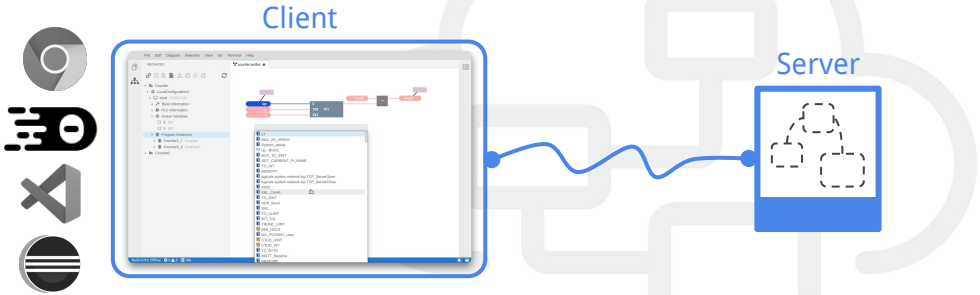


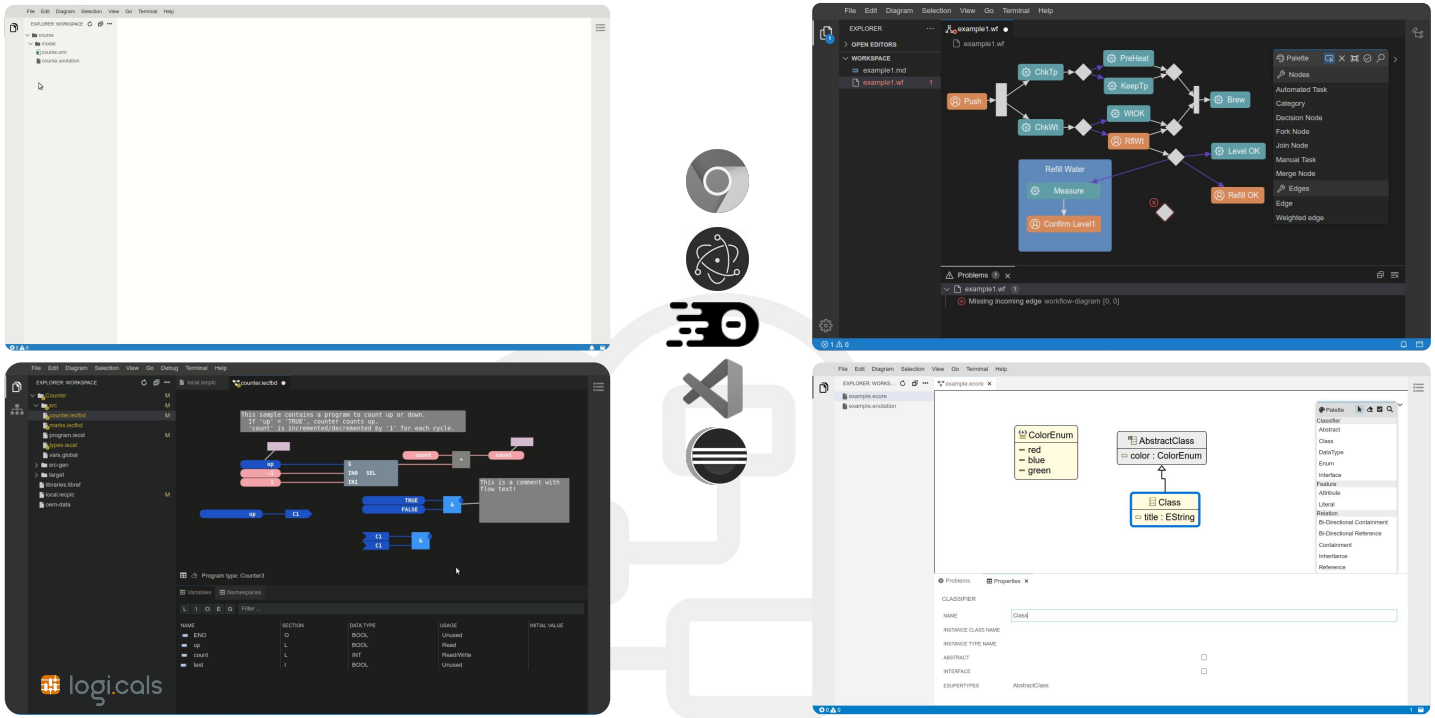
Diagram Editors with GLSP

Why flexibility is key

Philip Langer

planger@eclipsesource.com

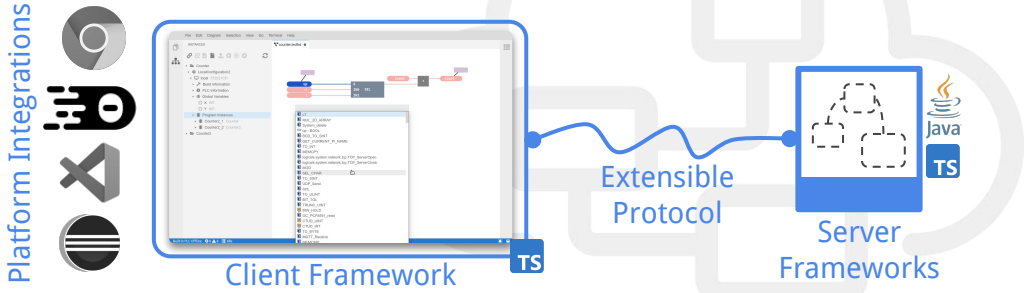
Eclipse Graphical Language Server Platform (GLSP)

