A General Architecture for Client-Agnostic Hybrid Model Editors as a Service

Liam Walsh, Juergen Dingel, Karim Jahed

October 24th, 2022
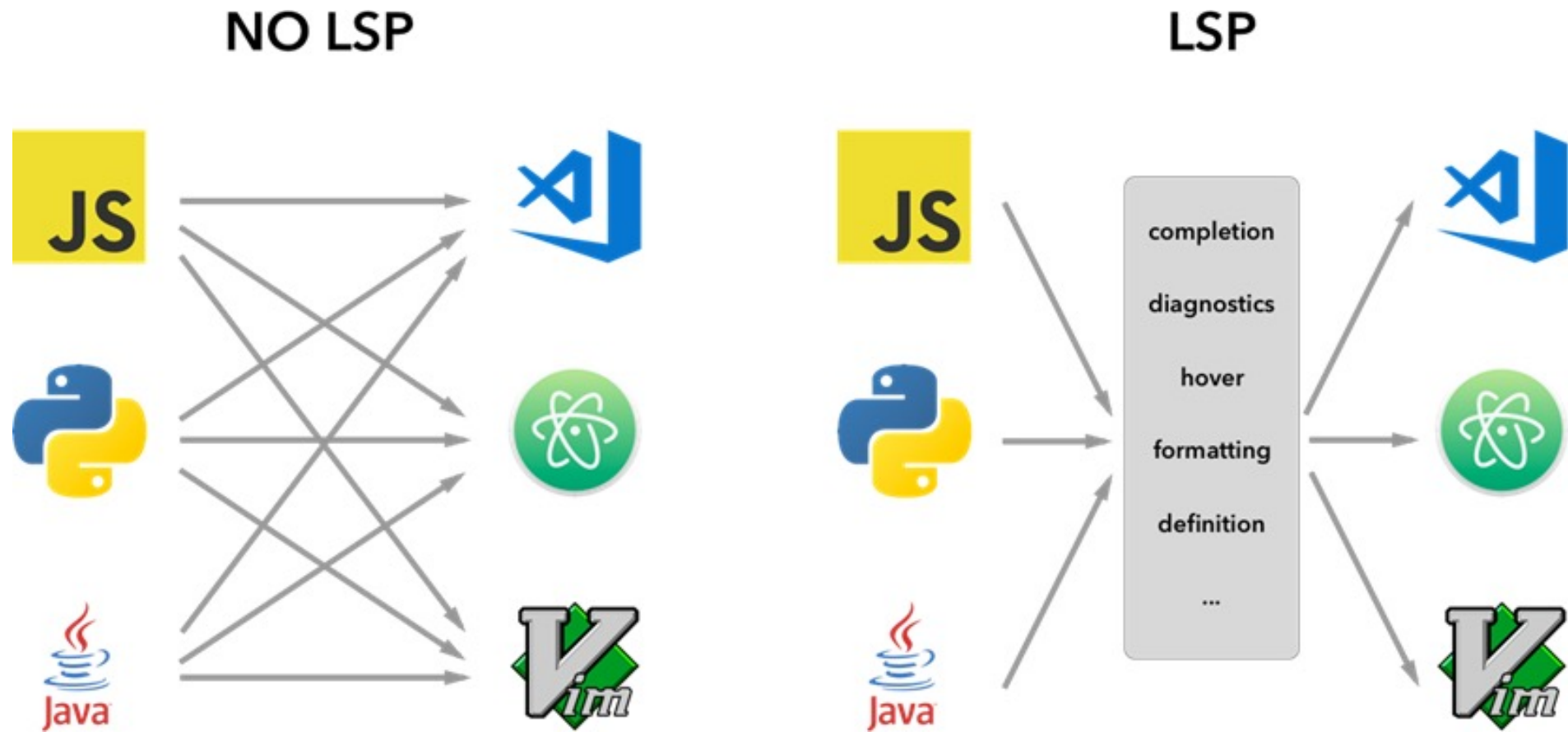
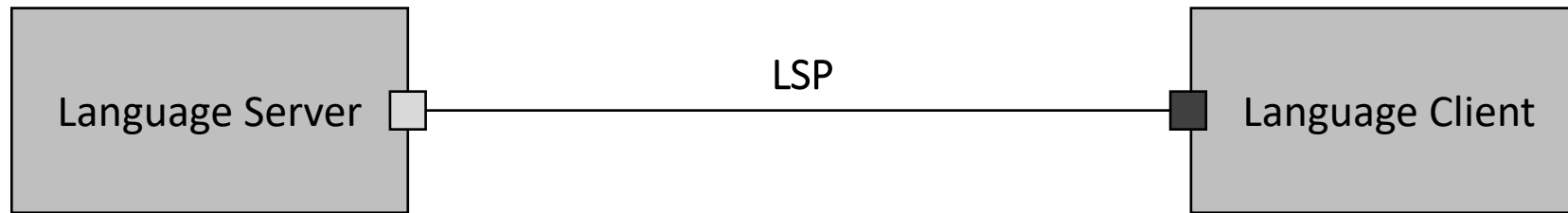Modeling Language Engineering

@ MODELS 2022

Problem

- Hybrid Modeling is generally tied to older platforms.

