

# Tooling matters!



*FeatureIDE feature models outside the (Eclipse) box*

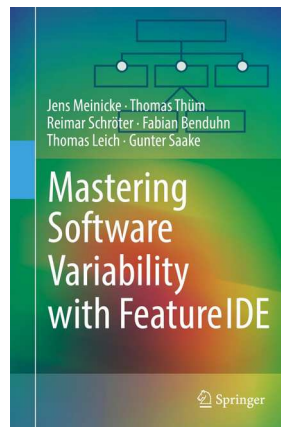
**Stefan Sobernig**, WU Vienna

FOSD 2024 @ TU Eindhoven

# Tooling Matters!

*„Defining languages and notations is not enough per se – you have to provide **good tool** support for them, too.“ [1]*

**f** IDE  
feature



### Languages and tools

- CIF**  
CIF is a modeling language and extensive toolset supporting the entire development process of supervisory controllers.  
[Learn more](#)
- Chi**  
Chi is a modeling language and toolset to analyze the performance of supervisory controllers.  
[Learn more](#)
- ToolDef**  
ToolDef is a cross-platform and machine-independent scripting language to automate CIF and Chi tools.  
[Learn more](#)

# Tooling Matters!

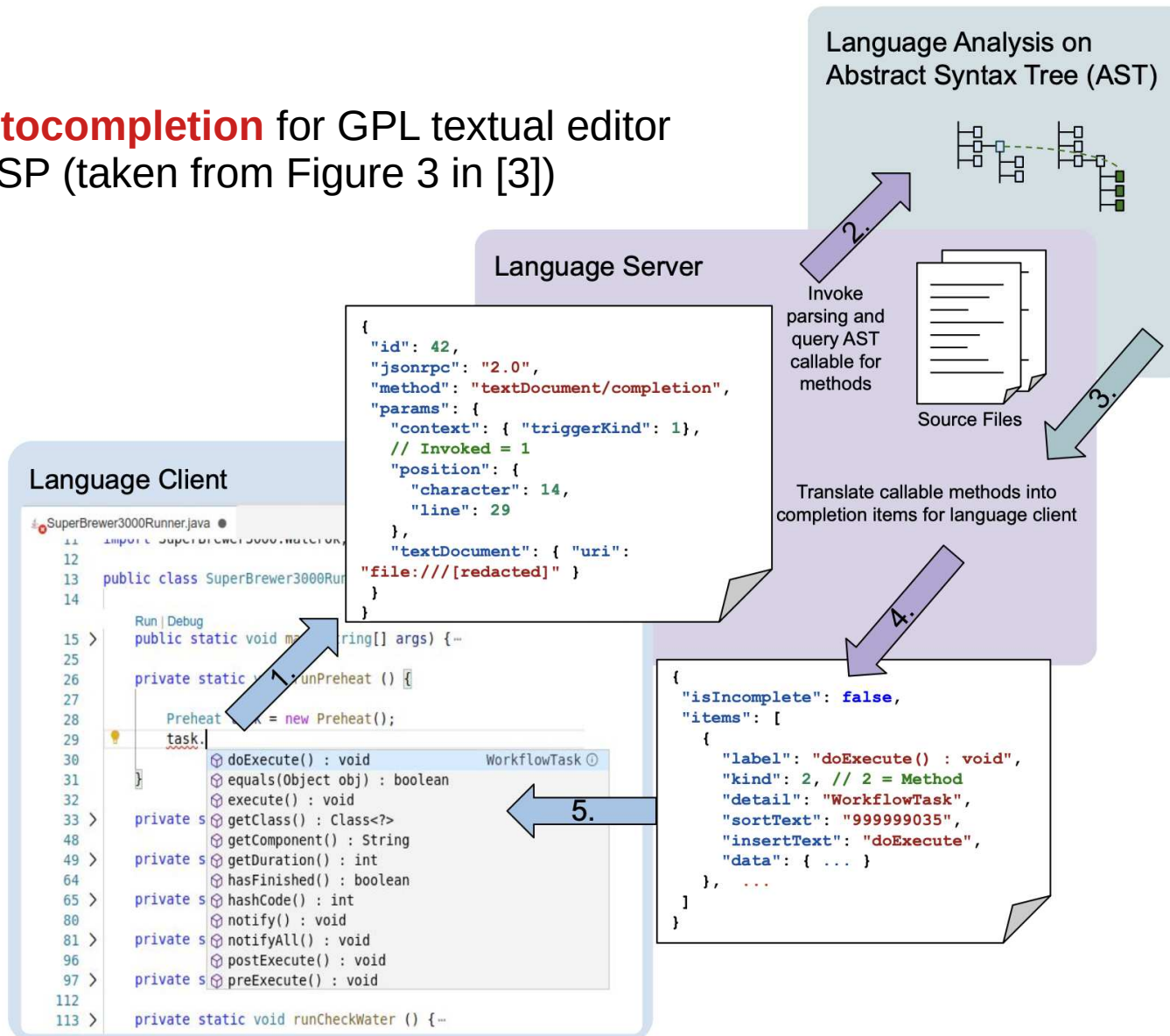
- What is the supporting (modelling) **tool**?
- What is a **good** (modelling) tool?
- Tool conundrum in modelling (see Selic [4]):
  - Produce **general-purpose** tools that cover the broadest possible audience to control for costs/ effort?
  - Develop **special-purpose**, domain-specific tools to boost productivity/ acceptance?