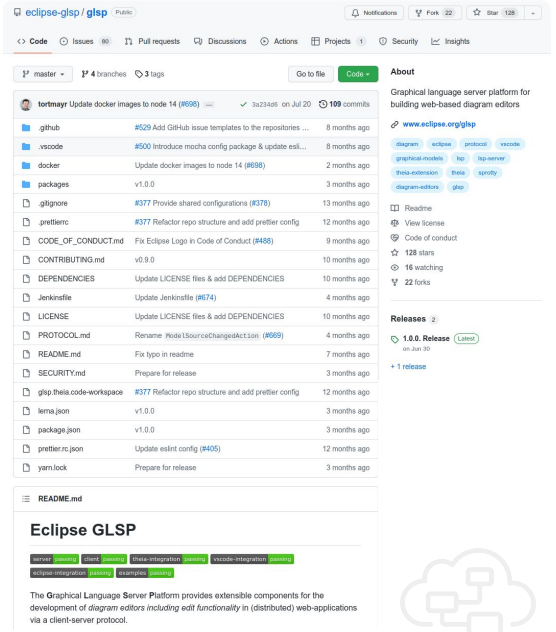


Eclipse Graphical Language Server Platform (GLSP)

Applying the architectural pattern of LSP to graphical modeling

- Development of browser-based diagram clients
- Frontend focused on rendering & user interaction
- Encapsulate language smarts on the diagram server

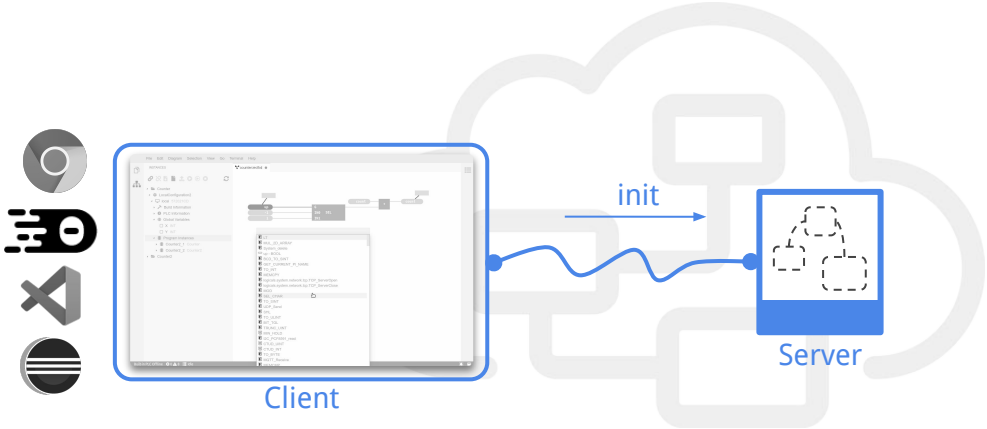




github.com/eclipse-glsp/glsp

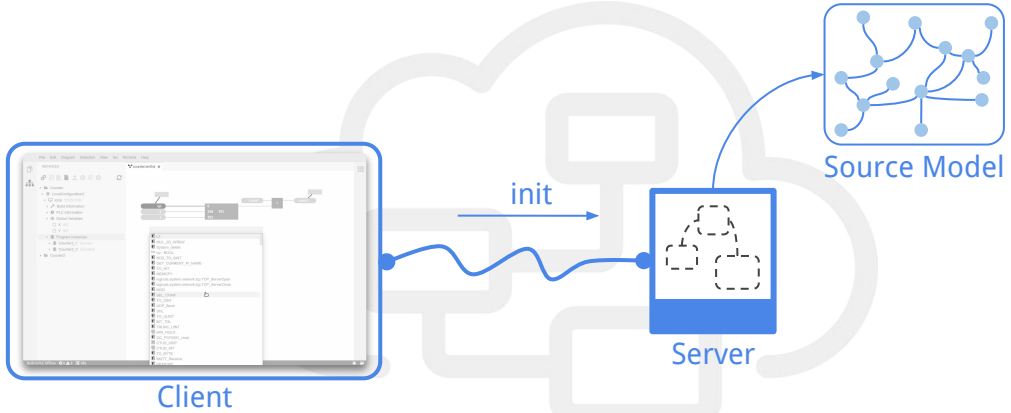
Eclipse Graphical Language Server Platform (GLSP)

- Initialization with parameters
  - URI
  - Diagram type
  - ...



Eclipse Graphical Language Server Platform (GLSP)

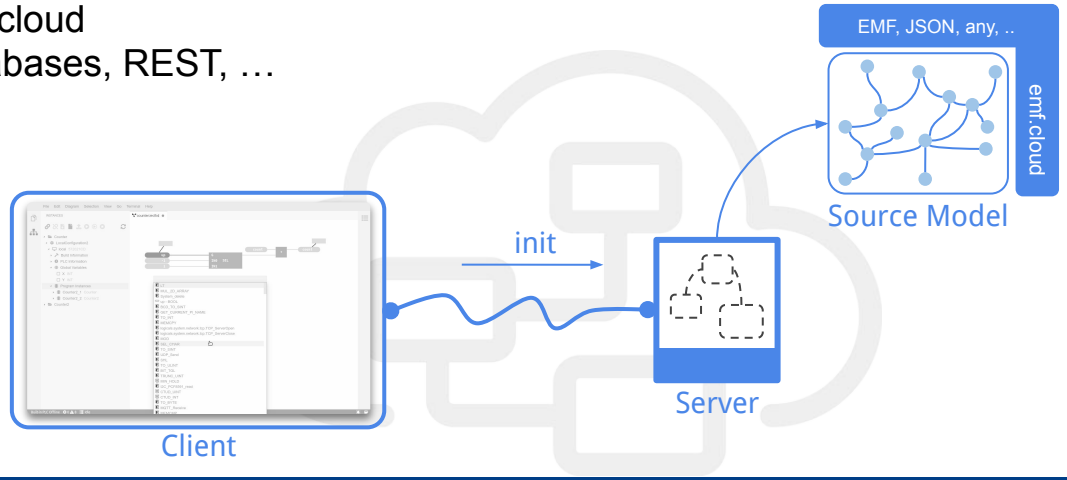
- Server has responsibility to obtain source model



Eclipse Graphical Language Server Platform (GLSP)

