Transparency of Tools
Beyond Usability in Modeling Tools

Alfonso Pierantonio

SWEN / Software Engineering Research Group

Università degli Studi dell’Aquila, Italy

Context

Modeling tools are fundamental enablers in MDE

- They provide the environment for creating, manipulating, transforming, and managing domain-specific notations

- In education, they facilitate abstraction, guiding students from concrete thinking to higher-level modeling across levels

Bencomo, N., Cabot, J., Chechik, M., Cheng, B. H., Combemale, B., Wąsowski, A., & Zschaler, S. (2024). Abstraction Engineering. arXiv preprint arXiv:2408.14074.

Objective

This presentation aims to

- Analyze the limitations of current modeling tools and their impact on usability, accessability, and efficiency
- Explore key characteristics that address existing limitations and prepare them for future advancements

Foundational concepts in philosophy and cognitive psychology can offer new perspectives

Martin Heidegger (1889 – 1976)
Phenomenology, Philosophy of Technology

Directly defines tool transparency (ready-to-hand)

Jean Piaget (1896–1980)
Developmental and Cognitive Psychology

Defined the concept of cognitive schemata

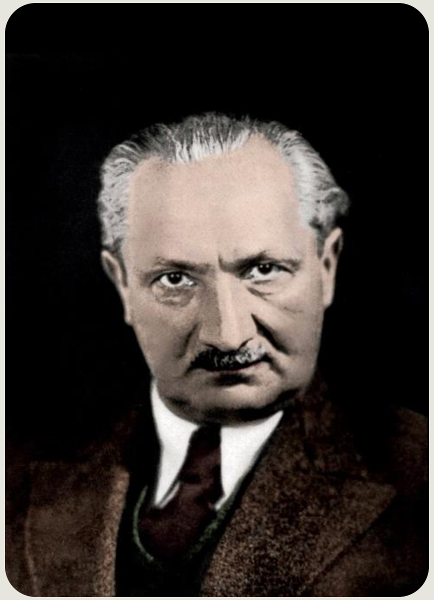
A tool is transparent when its users develop a cognitive schema

From Heidegger’s tool transparency to Piaget’s cognitive schemata—understanding how tools shape thought and action

Philosophical Foundations

Martin Heidegger (**1889–1976**) provides the most direct philosophical foundation for transparency in tools

- In his philosophy, tools (or equipments) are not merely objects, but mediators that shape human interaction with the world



Heidegger, M. (1927) Sein und
Zeit. Halle: Max Niemeyer Verlag

Classification of Tools (ready-to-hand)

Heidegger suggests how tools can either facilitate or hinder user engagement

- A tool is ready-to-hand when it seamlessly integrates into the user’s actions as an extension of their capabilities, allowing full focus on the task without conscious thought
- For a tool to become ready-to-hand, the user must develop or adapt their cognitive schemata

A Flushing System

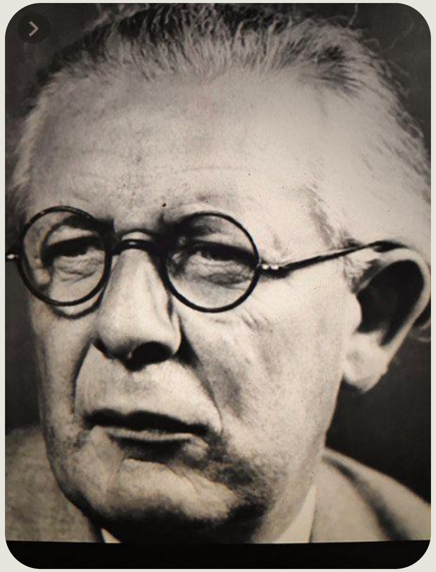
The interface for flushing the toilet is immediately clear and intuitive

- The buttons are designed so that users interact with them without conscious thought
- The flushing system is ready-to-hand !

Schemata in Cognitive Psychology

There is a strong conceptual parallel between being ready-to-hand and cognitive schemata

- Cognitive schemata reside in our knowledge system, shaping how we perceive, learn, and make decisions
- They are constantly being created, adapted, and reorganized as we interact with the world



Piaget, J. (1926). The Language and Thought of the Child. London: Routledge & Kegan Paul.