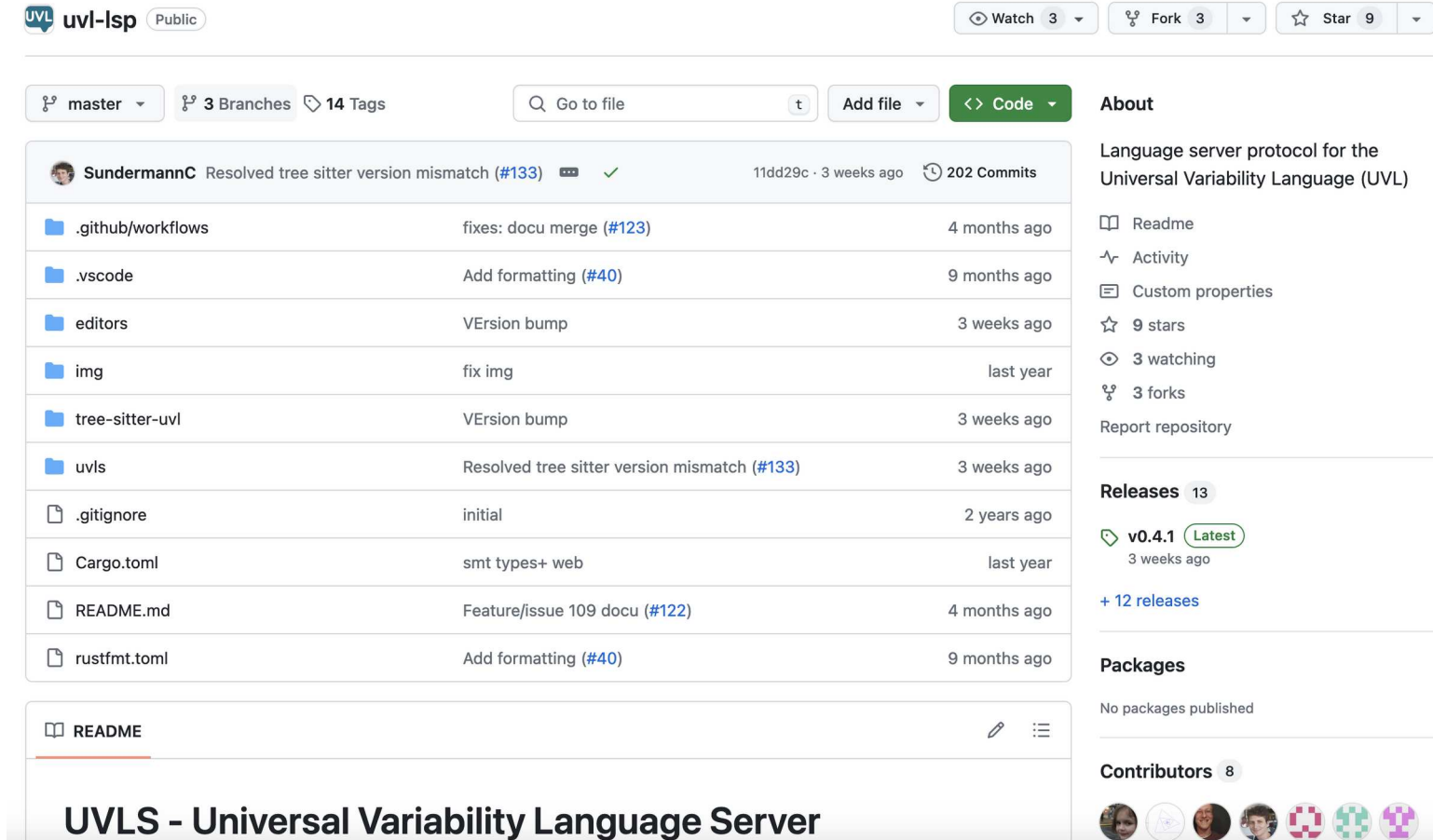
- Recent trends to address this tradeoff in IDEs: Decoupling IDE platform from language-dependent tools/ services.
- **Decoupling** based on language-tool platforms:
  - Language-aware „servers“ (language-specific backend infrastructure)
  - Text and diagram editors (language-agnostic frontend infrastructure)
  - Reusable language services (e.g., some auto-completion service)
- **Language Server Protocol (LSP)**:
  - Complements language workbenches and language-product lines
  - In this spirit: Computational notebooks backed by language kernels (but there are competing kernel protocols)