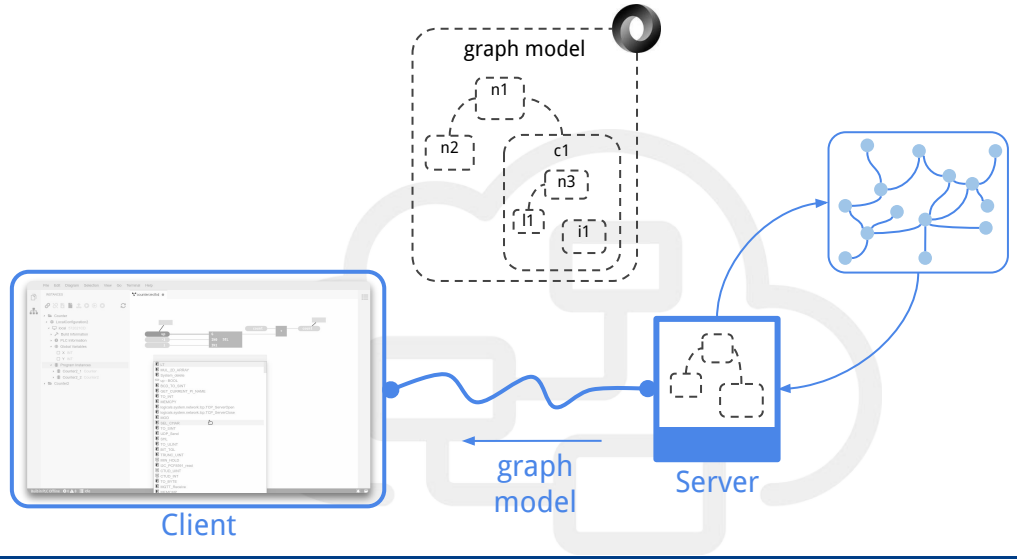
→ Source model can be anything

- EMF
- JSON
- Xtext
- emf.cloud
- Databases, REST, …



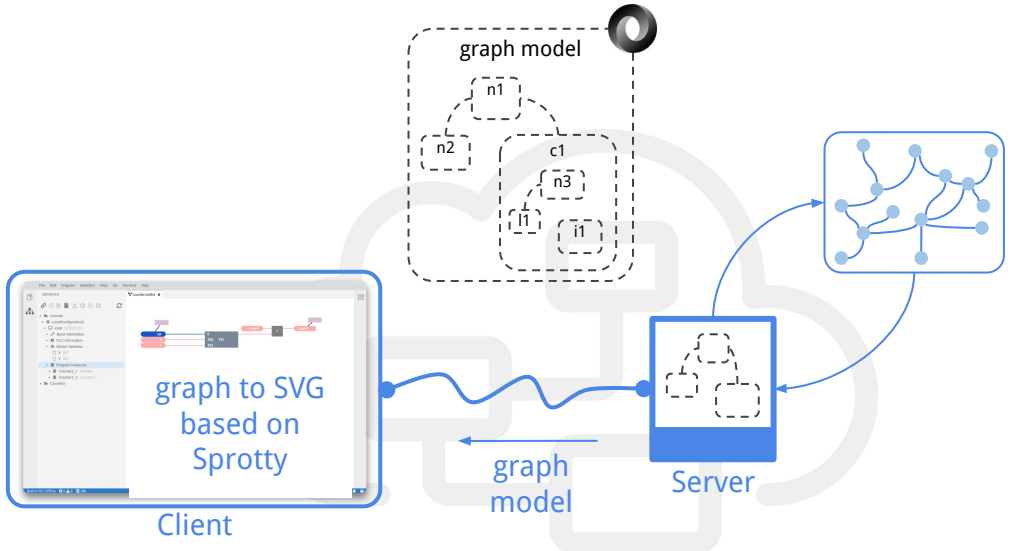
Eclipse Graphical Language Server Platform (GLSP)

- Server maps source model into *graph model*



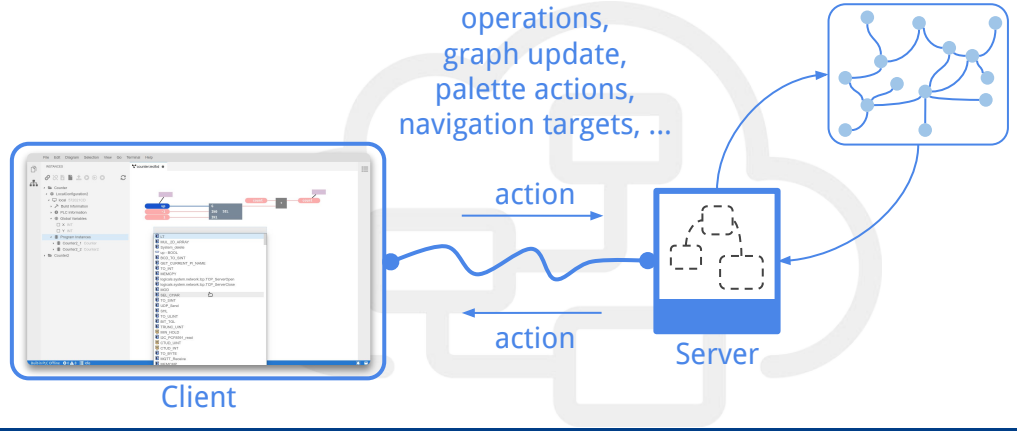
Eclipse Graphical Language Server Platform (GLSP)

- Client translates *graph model* into SVG with Eclipse Sprotty 👍

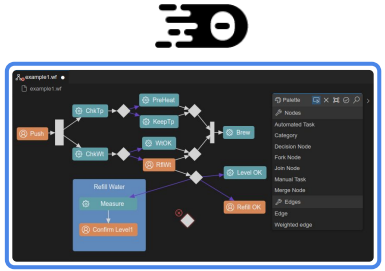


Eclipse Graphical Language Server Platform (GLSP)

- Editing tools on the client
  - Communicate with server via actions (defined in protocol)
- Extensible with custom tools and actions
  - Add domain-specific functionality
  - Adjust default behavior

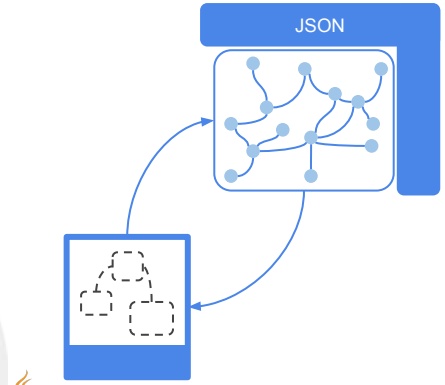


Eclipse Graphical Language Server Platform (GLSP)



TS Client Module

Demo



Java Server Module

What's new?