Classification of Tools (present-at-hand)

Heidegger suggests how tools can either facilitate or hinder user engagement

- A tool is present-at-hand when it becomes the focus of attention rather than an extension of action
  - The tool malfunctions or breaks, requiring counscous effort to understand
  - The user is unfamiliar with how to operate it
  - The tool design is unintuitive, creating frictions that disrupt workflow
- The tool is an object of concern

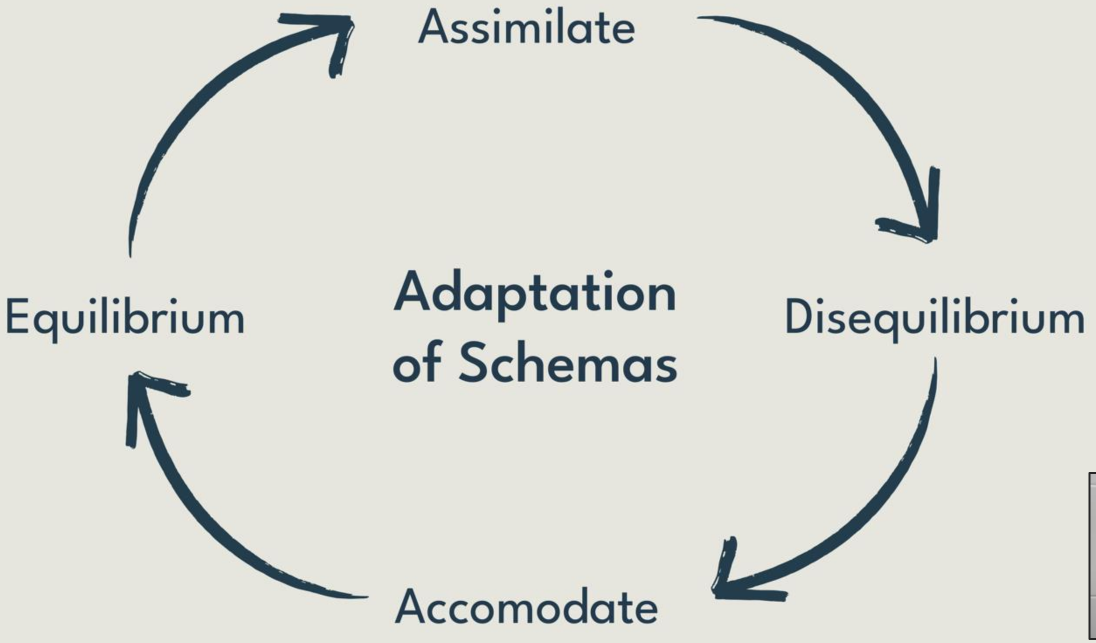
When Tools Disrupt the Task

- The operation is less intuitive and requires reflection and understanding how to proceed
- The focus is no longer on the task but on how to operate the system
- As a conseguence, user must adapt their cognitive schemata

The Process of Adaptation

Intellectual growth is a process of adaptation to the world

new information are classified according to existing schemata



schemata explain what it is perceived, the user reach a state of cognitive

new information clashes with existing schemata, causing discomfort

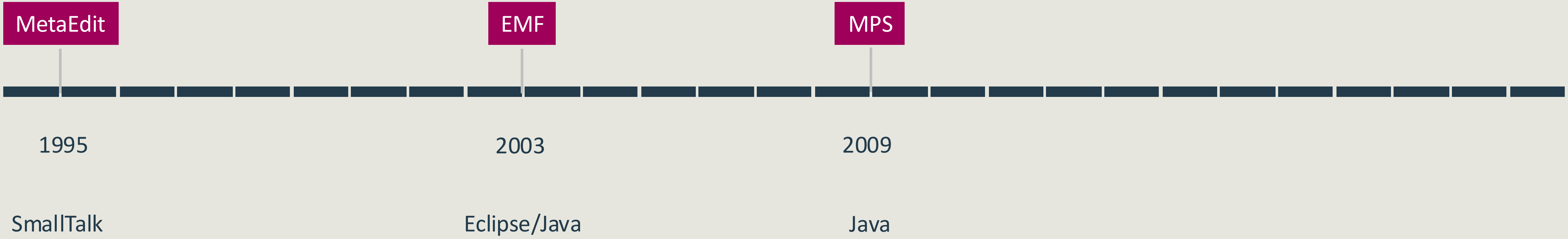
existing schemata are revised to incorporate new information

Piaget, J. (1952). The Origins of Intelligence in Children. New York: W.W. Norton & Company.