# Tooling Matters!

Ex.: **Autocompletion** for GPL textual editor using LSP (taken from Figure 3 in [3])



# LSP for UVL



The screenshot shows the GitHub repository page for 'uvl-lsp'. At the top, it indicates the repository is 'Public' and has 3 watches, 3 forks, and 9 stars. The main content area shows a list of files and folders with their commit history. A 'README' section is partially visible at the bottom, starting with 'UVLS - Universal Variability Language Server'. On the right side, there are sections for 'About' (Language server protocol for the Universal Variability Language (UVL)), 'Releases' (13 releases, with v0.4.1 being the latest), 'Packages' (No packages published), and 'Contributors' (8 contributors).

uvl-lsp Public

Watch 3 Fork 3 Star 9

master 3 Branches 14 Tags

Go to file t Add file Code

**SundermannC** Resolved tree sitter version mismatch (#133) 11dd29c · 3 weeks ago 202 Commits

.github/workflows	fixes: docu merge (#123)	4 months ago
.vscode	Add formatting (#40)	9 months ago
editors	VErsion bump	3 weeks ago
img	fix img	last year
tree-sitter-uvl	VErsion bump	3 weeks ago
uvls	Resolved tree sitter version mismatch (#133)	3 weeks ago
.gitignore	initial	2 years ago
Cargo.toml	smt types+ web	last year
README.md	Feature/issue 109 docu (#122)	4 months ago
rustfmt.toml	Add formatting (#40)	9 months ago

**README**

## UVLS - Universal Variability Language Server

**About**  
Language server protocol for the Universal Variability Language (UVL)

- Readme
- Activity
- Custom properties
- 9 stars
- 3 watching
- 3 forks

Report repository

**Releases** 13

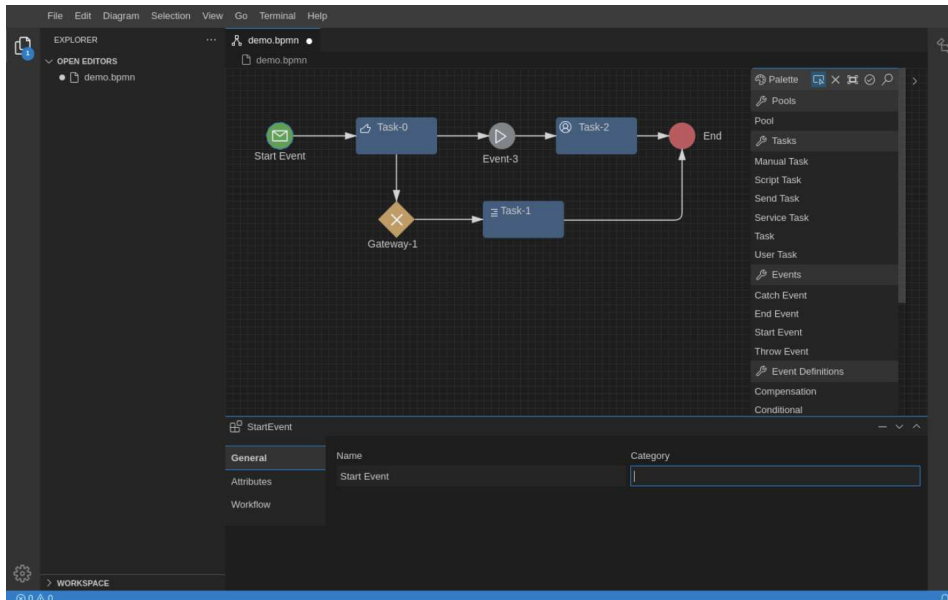
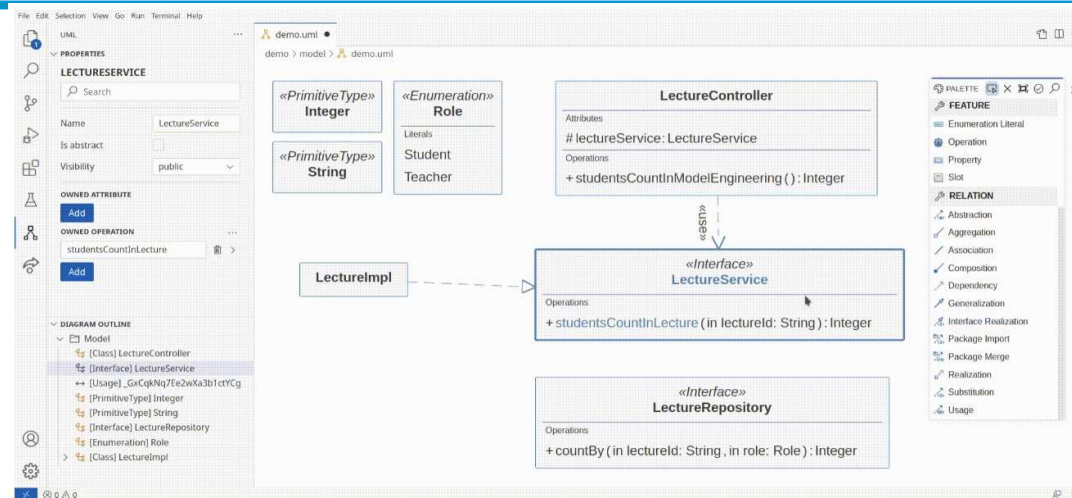
v0.4.1 Latest  
3 weeks ago