- Not supported by modern tools and technologies

Obtained from: *"Language Server Extension Guide", October 6, 2022,* code.visualstudio.com/api/language-extensions/language-server-extension-guide

Language Server

# GLSP (Graphical Language Server Protocol)

- Essentially LSP but made with graphical languages in mind
- Framework for both Servers and Clients
- Compatible with any IDE capable of rendering SVG structures

```
┌──────────────────┐                                    ┌──────────────────┐
│    Graphical     ├■──────── GLSP ──────────■┤    Graphical     │
│ Language Server  │                                    │ Language Client  │
└──────────────────┘                                    └──────────────────┘
```

Specific Goals

- Creation of a modern, platform-agnostic Architecture for Hybrid Modeling Language Editors

- One facilitated by Language Servers

- Leveraging of existing community resources on language servers

- Creation of Hybrid editors for existing textual Domain-Specific Languages

Hybrid Editor Behaviour (Required Definitions)
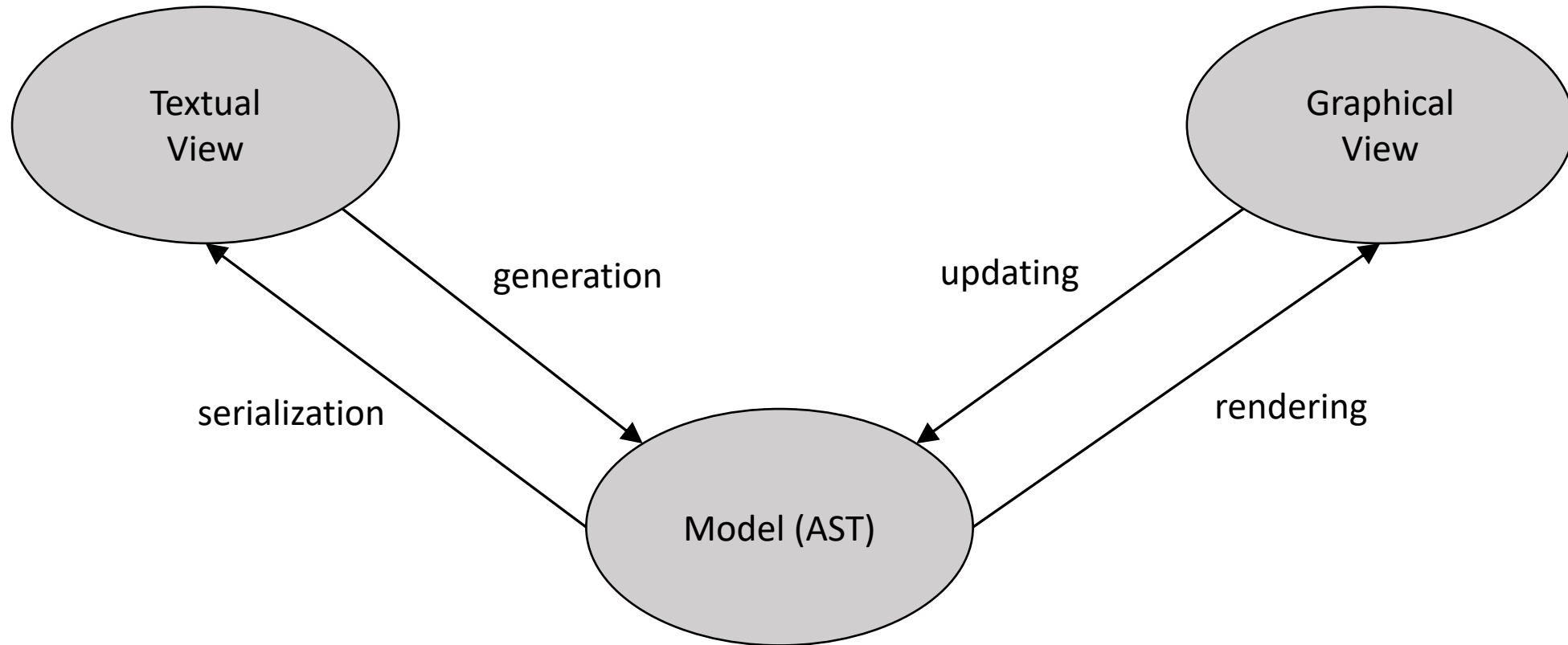
Hybrid Editor Behaviour (Illustration)