When Do Tools Become an Obstacle?

- outdated technology stacks limit flexibility, and usability
- lack of integration limits governance
- accidental complexity adds complications
- steep learning curves slow adoption
- emphasis on tools hides core modeling principles

A Dirsruptive Timeline



MetaEdit

1995

SmallTalk

EMF

2003

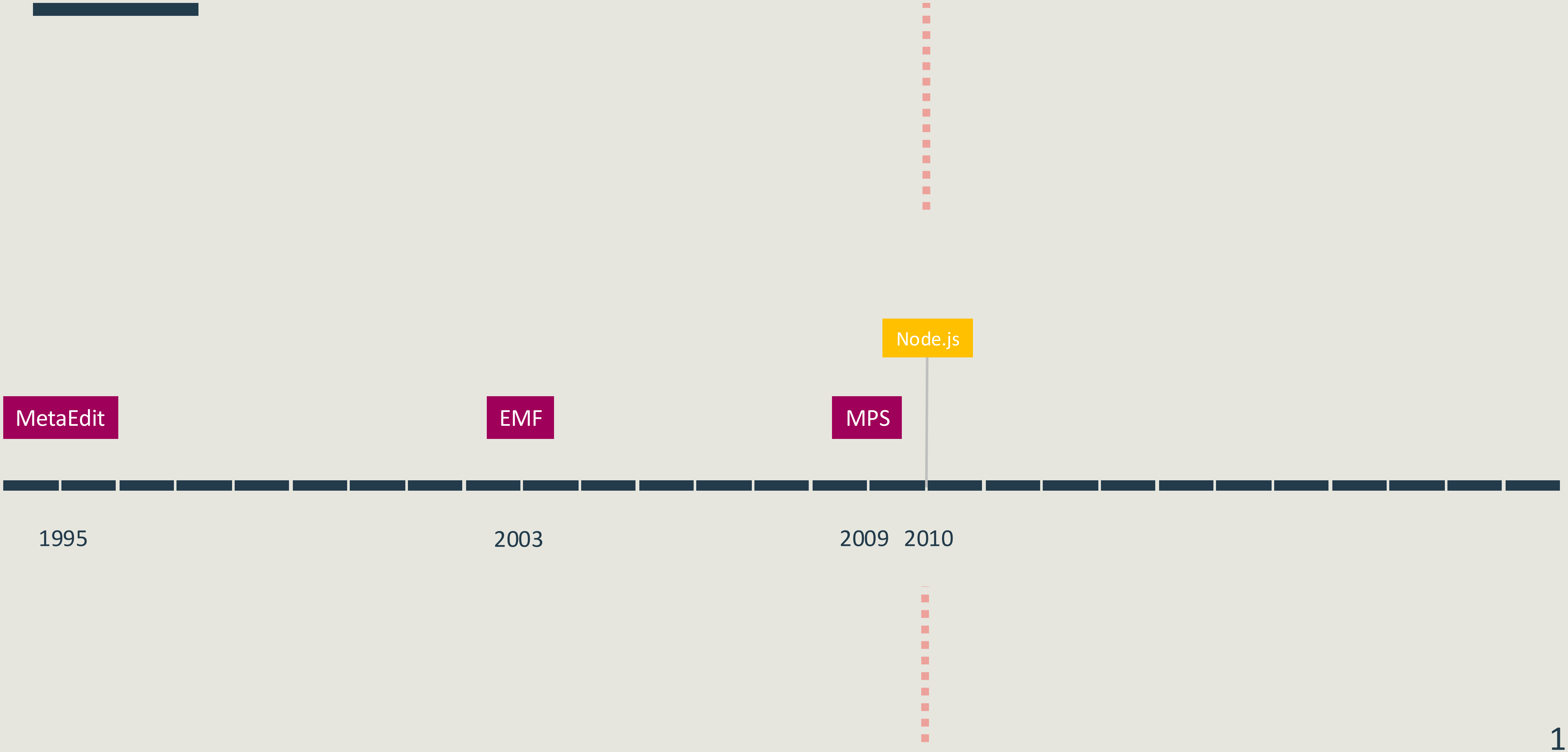
Eclipse/Java

MPS

2009

Java

A Dirsruptive Timeline



MetaEdit

EMF

MPS

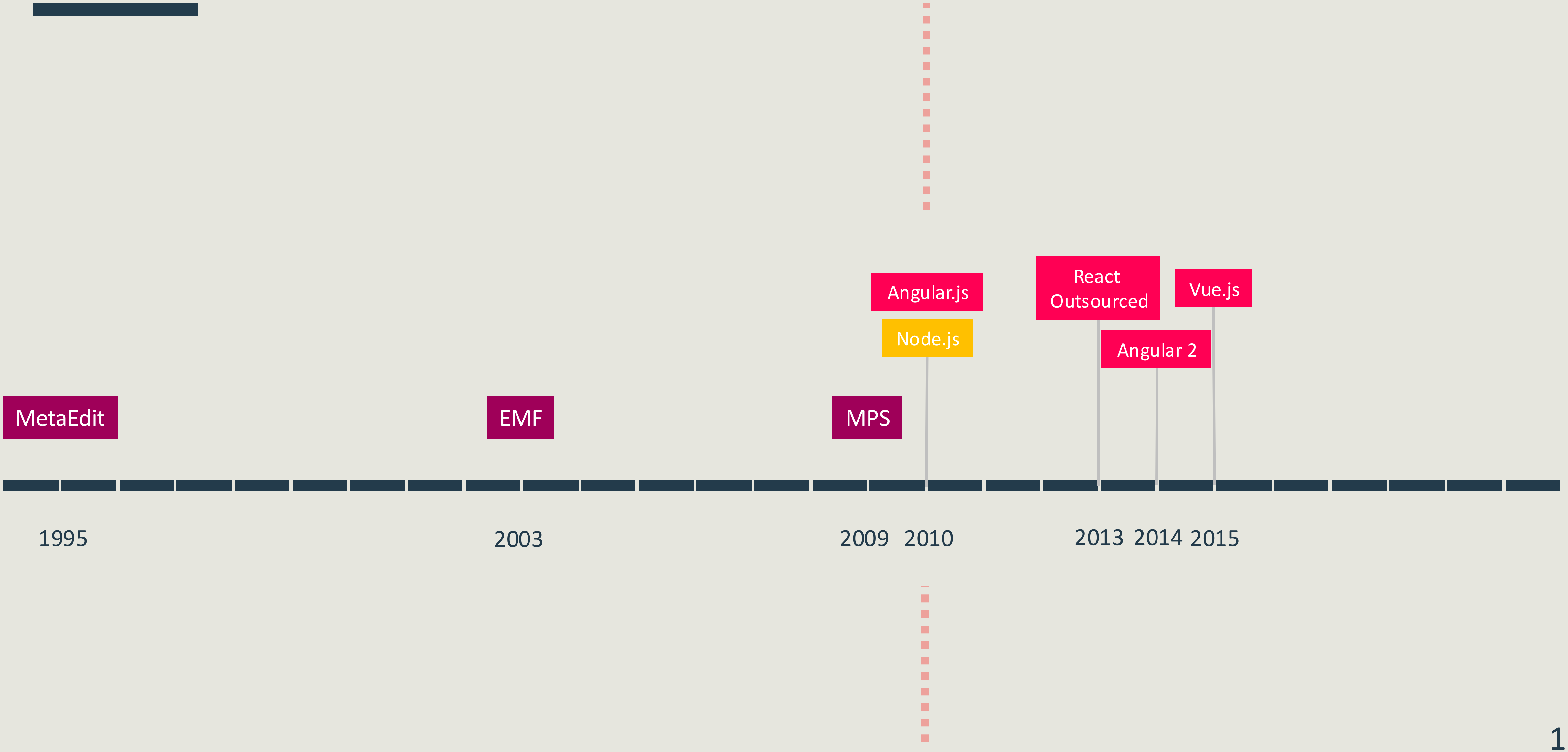