+ 12 releases

**Packages**  
No packages published

**Contributors** 8

# Tooling for **Modelling** Matters!



**Open  
BPMN**

# Unboxing FeatureIDE modeling?

Open

IDE Integration for Visual Studio #1345

Febbe opened this issue on May 7, 2022 · 1 comment



tthuem commented on May 9, 2022

Member



Hi Fabian,

thanks for your interest in FeatureIDE.

So far there are no concrete plans to port FeatureIDE to other IDEs, but we are refactoring the code for version 4.0.0 such that it will be significantly easier to integrate it with other IDEs.

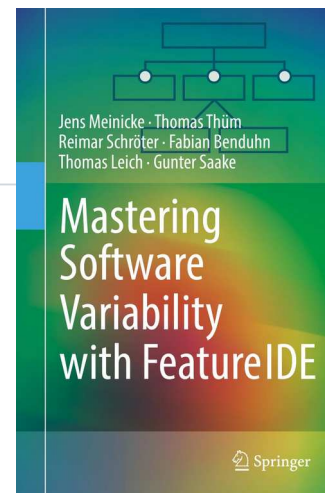
Nevertheless, it could be helpful for others to understand what functionality of FeatureIDE you would like to see ported to Visual Studio. Maybe there are other users interested in this and work jointly towards such an integration.