1

- API Stabilization and Server API Refactoring
  - Flexibilization of source model technology
  - Ready-to-use modules for EMF, JSON, EMF.cloud
- Node-based Server Framework
  - Typescript for client and server
  - Alleviates runtime requirements
- Documentation and Project Templates
  - Getting started quickly for your architecture

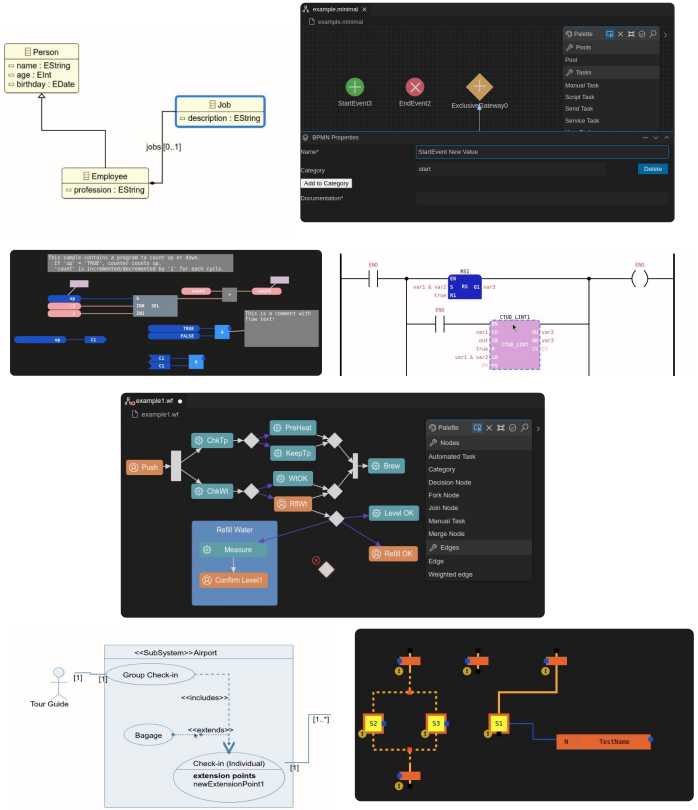
and more

contributors ~26
commits ~1027
npmjs weekly downloads ~1218

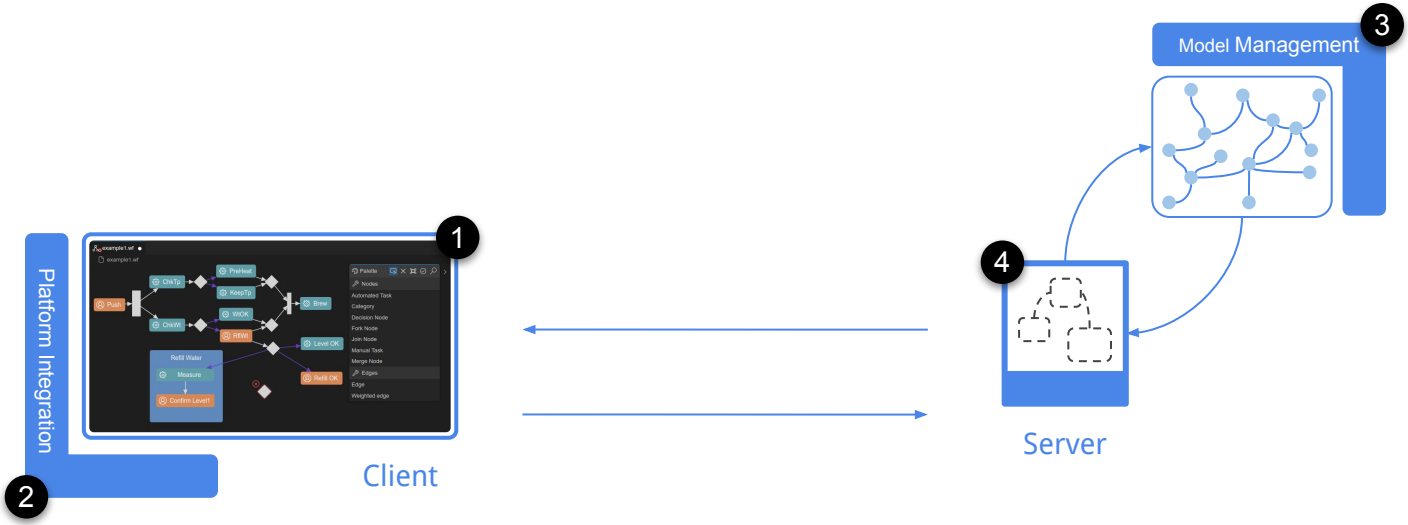
Why Flexibility is Key

- Not just about getting onto web stack / cloud
  - Architectural paradigm shift
  - Modern tech stack: fluid and diverse
  - Modularity and combinability

- Diagram editors: Specific to language by nature
  - Different domains
  - Different data sources
  - Different workflows
  - Different integrations with other tool components



Flexibility in GLSP's Architecture



Full Flexibility and Customizability on the Client

- Direct access to excellent base technologies
  - Customizable editing tools with Typescript
  - Custom UI controls with HTML
  - Dynamic SVG views and CSS

Demo

- No limiting abstraction layers in the middle
  - Well known technologies
  - Benefit from full power
  - Great debugging experience

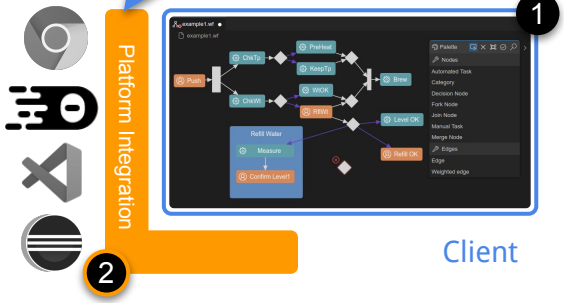
Maximum Reuse Across Tool Platforms

Editor registration, global menus, platform styling, etc.