Node.js

1995

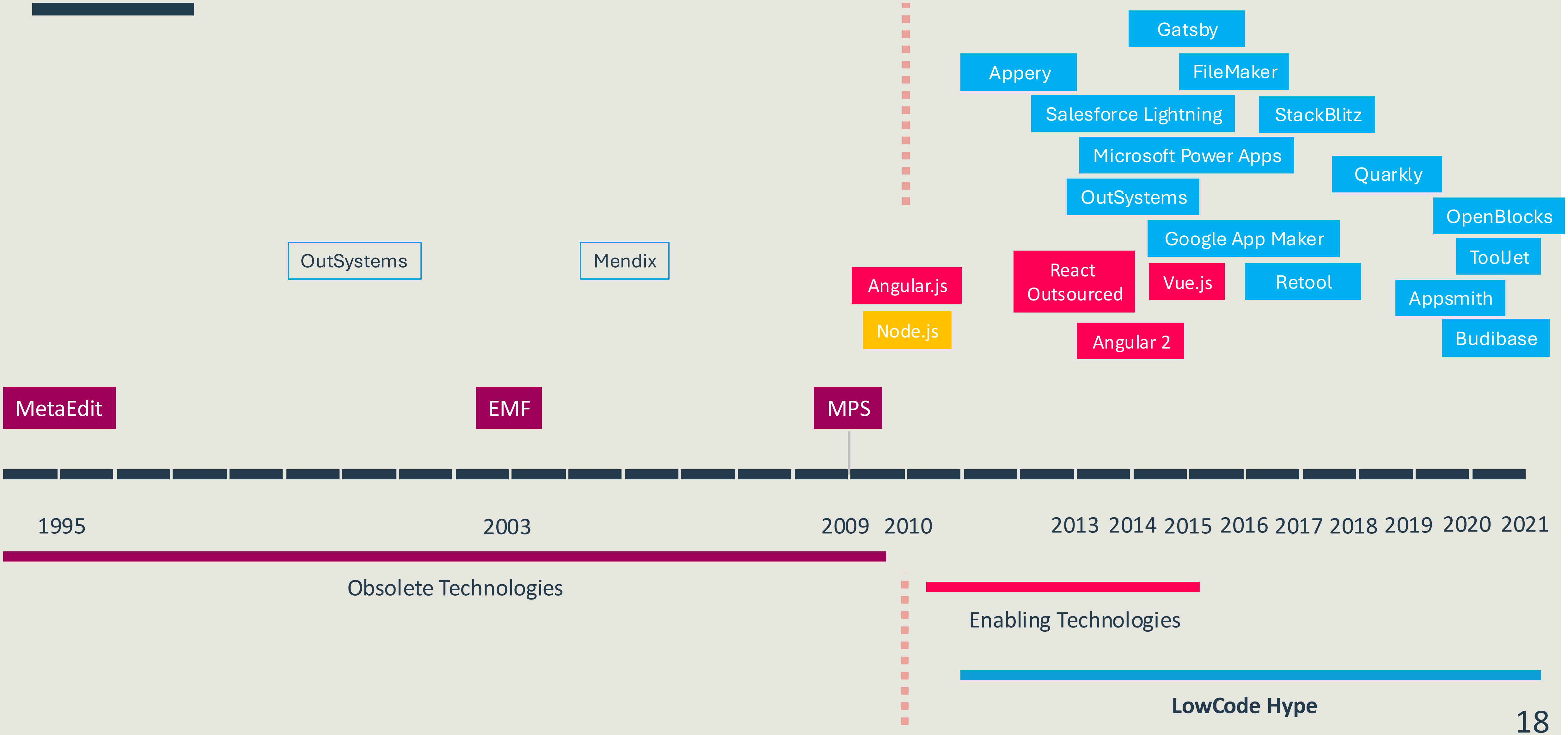
2003

2009 2010

A Dirsruptive Timeline



A Dirsruptive Timeline



MetaEdit

OutSystems

EMF

Mendix

MPS

Angular.js

Node.js

React Outsourced

Angular 2

Appery

Gatsby

FileMaker

Salesforce Lightning

StackBlitz

Microsoft Power Apps

OutSystems

Google App Maker

Vue.js

Retool

Quarkly

OpenBlocks

TooUet

Appsmith

Budibase

1995 2003 2009 2010 2013 2014 2015 2016 2017 2018 2019 2020 2021

Obsolete Technologies

Enabling Technologies

LowCode Hype

Component-based vs Integrated Environment

In the landscape of MDE tools, we distinguish two major architectures

- Component-based systems: EMF
- Integrated systems: MPS, MetaEdit+, Jjodel

In addition, both EMF and MPS are open-source but with different organizational models

Generative vs Reflective platforms

Two approaches

- In generative approaches, tools are typically created through the following pipeline, eg EMF, MPS