Best regards

Thomas

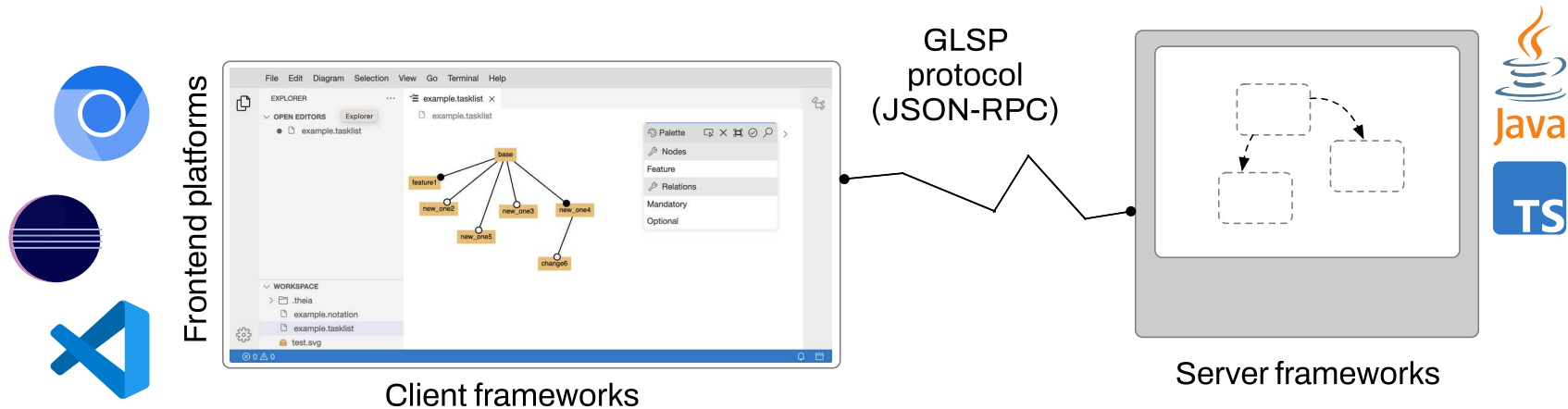


**f** IDE  
feature



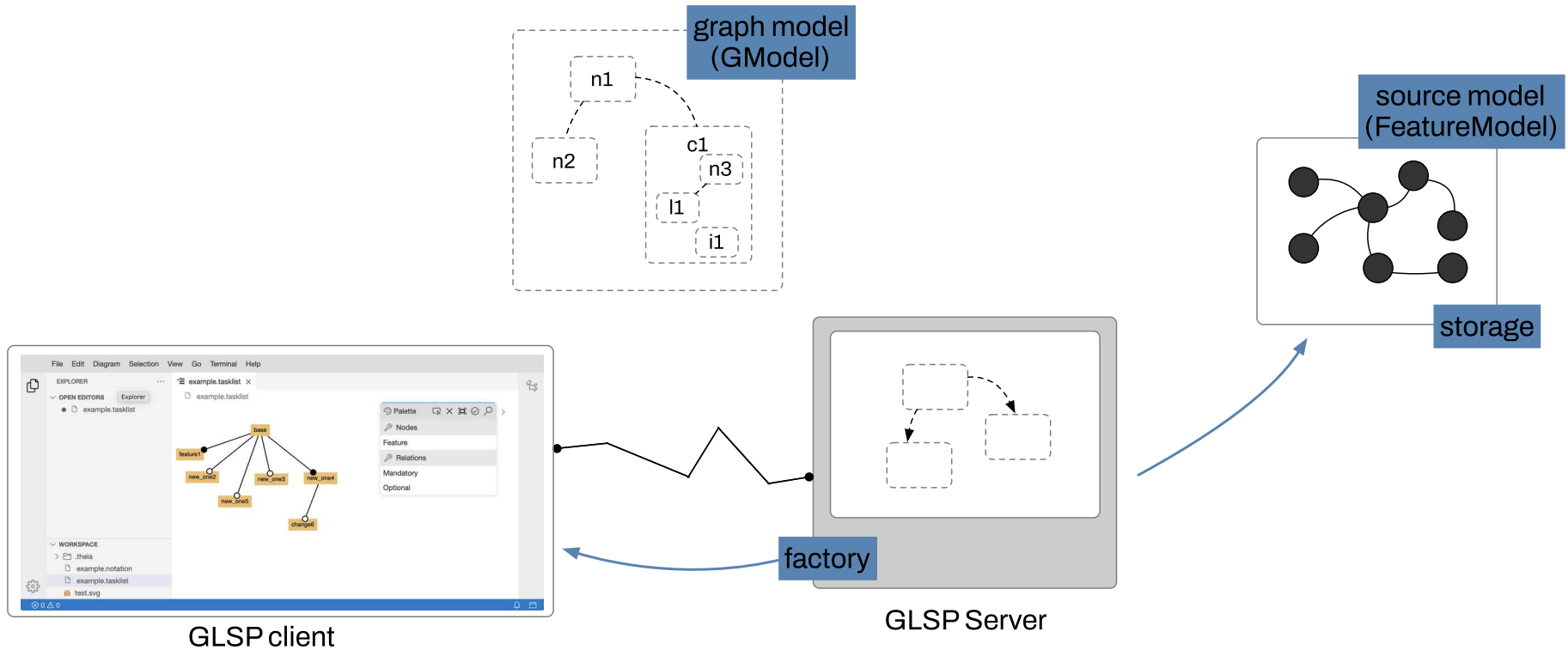


# Eclipse Graphical Language Server Protocol (GLSP; 1)



- LSP for graphical modeling
  - Allows for developing browser-based diagram clients;
  - Frontend focuses on diagram rendering & modeler interaction;
  - Backend and diagram server provides the various language services;
- Integrates with **multiple frontend** infrastructures (e.g., Chromium, WebViews, Eclipse RCP, Visual Studio)
- Different **deployment** options in the front- and backend (e.g., desktop, cloud, browser-only)
- Integrates with **various backend models** (custom class models, abstract-syntax models, EMF/ Ecore models)