statemachine **selfLoop**
    initial state init;
    state s1;
    transition t0
        init => s1;
    transition t1
        s1 => s1;

generation

updating

selfLoop

init

t1

t0

s1

serialization
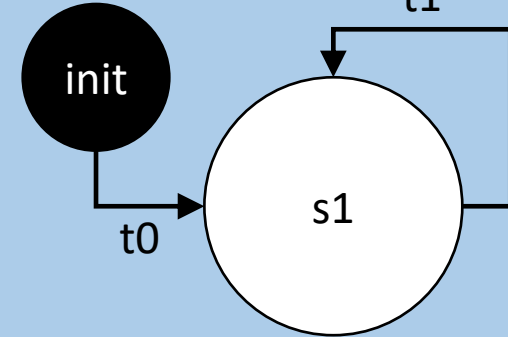
rendering

<StateMachine id="selfLoop">
<PseudoState id="init"/>
<State id="s1"/>
<Transition id="t0"
      source="init" target="s1"/>
<Transition id="t1"
      source="s1" target="s1"/>
</StateMachine>

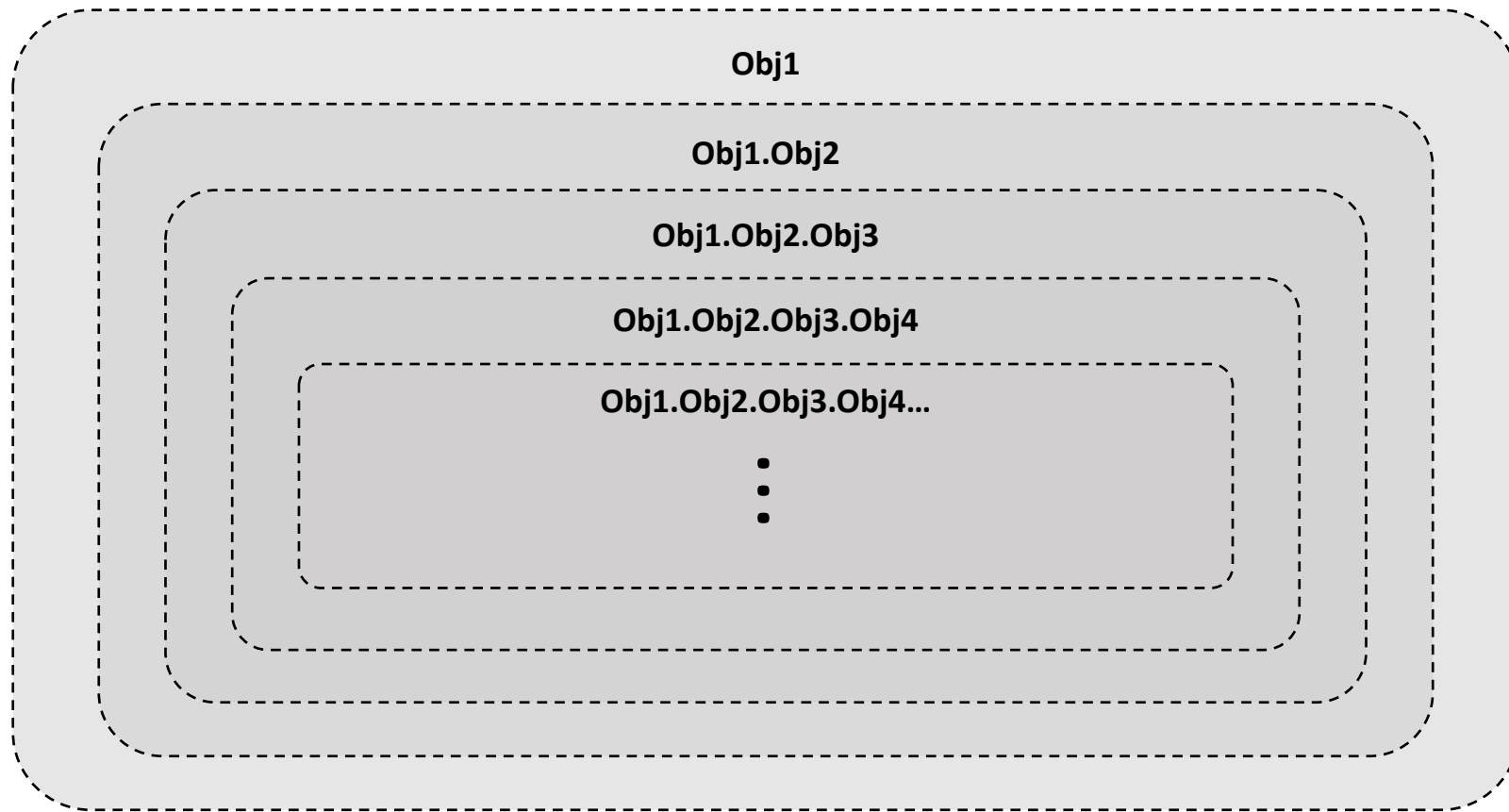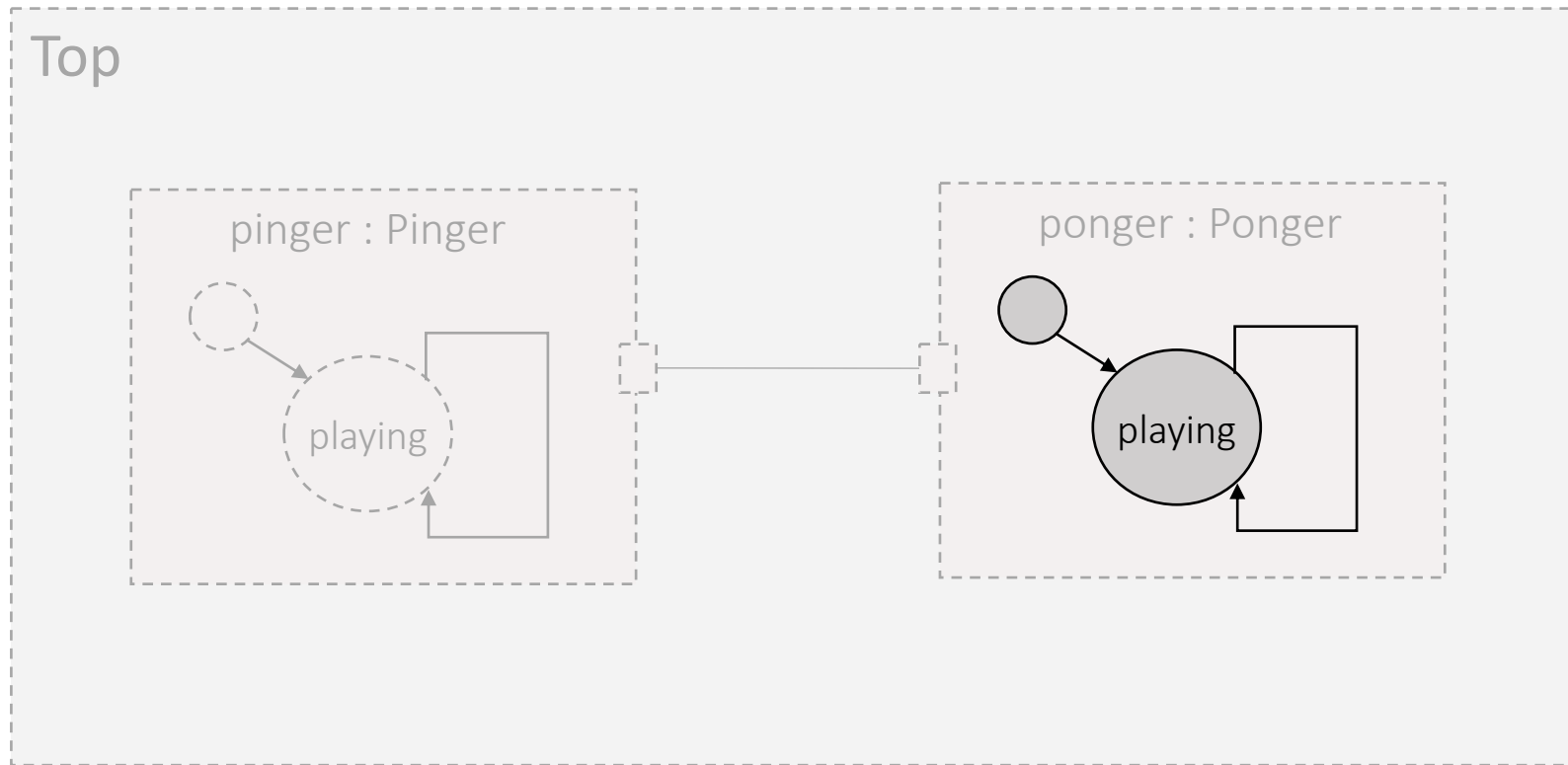# Approach to Architecture Design
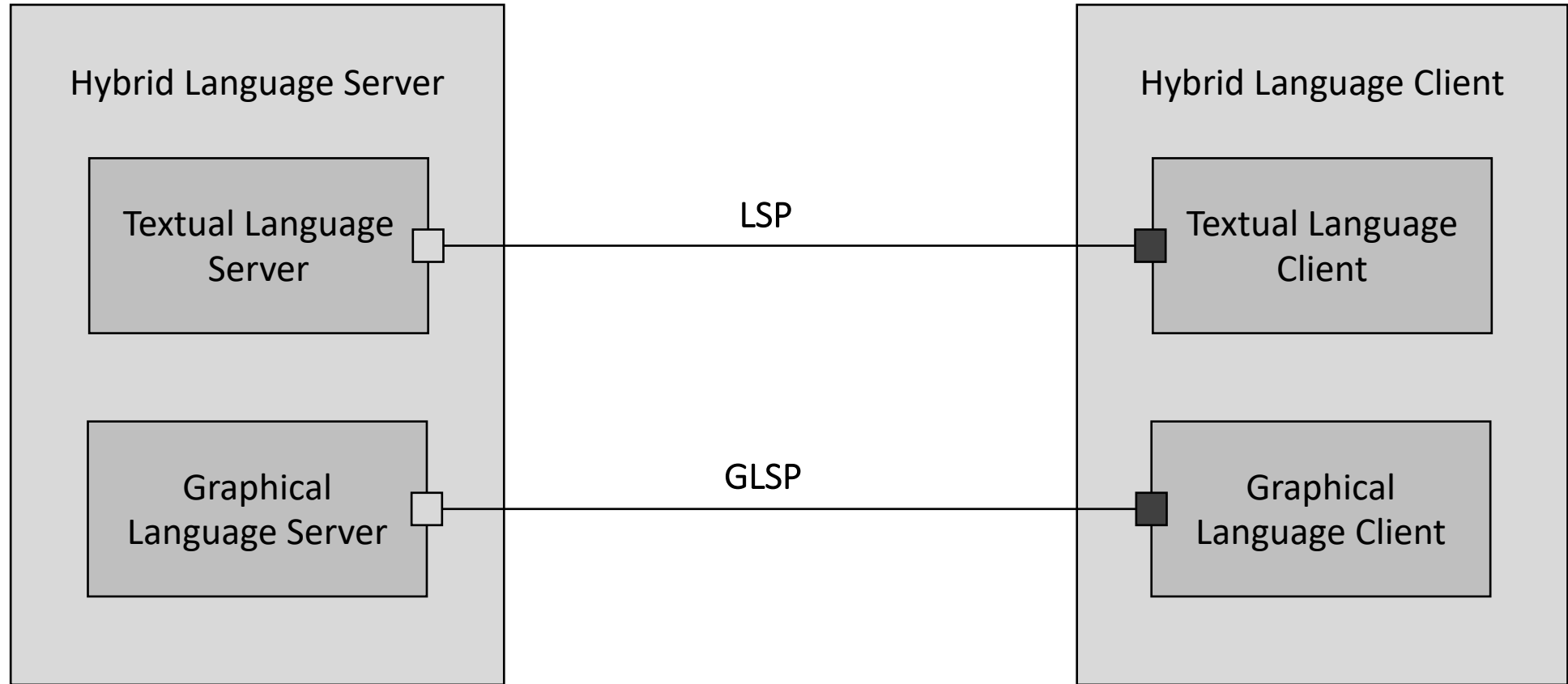
Proposed Architecture
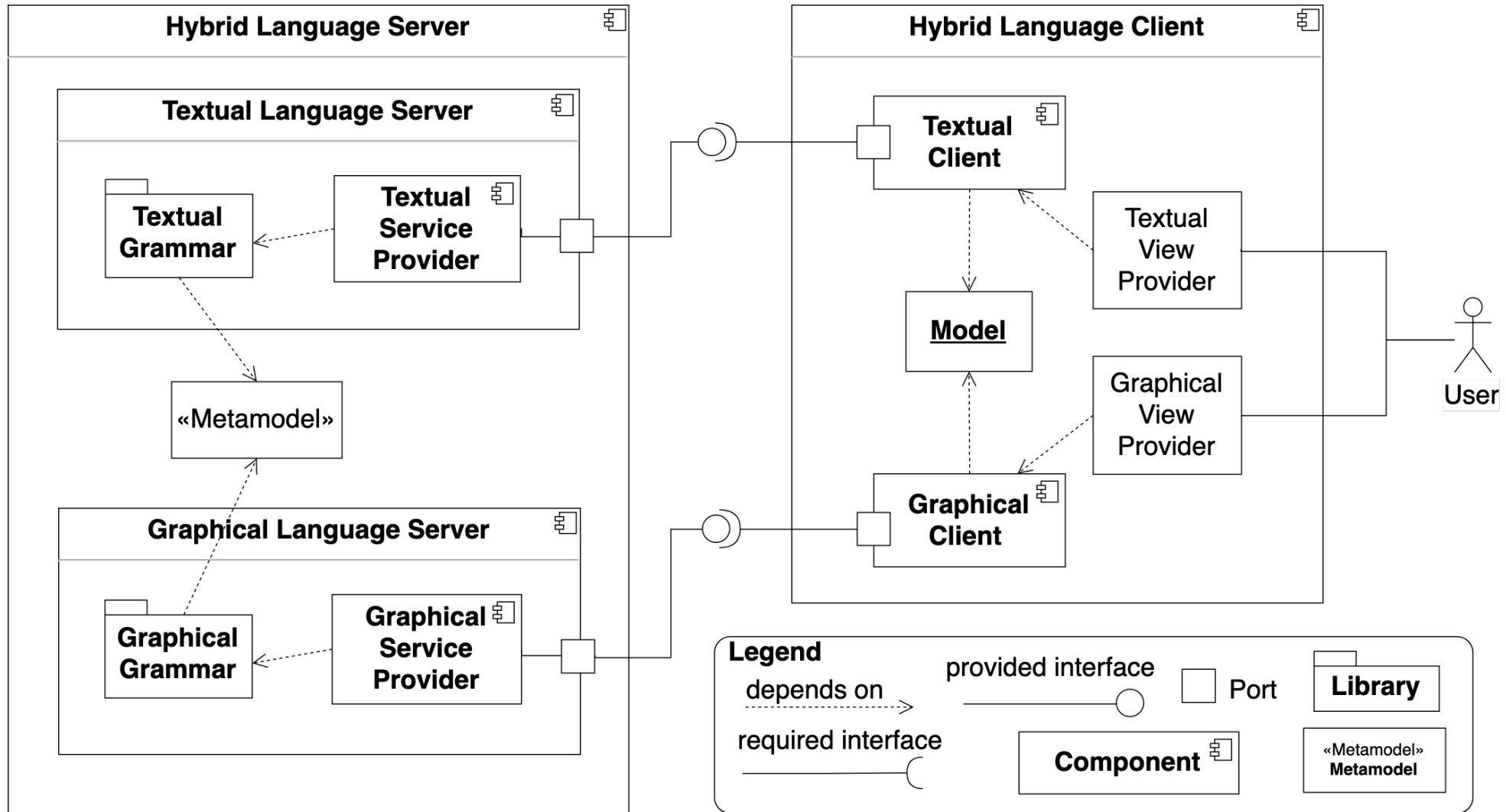
Single-View Limitations