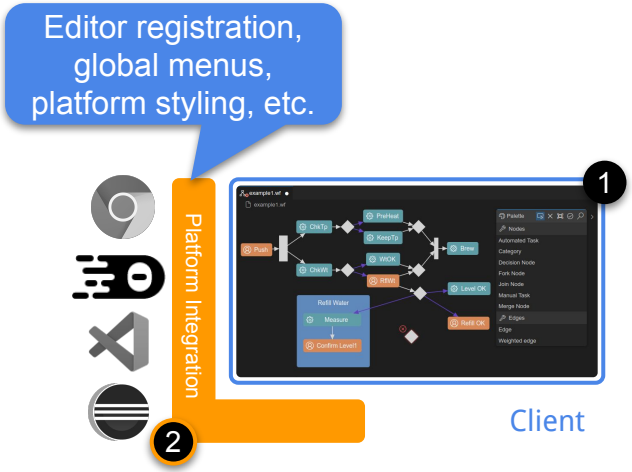
Functionality expected from the platform

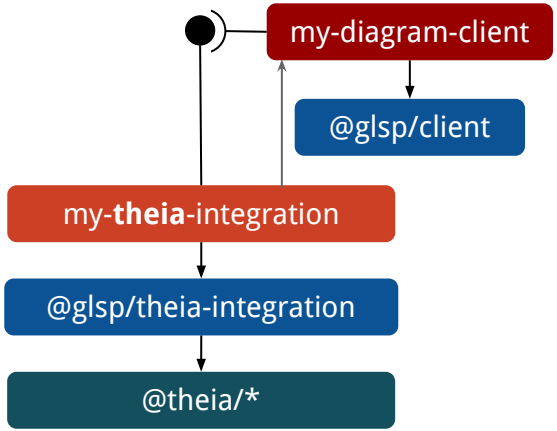
my-diagram-client

@glsp/client

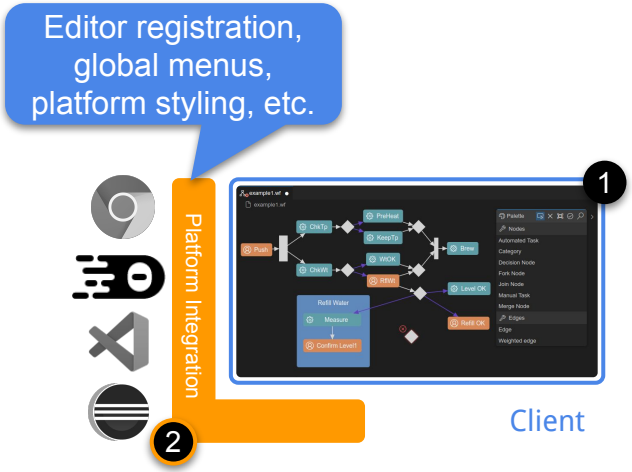


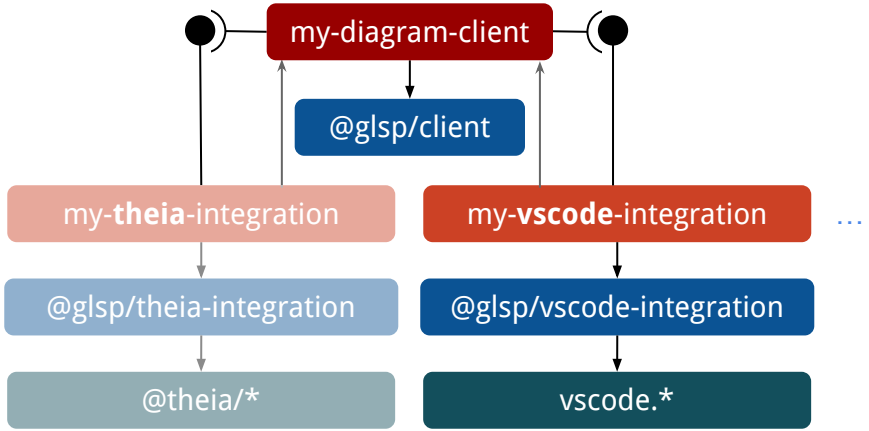
Maximum Reuse Across Tool Platforms





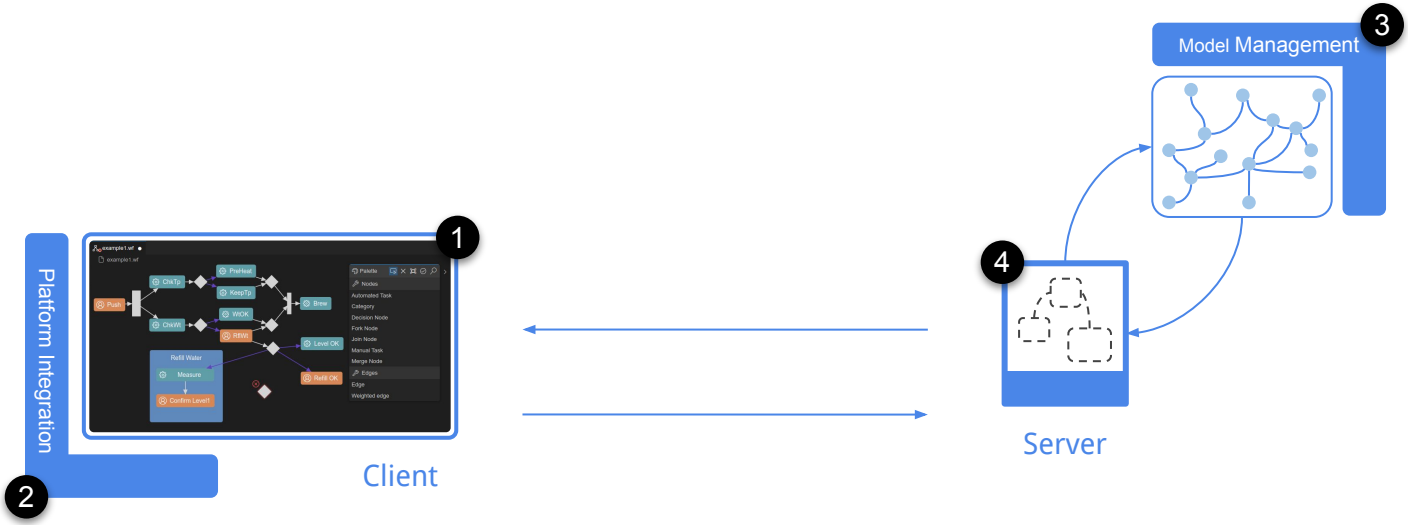
Maximum Reuse Across Tool Platforms





Demo

Flexibility on the Server



Model Management is Your Choice

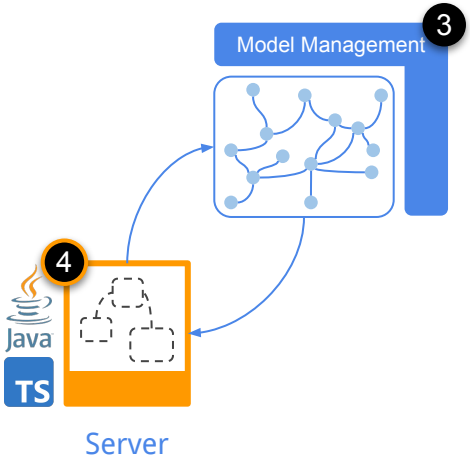
- Model Management
  - Format, structure, framework
  - Local filesystem or remote