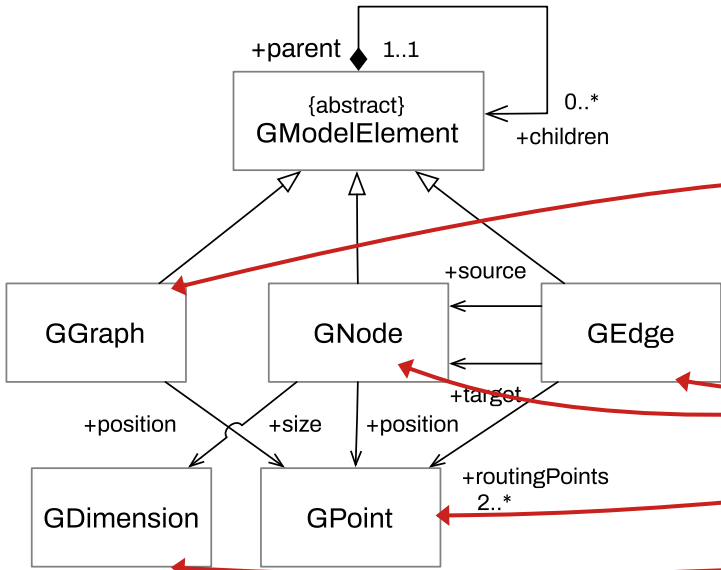
# GLSP (2): Provisioning



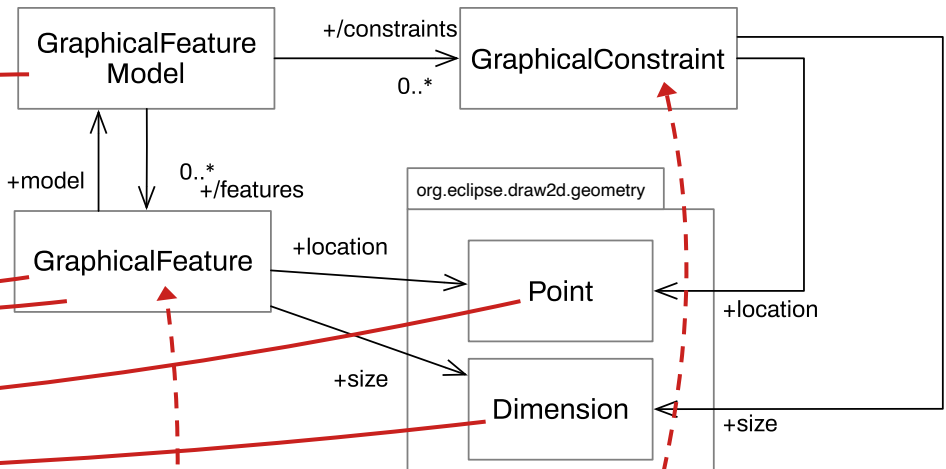
# GLSP (3): Mapping (backend)