Design > Generate > Compile > Deploy

- In reflective approaches, the platform reflects on its own properties and adapts its behavior accordingly, eg MetaEdit+, Jjodel

Current Stacks in Modeling Tools

Platform	Technology Stack	Year	Integration	Reflective	Cloud/SaaS	Built-in Governance	UIX Awareness
MPS	Legacy (Java-based)	2009	Partial	Generative	Limited/No	Yes	Basic
EMF	Legacy (Eclipse)	2004	Weak	Generative	No	No	Minimal
MetaEdit+	Legacy (Smalltalk)	1995	Strong	Reflective	No	Yes	Intermediate
jjodel	Modern (Cloud-based)	2024	Strong	Reflective	Yes/Yes	Yes	Advanced

From 4GL to Low-Code

The transition from 4GL in 1980s to modern Low-Code Development Platforms (LCDP) should be in-depth analyzed

- What once miserably failed has now succeeded in a disruptive manner, driven by socio-technical aspects and emerging new technologies

Low Code Benefits

Lower barrier to entry & deployment costs

Shorter development cycle

Reduced maintenance burder

Enhanced customer experience

Improved productivity

Strong built-in governance

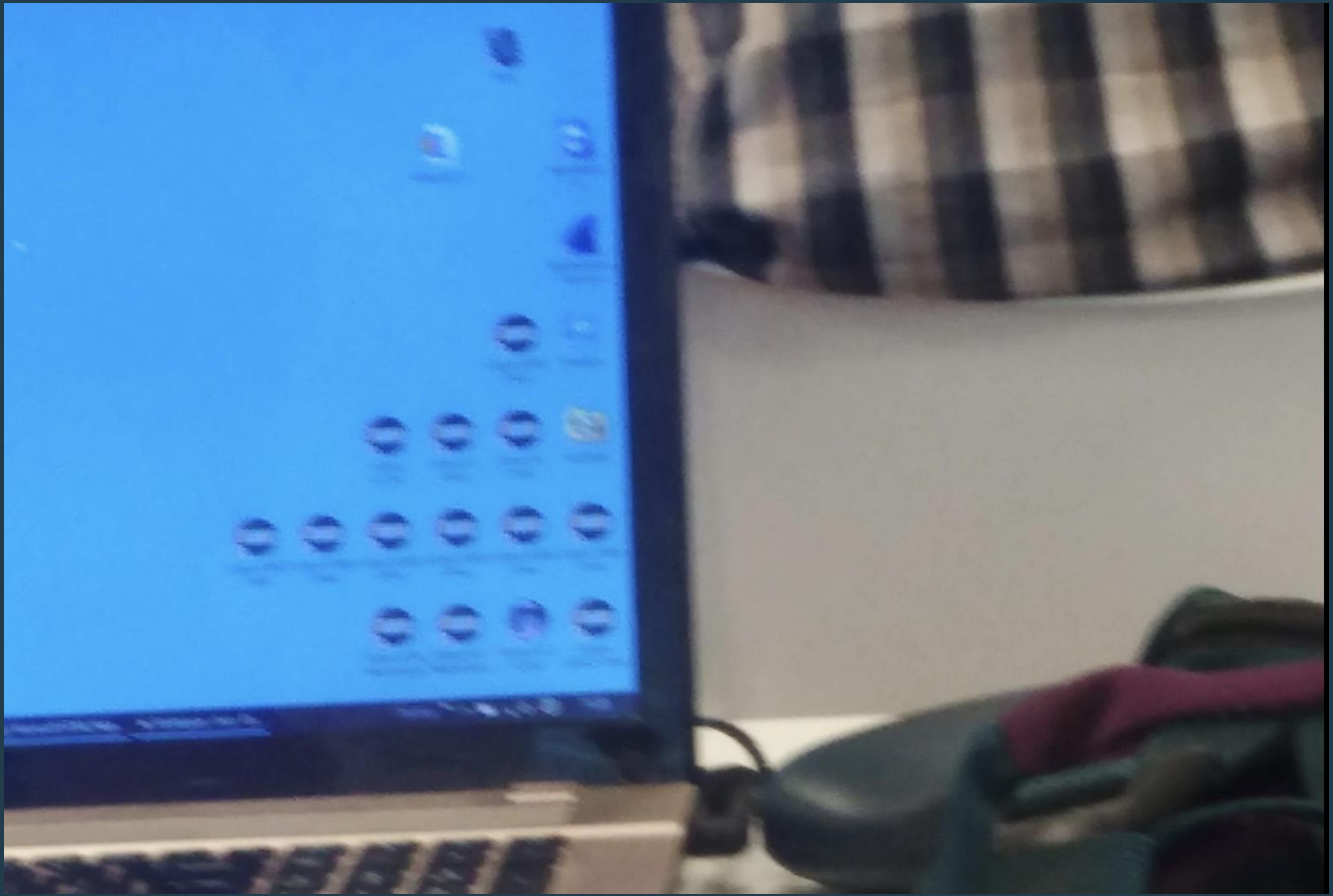
Rapid prototyping

Software development democratization

Low Code Benefits

«zero-setup»
(SaaS)

- Lower barrier to entry & deployment costs
- Shorter development cycle
- Reduced maintenance burder
- Enhanced customer experience
- Improved productivity
- Strong built-in governance
- Rapid prototyping
- Software development democratization



14 Eclipse instances

(Picture taken during STAF 2015, Wien)

Low Code Benefits

Enhanced Usability

Lower barrier to entry & deployment costs

Shorter development cycle