  - Shared across users or isolated

- Base modules available
- Facilitates migration
- Enables reuse across multiple deployments

See Project Templates

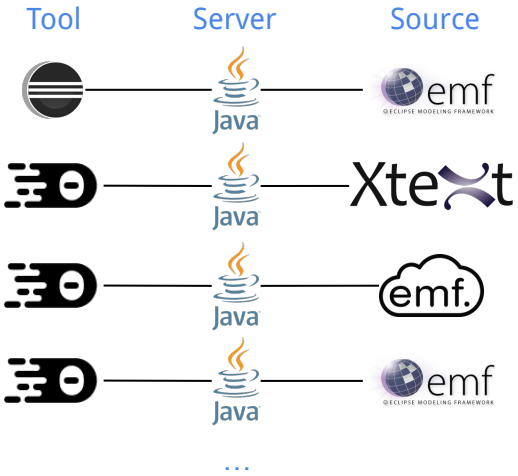
A new Server Framework alongside Java

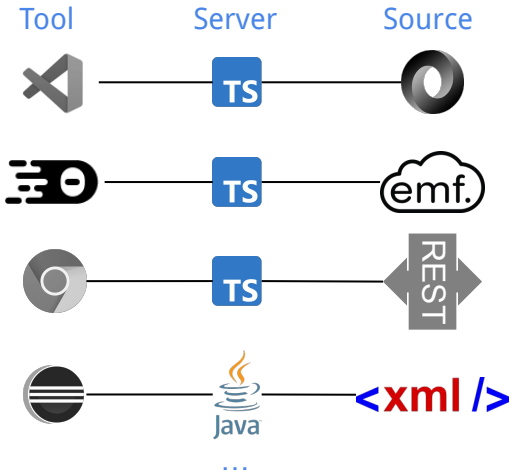
- GLSP servers can use any language
  - Protocol and some IO (e.g. socket)
  - More efficient to use existing base framework
- TypeScript GLSP Server Framework (1.0)
  - Homogeneous dev env for client & server
  - No need for a JVM on the user machine
  - Opens door to new deployment options