Ecore/ Java

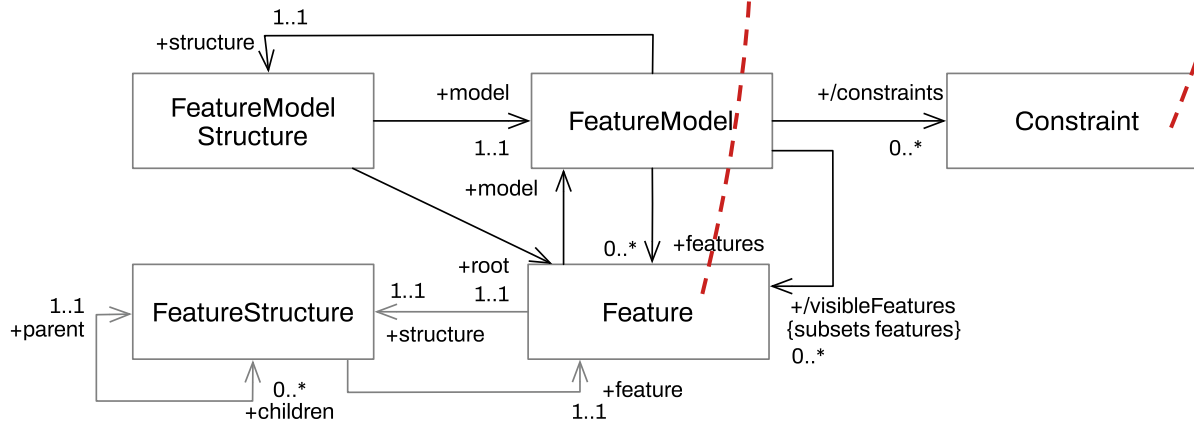
## GLSP GModel



## FeatureIDE GraphicalFeatureModel



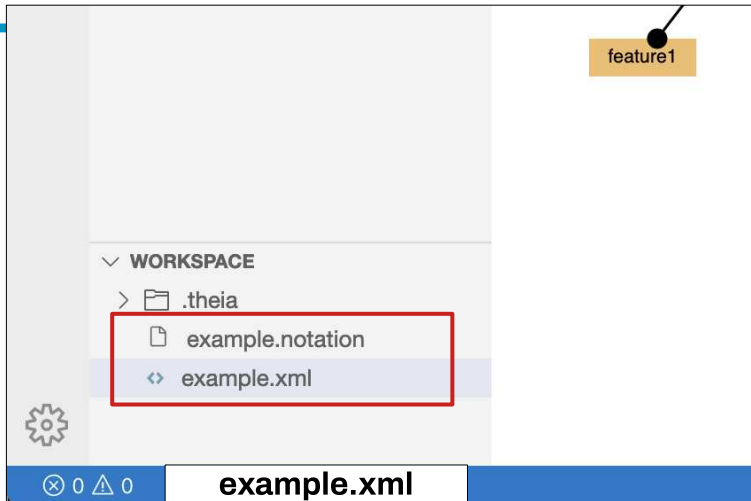
## FeatureIDE FeatureModel



Java

Java

# GLSP (4): Storage



feature1

## example.notation

```
<?xml version="1.0" encoding="UTF-8"?>
<layout chosenLayoutAlgorithm="0" horizontalLayout="true">
  <struct>
    <feature X="124" Y="7" name="Base" />
    <feature X="58" Y="66" name="new_one1" />
    <feature X="30" Y="152" name="feature1" />
    <feature X="123" Y="132" name="new_one2" />
    <feature X="185" Y="32" name="new_one4" />
    <feature X="196" Y="68" name="new_one5" />
    <feature X="178" Y="170" name="XXX" />
    <feature X="240" Y="144" name="xxx" />
  </struct>
  <constraints />
  <legend />
</layout>
```

## example.xml

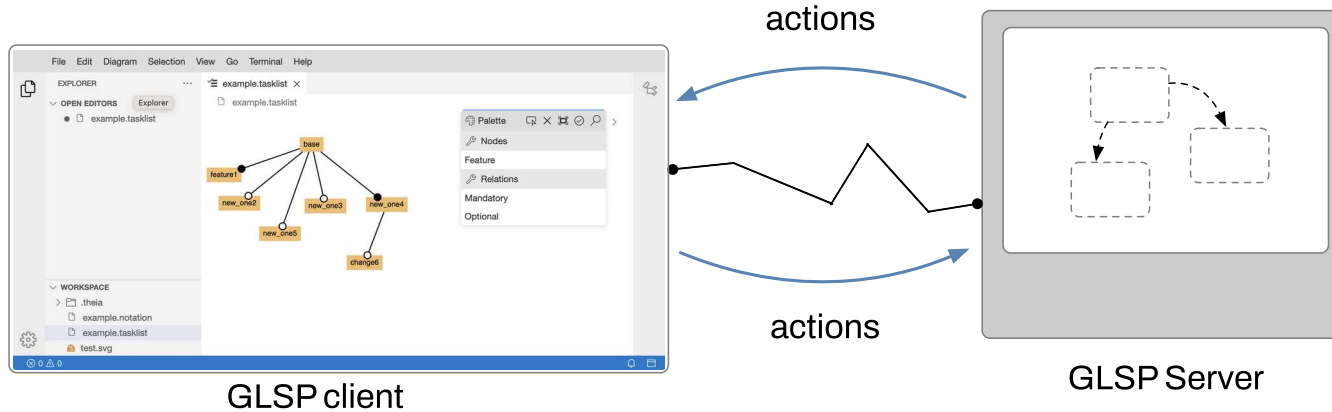
```
<?xml version="1.0" encoding="UTF-8"
<featureModel>
  <struct>
    <and mandatory="true" name="Base">
      <and mandatory="true" name="new_one1">
        <feature mandatory="true" name="feature1" />
      </and>
    <feature name="new_one2" />
    <and name="new_one4">
      <and name="new_one5">
        <feature mandatory="true" name="XXX" />
        <feature mandatory="true" name="xxx" />
      </and>
    </and>
  </and>
</and>
</struct>
</featureModel>
```

# GLSP (5): Diagram and model edits

✓ model operations (edits)

✓ palette actions

✓ navigation targets



source model  
(FeatureModel)


➔ more than **80 actions** (in default configuration)

# GLSP (6): FM4 as my MVP

FeatureIDE FM	Gmodel/ UI	Concrete syntax	Abstract syntax	Semantics/ services
Feature hierarchy (AND)	●	●	●	◐
Feature groups (XOR, OR)	○	○	○	○
Attributes	◐	◐	◐	◐
Cross-tree constraints				
Feature order				
Autolayouting				
Validation				
Diffing				
...				

