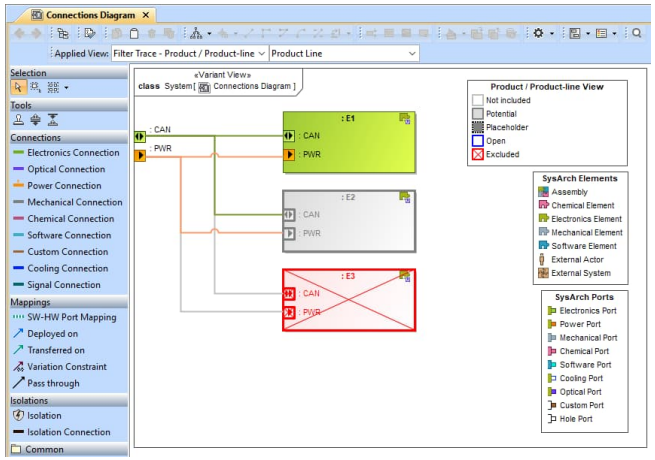
Categorization of UXD



Usability Heuristics

Help language engineers in making development decisions that are ultimately beneficial to users in achieving modelling with a greater sense of effectiveness and satisfaction.

- **Knowability (Clarity)**: constructs of the modelling language should be self-explanatory.
- **Knowability (Helpfulness)**: the graphical DSML should provide helpful annotations and documentation to users and also identify deprecated elements.
- **Operability**: the graphical DSML should provide the necessary domain-specific functionalities in addition to being extensible for further language compositions.
- **Robustness**: the graphical DSML should be well-formed and be free from bugs and vulnerabilities that could potentially expose flaws in the system.
- **Safety**: the graphical DSML should not compromise the confidentiality or the assets of a user.



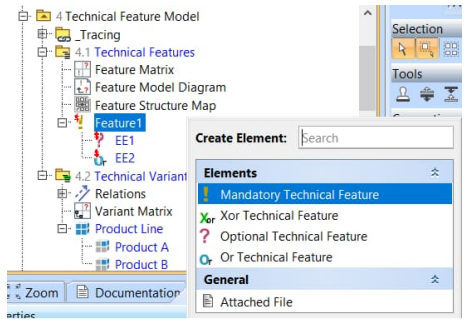
Scope of UXD

- The design decisions we list are non-exhaustive, but important in the development of any graphical DSML.
- Constraints in project duration, resources, and budget introduces trade-offs in design decisions that language engineers must consider.
- It is also common for language engineers to focus more on the functional aspects of the modelling language.
- Questions that language engineers must address during language development:
  - **Q1: Does a specific design decision fulfil a user's needs or a modelling goal?**
  - **Q2: Is the design decision a cause for any potential conflict, either between the constructs of the modelling language or with the existing functionalities of the modelling tool?**
  - **Q3: Is the design decision specific, non-subjective, and has relevance to the domain in consideration?**

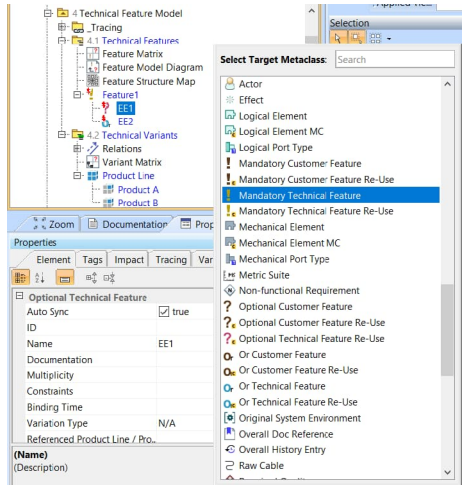
Case Study

- We present an example of a Feature Model used to configure language elements, products, and product lines specifically for Siemens Healthineers.
- Design decisions such as **icons**, **colour**, **default name**, and **creation views** are incorporated directly into the language definition, using stereotypes and customizations.
- **Modal dialogs**, **custom view**, **dynamic view plugin**, and **custom UI** is created by leveraging MagicDraw OpenAPI Java API capabilities, including Java swing code.
- **Model browser**, **project templates**, and **perspectives** are configured directly using in-built MagicDraw functionalities.
- **Model transformations** is achieved to refactor features in a feature tree to other formalisms, such as from an already defined optional feature to a mandatory feature.