- Graphical languages are designed around graphical editing capabilities
- Hybrid languages may go beyond what is specifiable using a graphical editor
- Single-view graph representation of a language is insufficient
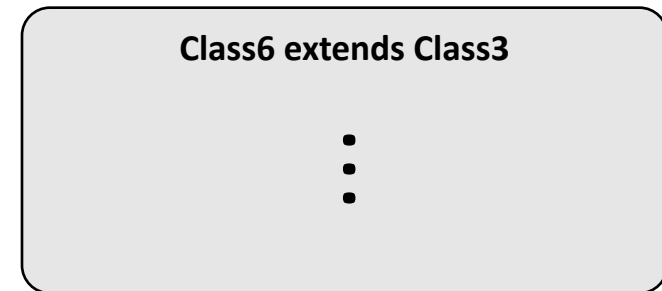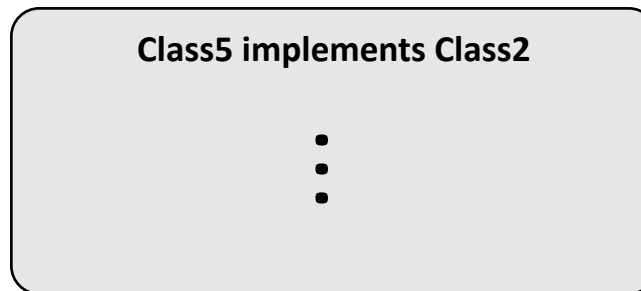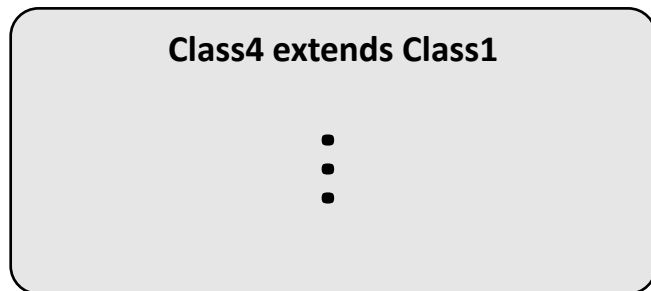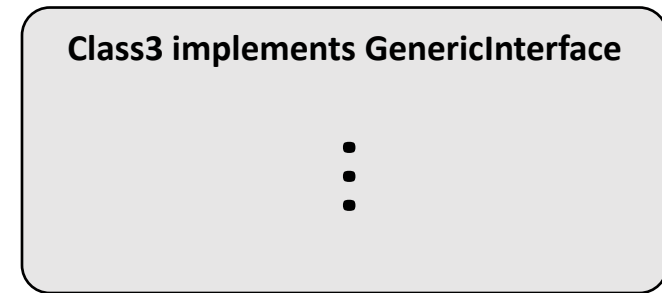
Containment