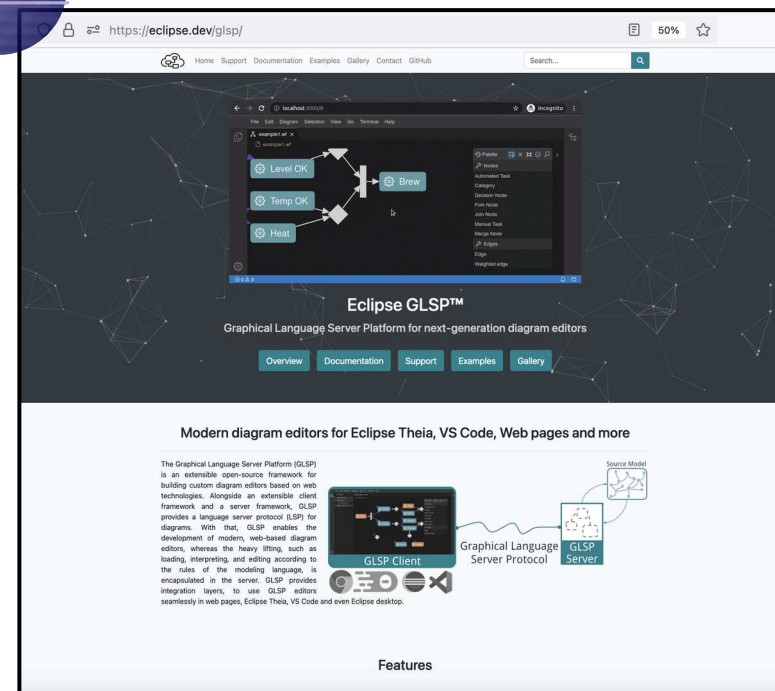
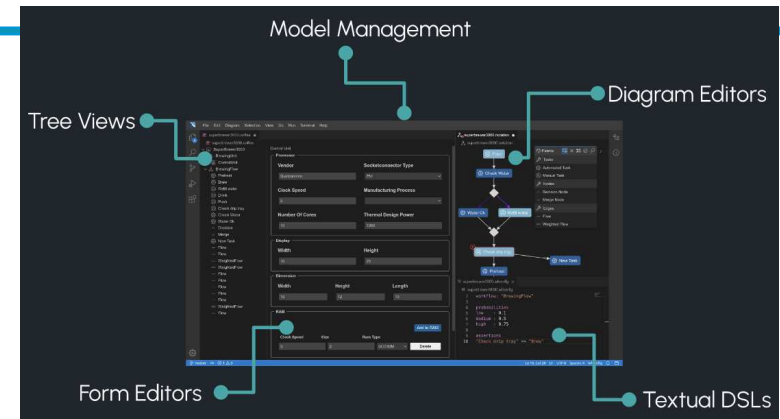
  

... the fastest way (for me) to run a first „Build-Measure-Learn“ loop

- FeatureIDE Library is eligible as a GLSP provider 
- Possible enhancements:
  - Provide for an „essential“ library (e.g., only two formats) and no external library dependencies (e.g., uvl-parser.jar)?
  - GraphicalFeatureModel
    - Make it a member of core?
    - Remove dependency on GEF (Point, Dimension)
    - Improved separation of concerns:
      - Representing graph embedding (e.g., size, location, shape)
      - Integrate with GEF/ Eclipse editor's event loop
  - Refactor „godness“ classes: Utility classes with many imports (e.g., FeatureUIHelper re: FeatureConnection)

# GLSP (8): Standing on shoulders

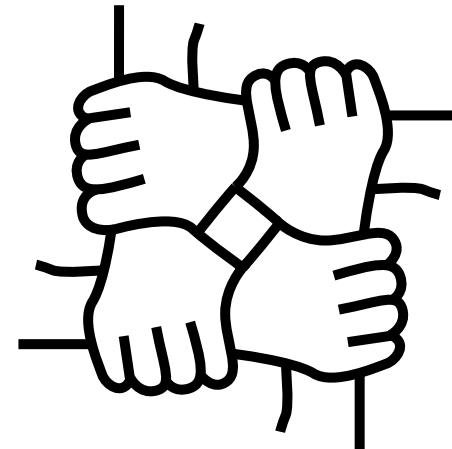
- **Testing support:** Unit, integration, and end-to-end testing
- **Mixed authoring:**
  - Syntax-driven textual editing
  - Form-based editing
  - SCM integration
- **Collaboration mode:** Multi-user authoring of diagrams by sharing model state and command stack
- Improvements to accessibility (incl. zooming)
- *(Instrumentation for activity tracking)*





# Next steps

- Complete the „**base feature**“
  - Feedback and notification channels to clients
  - Notational completeness (feature groups, constraints)
- Complete **packaging** the MVP (yarn + Maven)
- Adding basic (end-to-end) interaction **tests** (playwright)
- **Collaborations:**
  - Plan is to assign work packages to student projects (e.g. support for attributed feature models);
  - Call for contributors in the FOSD/ FIDE community 🗣️ 📣



# References

- [1] Markus Völter, “MD\* Best Practices”, Journal of Object Technology, Volume 8, no. 6 (September 2009), pp. 79-102, doi: 10.5381/jot.2009.8.6.c6.
- [2] Dominik Bork, Philip Langer, Tobias Ortmayr: A Vision for Flexible GLSP-Based Web Modeling Tools. PoEM 2023: 109-124, doi: 10.1007/978-3-031-48583-1\_7
- [3] Dominik Bork, Philip Langer: Language Server Protocol: An Introduction to the Protocol, its Use, and Adoption for Web Modeling Tools. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. 18: 9:1-16 (2023), doi: 10.18417/emisa.18.9
- [4] Bran Selic: What will it take? A view on adoption of model-based methods in practice. Softw. Syst. Model. 11(4): 513-526 (2012)