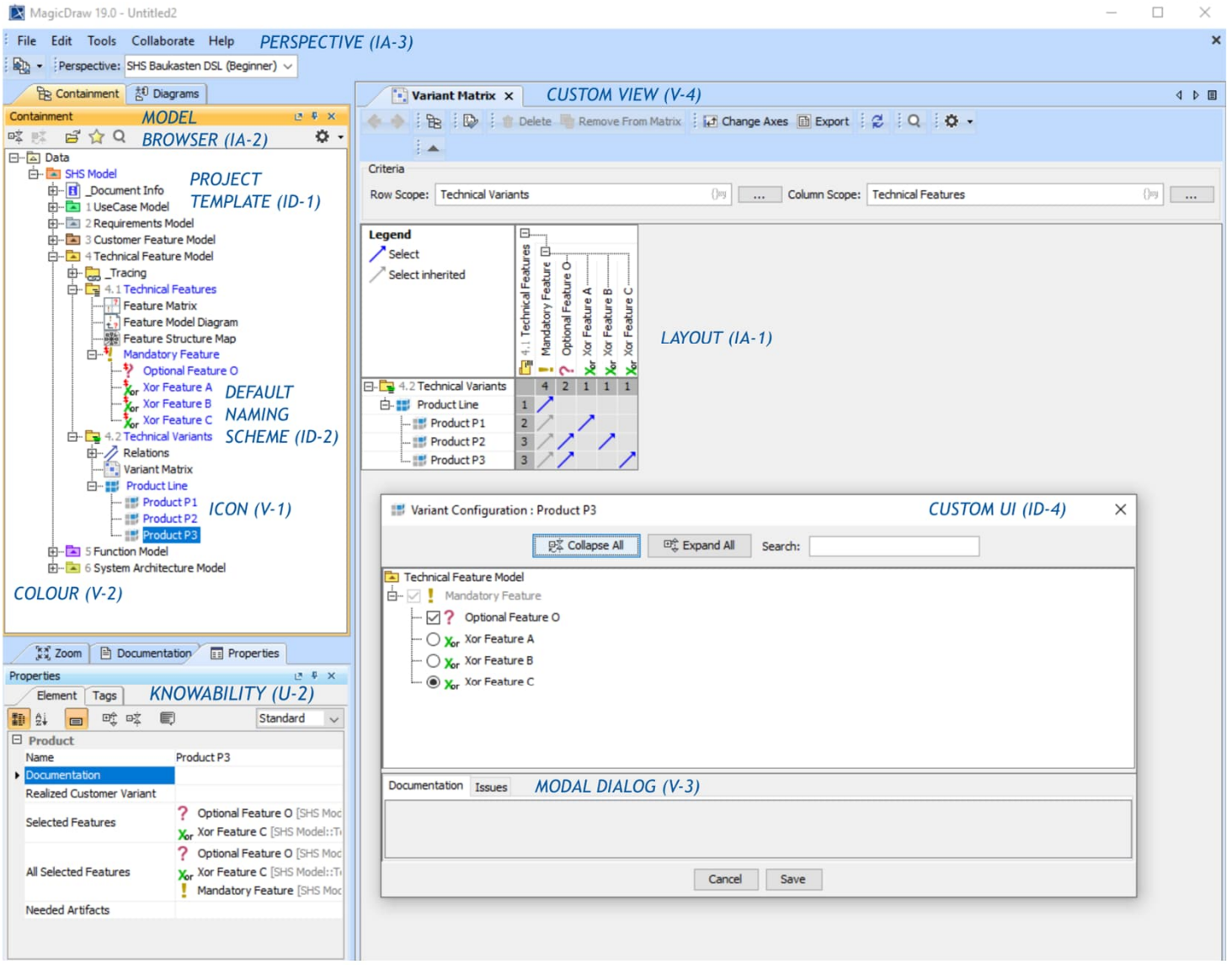
Case Study



Creation view to create only specific features.



Model transformation on existing features.



Discussion

- In this paper, we present definitions for UX and UXD for graphical DSMLs. Further, we categorize UXD aspects, detail the scope, and provide a case study for a real industrial DSML in the healthcare industry.
- Language engineers are not generally not UX experts, therefore they often need additional trainings. These guidelines are aimed for all kinds of language engineers: novice or advanced.
- **Visual designs** improve the aesthetics of models, **information architecture** designs help organize and structure models, **interaction designs** help users interact effectively with their modelling, and **usability heuristics** improve the overall effectiveness and satisfaction of users.
- While we used MagicDraw as an example tool, the guidelines apply generally to other modelling tools as the basic foundation remains the same: language development.
- No cookie cutter solution exists for improving UX, but involving all stakeholders (including UX experts) from the project start is key to improving the modelling experience of all kinds of users.

Conclusions

- The challenge of providing UX and UXD guidelines for language engineers still exists.
- We leverage the standards of human-centred design and UX, and propose aspects of UX and UXD that should be considered when building graphical DSMLs.
- We categorize the design decisions by utilizing the wide range of customizations offered by graphical modelling tools such as MagicDraw.
- We have built many DSMLs both in practice and research over the years, with constant feedback from users and domain-experts on how to improve the UX of DSMLs in practice.
- Naturally, UX is a subjective topic, and our proposed list of design decisions is non-exhaustive.
- Our aim is to provide guidance to language engineers to improve usability and UX in graphical modelling independent of graphical modelling tools.

Contact

Rohit Gupta
Siemens AG, Munich, Germany, rg.gupta@siemens.com

Nico Jansen
RWTH Aachen University, Aachen, Germany, jansen@se-rwth.de

Nikolaus Regnat
Siemens AG, Munich, Germany, nikolaus.regnat@siemens.com

Bernhard Rumpe
RWTH Aachen University, Aachen, Germany, rumpe@se-rwth.de