Reduced maintenance burder

Enhanced customer experience

Improved productivity

Strong built-in governance

Rapid prototyping

Software development democratization

Low Code Benefits

Reduced Cognitive Load

Lower barrier to entry & deployment costs

Shorter development cycle

Reduced maintenance burder

Enhanced customer experience

Improved productivity

Strong built-in governance

Rapid prototyping

Software development democratization

Low Code Benefits

Integrated Environment

Lower barrier to entry & deployment costs

Shorter development cycle

Reduced maintenance burder

Enhanced customer experience

Improved productivity

Strong built-in governance

Rapid prototyping

Software development democratization

What MDE Can Learn From Low-Code

The following aspects have been identified

- Generic vs. specific platforms
- Opening up web/cloud-based platforms
- Counteracting vendor lock-in
- Managing software evolution
- Fostering ecosystems

Other considerations are missing, nothing is said about the Technology Stack and Software Delivery Model (eg Saas)

Di Ruscio, D., Kolovos, D., de Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Correction to: Low-code development and model-driven engineering: Two sides of the same coin?. Software and Systems Modeling, 21(5), 1687-1687.

What is Jjodel?

A modeling SaaS platform designed to make MDE more accessible, transparent, and flexible

- Built around the principle of tool transparency
- Strengthened support for built-in governance, including co-evolution
- Syntax beyond topological notations
- Collaborative modeling