Projection

Static Analysis

Find all usages of GenericInterface:

Class1 implements GenericInterface

.
.
.

Interface2 extends GenericInterface

.
.
.

Class3 implements GenericInterface

.
.
.

Class4 extends Class1

.
.
.

Class5 implements Class2

.
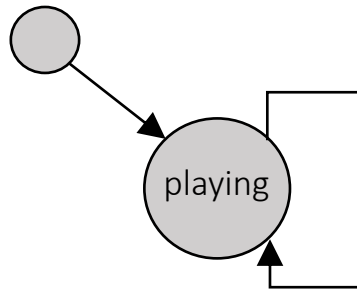.
.

Class6 extends Class3

.
.
.
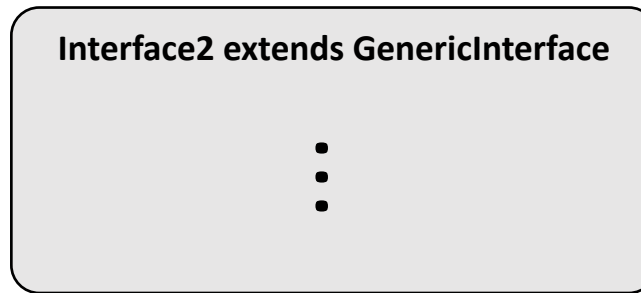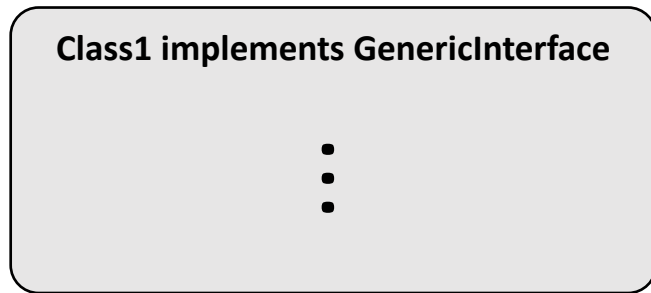
Multiple Graphical Views
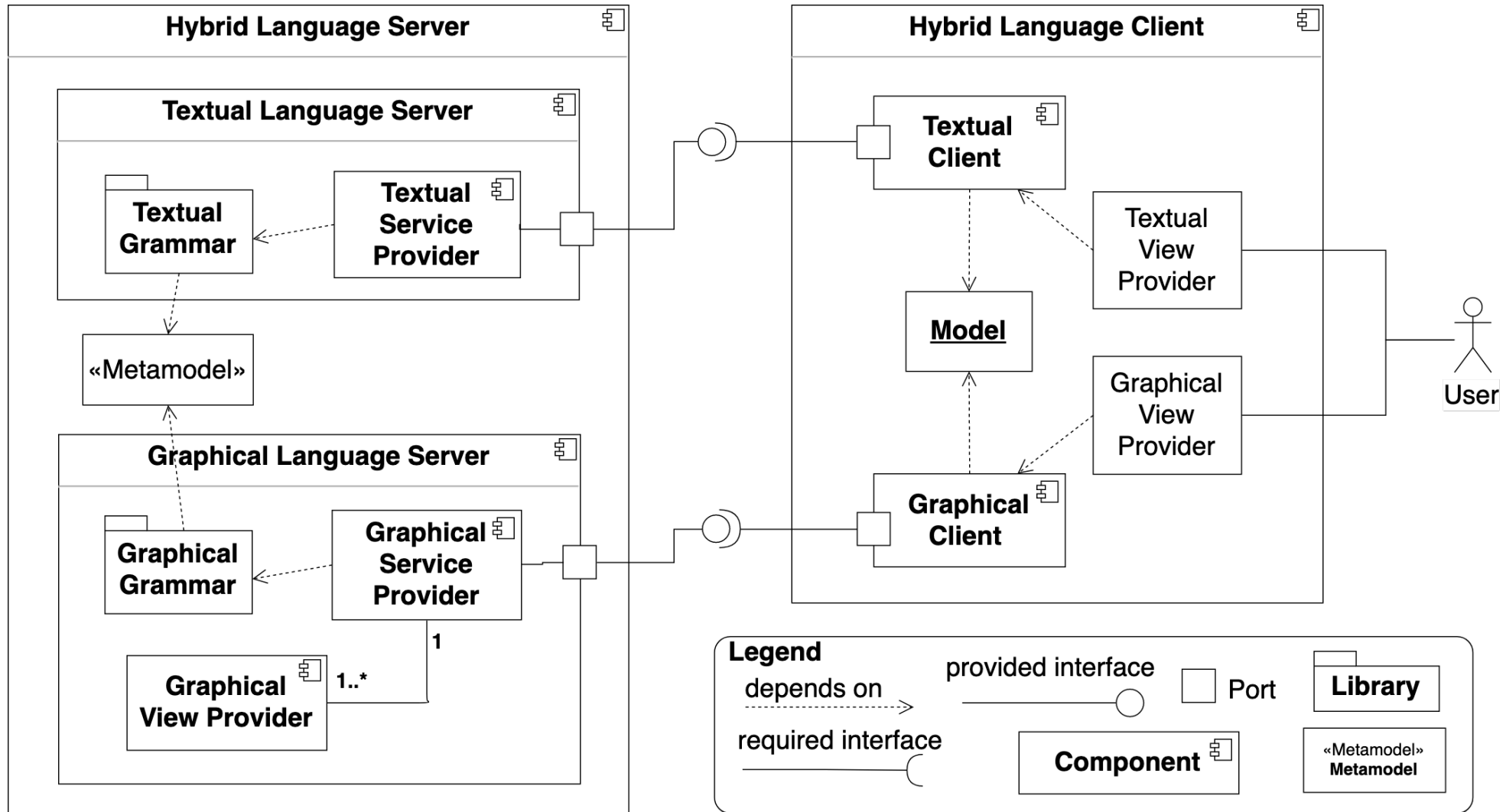
[Structure] Top
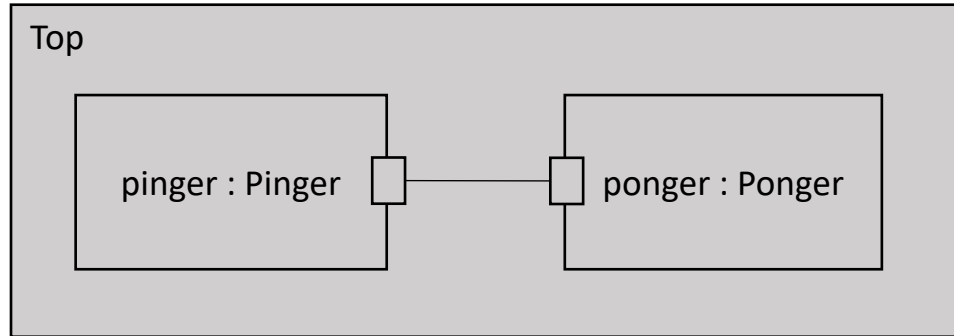
[Behaviour] Pinger

- Visual clutter is managed by distributed views
- Issues with complex views are sidestepped by starting small and aggregating

Revised Architecture

Conclusion

- A general architecture for hybrid modeling languages is feasible in the context of language servers

- Hybrid languages are not entirely analogous to purely textual and purely graphical languages

- This architecture will ideally inform future work in further standardizing the implementation of such languages.

Thank you!