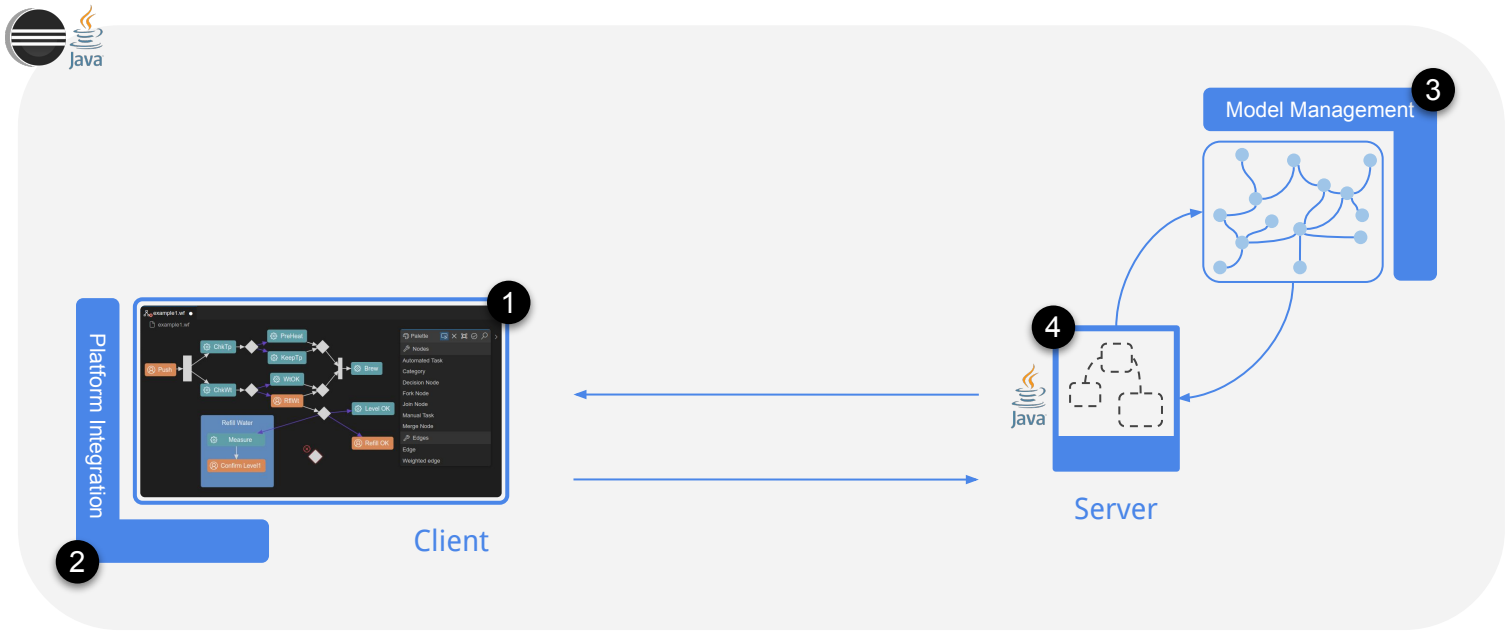
Demo

Deployment Options

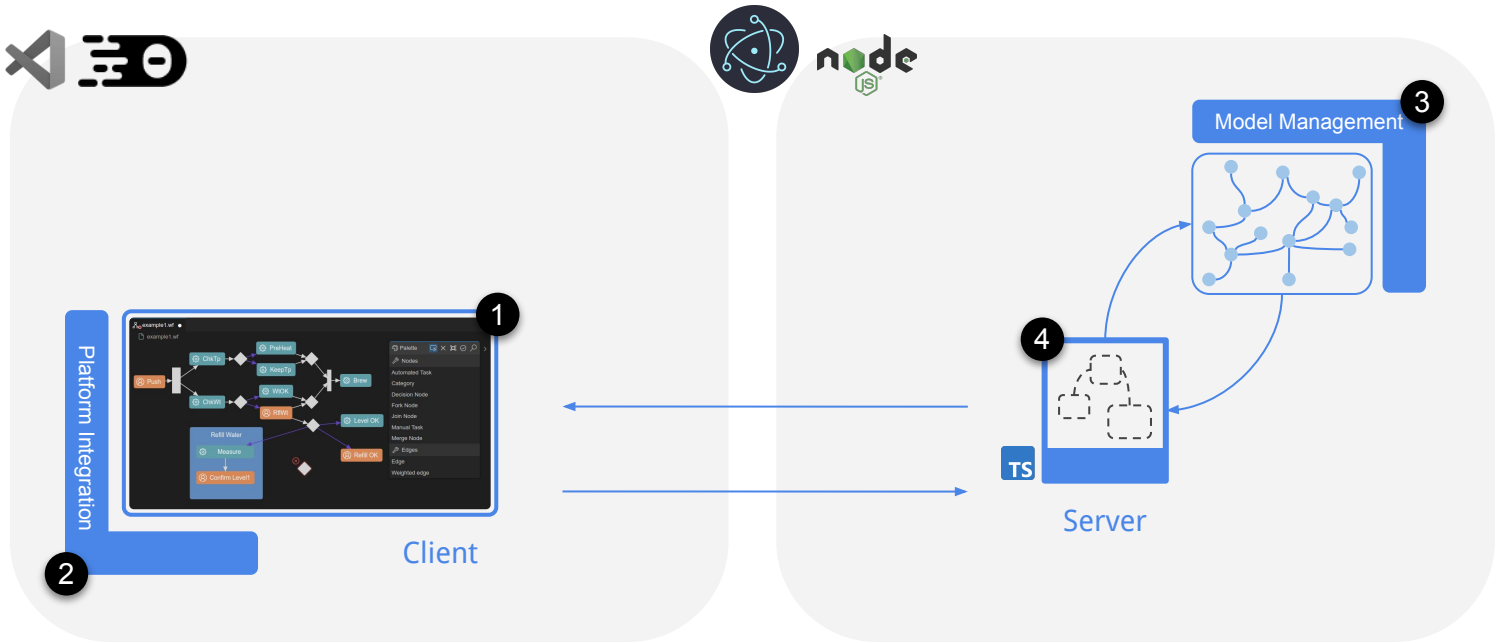




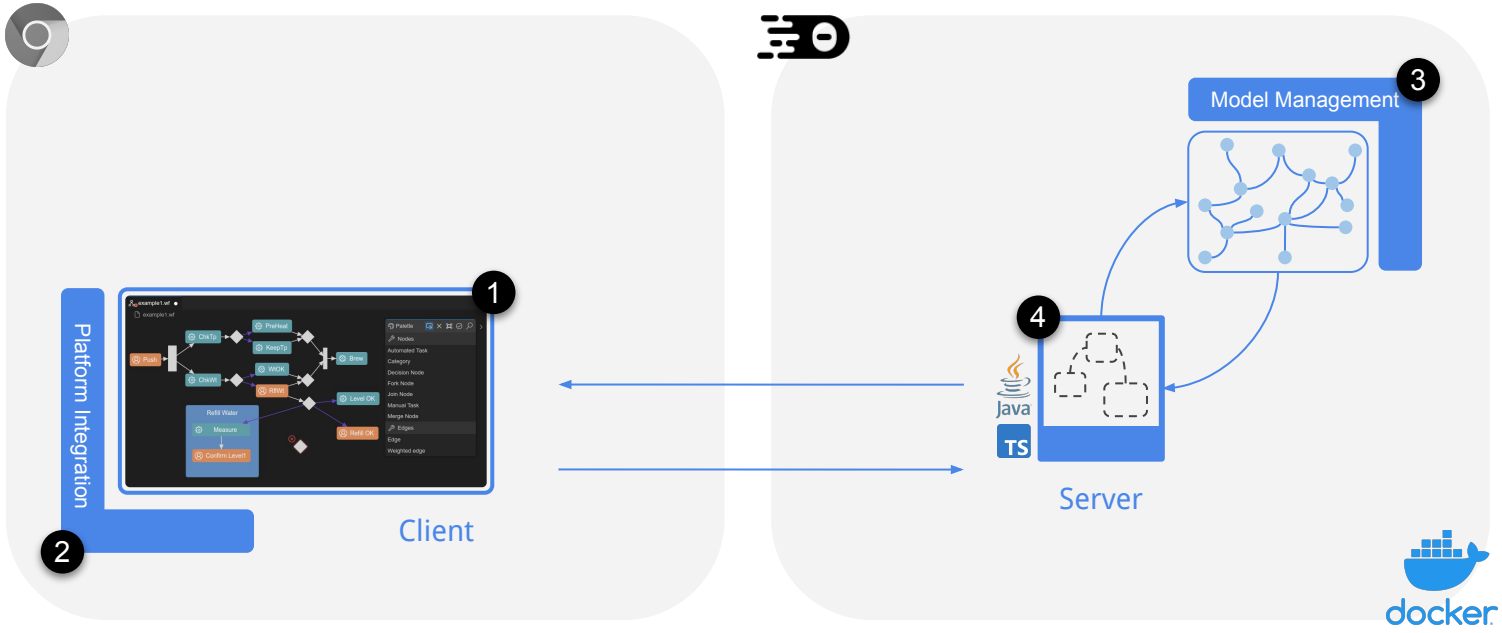
Deployment Options



Deployment Options

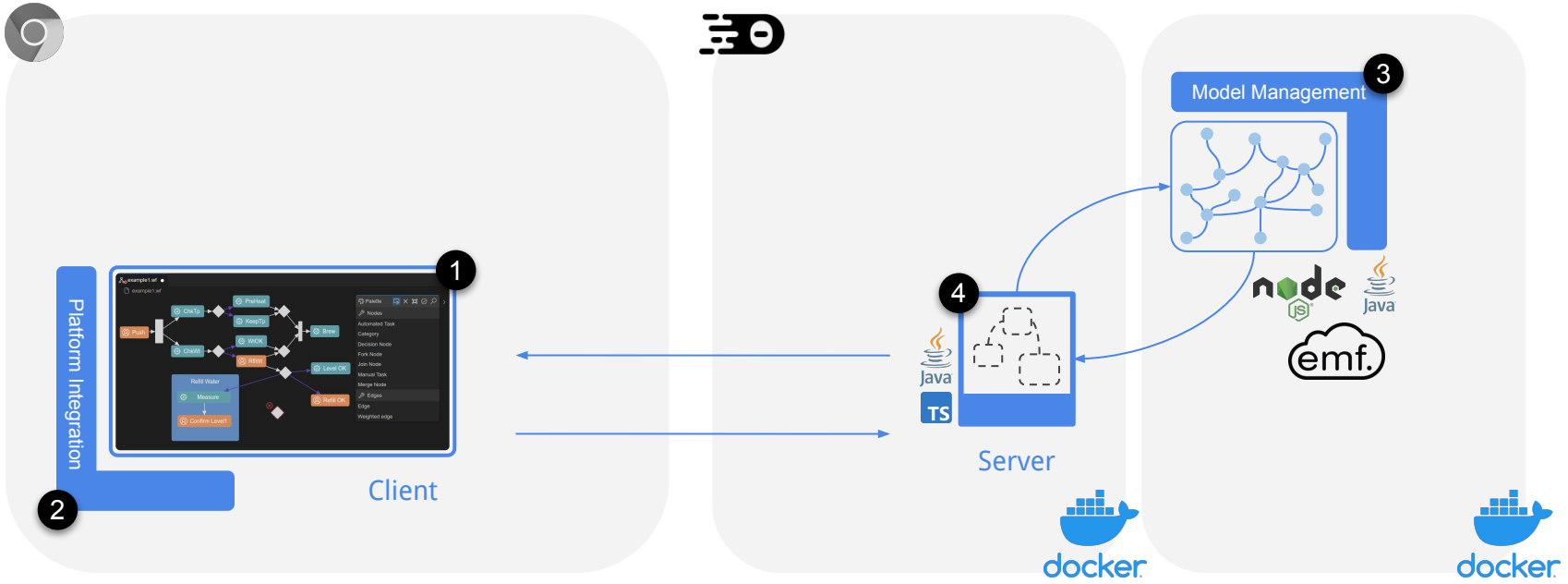


Deployment Options



Deployment Options

Theia.cloud - Running Theia-based products in the cloud
Tomorrow, 17:00 CET, Bürgersaal 2



Deployment Options

Demo

Conclusion

- Flexibility across all levels is key
  - Modern technology stack: more fluid and diverse
  - Multiple deployment / distribution channels
- Getting started is easier than ever
  - API stability with 1.0
  - Documentation
  - Project templates
- **Try it out and get in contact with us!**

  https://www.eclipse.org/glsp

Related Talks & Events

Getting started with Theia
