PAPYRUS FUTURE

CEA Papyrus Team

- **Existing:**
  - The definition of a DSML abstract syntax in Papyrus is done with the profile editor. It lets define abstract syntax constraints in OCL and Java.

- **Ongoing:**
  - "Façade"[1] lets you define a DSL that behaves and looks like a "from scratch Ecore metamodel-based DSML" from the abstract syntax point of view but which is just a proxy for a UML model. It is currently re-implemented using Active Operation Framework [2].
  - Stereotypes can also be defined during modeling with "on-the-fly" creation of Stereotypes (https://www.youtube.com/watch?v=quoDrUtsFsI).

- **Existing:**
  - Today, in Papyrus, the definition of a diagrammatic concrete syntax is done thanks to GMF-Tooling, ElementTypesConfigurations and hard-coded Java and ultimately relies on GEF and GMF.
    - A diagram can "inherit" from another.
    - ElementTypesConfigurations are used to capture all the elementtypes with model-based definitions. Ultimately, the elementTypes are those responsible for the semantic model edition.
  - Tabular concrete syntax is defined thanks to a Papyrus specific metamodel and relies on Nattable.
  - Textual concrete syntax: Textual editors are defined and implemented thanks to XText.
  - "Symbols" framework in Papyrus lets you override the rendering of a figure in an existing Papyrus diagram using any image including SVG.
  - CSS: An editor and generator let you customize the styles of the diagrams from scratch or from a styled model.

- **Ongoing:**
  - Diagram Definition [3] implementation is ongoing. This covers the implementation of DI and DG editors, DG SVG-based renderer, UMLDI extension of DI, implementation of the UMLDI to DG transformation and the export of Papyrus diagrams models to UMLDI models.
  - We have an ongoing PhD on the graphical semiology and pragmatics of DSLs which basically aims a providing methods, guidance and tooling to have "good DSLs" and better interface between the designer and the modeling language.

- **Foreseen improvements:**
  - Capture and exploit interactors: the impact of the user interaction with the model through the diagrammatic editors is hard-coded. That may be improved.
  - Capture and exploit semantic layouts: whenever the layout has a semantic (i.e. the position of an element in a diagram w.r.t. another element has a specific meaning), we hard-code the edition commands. We may progress on that.

- **Existing:**
  - Papyrus provides a reference implementation of the ISO42010 metamodel and provides a specialization of it for Papyrus needs. So far, "Viewpoints in Papyrus enable the specialization and customization of the user experience by constraining what can be seen and interacted with in Papyrus diagrams and tables".
- **Foreseen improvements:**
  - We expect this framework to be the aggregator/integrator of all the customization facilities in Papyrus. It should become the central point from which we will define the DSML.

- **Existing:**
  - To be sincere, no formal or guided methodology has been devised so far: the definition and implementation of a DSML in Papyrus relies heavily on expertise with both UML as such and Papyrus customization facilities.

- **Foreseen improvements:**
  - A systematic methodology should be devised and documented to drive the definition and the implementation of DSMLs in Papyrus

- **Existing:**
  - Custom Palettes: specific palettes in diagrammatic editors can be defined by a model with a dedicated editor. These palettes models can specify some pre and post semantic and graphical actions to the creations command.
  - Properties view: these are "form-based" edition means that are model-defined. A dedicated editor is used to define and preview them.
  - Modeling Assistants (https://www.youtube.com/watch?v=tyc0gWU4xj0): modeling assistant can be modeled. They can be generated from a UML profile. A specific editor is provided for that.
  - New Child menus: The creation menus can be modeled. It lets you create specific creation menus. A specific editor is provided for that.
  - Model-explorer: the model-explorer can be customized to provide a specific tree view of the model.
- **Ongoing:**
  - We have an ongoing PhD that aims at developing or adapting modern HCI for better interactions between the designer and the modeling tools and thereby ease the modeling process.
- **Foreseen improvements:**
  - Federate UI customizations and better couple them with architecture-framework definitions

THANK YOU FOR YOUR ATTENTION

DRT/LIST
DILS
LISE