The nextGen Eclipse Platform
Today, 10:45 CET

Introducing Eclipse CDT.cloud
C/C++ tooling in the web
Today, 11:45 CET

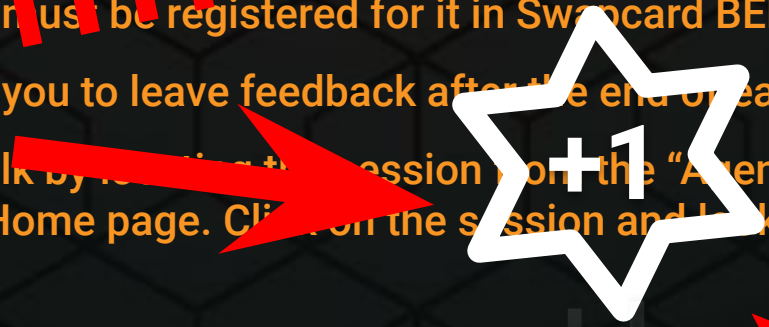
BoF: Building (web-based) tools with Eclipse
Today, 19:30 CET, Silchersaal

Theia.cloud - Running Theia-based products in the cloud
Tomorrow, 17:00 CET, Bürgersaal 2

Evaluate this Session

- Please help by leaving feedback on the sessions you attend!
- To rate a session you must be registered for it in Swapcard BEFORE the talk starts.
- Swapcard will prompt you to leave feedback after the end of each session.
- You may also rate a talk by selecting the session from the "Agenda" or "My Event" buttons on the Event Home page. Click on the session and look for the "Give your feedback" box.

WITH



ECLIPSE CON2022