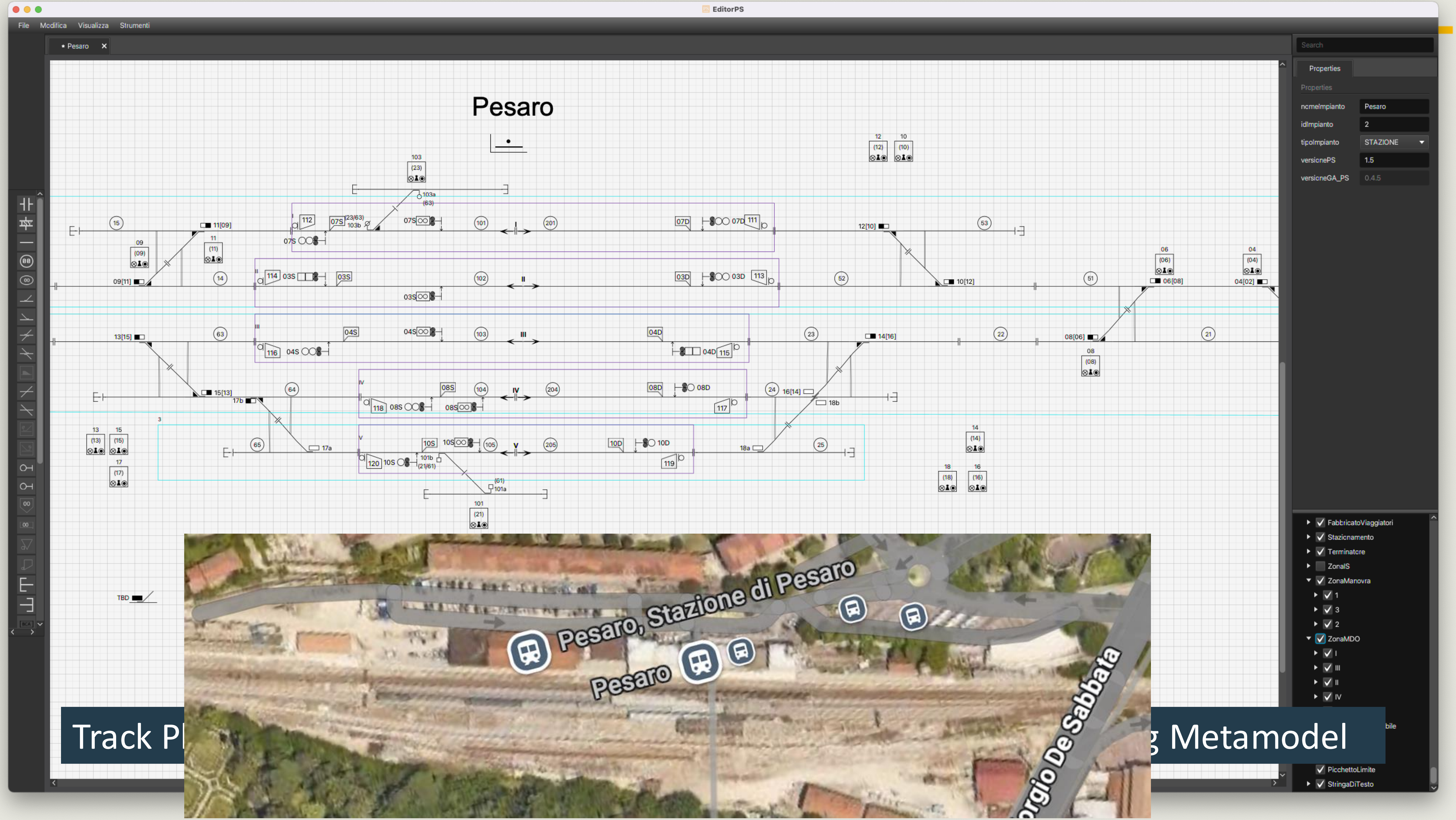
It seeks to make MDE courses accessible to bachelor students as a foundational approach to teaching abstraction jjodel.

Metamodeling for all.

MODELSWARD 2025

A SIMPLE ERD NOTATION

Positional Syntax



Properties

nomeImpianto	Pesaro
idImpianto	2
tipoImpianto	STAZIONE
versionePS	1.5
versioneGA_PS	0.4.5

- ☑ FabbricatoViaggiatori
- ☑ Stazionamento
- ☑ Terminatore
- □ ZonaIS
- ☑ ZonaManovra
  - ☑ 1
  - ☑ 3
  - ☑ 2
- ☑ ZonaMDO
  - ☑ I
  - ☑ III
  - ☑ II
  - ☑ IV
- ☑ PicchettoLimite
- ☑ StringaDiTesto

Track Pl

g Metamodel

Final Considerations

Transparency of tools is a more holistic view of software quality (maybe ISO/IEC 25010?)

- tools are integrated into a broader context of purposes and activities
- interaction with the tool must be intuitive and fluid, embodying a practical engagement with the world

The tool itself is never the focus – the task is!

Final Considerations

Solutions should be simple

Final Considerations

Solutions should be simple, not simplistic—hiding complexity to ease the user’s experience is challenging, but essential

As academics, we often underestimate that technology is not neutral—it shapes how we think about applications

Low-Code platforms capitalized on recent innovations

Jjodel Transparency

How Jjodel Implements Tool Transparency

- Live model validation without cognitive disruption
- Seamless metamodel/model co-evolution and round-tripping
- Adaptive modeling environments (e.g., incremental feature disclosure, semantic zooming, topological vs. positional notations)
- Streamlined modeling processes (e.g., projectional editing, blended/hybrid modeling)
- Documentation

Transparency vs AI

Tool transparency is good

However, seamless integration of unsupervised models, including LLMs and deep neural networks, presents risks

- Decision-making process without robust supervision is critical as such models are highly complex and difficult to interpret
- Transparency might lead to an illusion of control

Can we perform better? Probably, yes!

Can we perform better? Probably, yes!

However, building tools is little rewarding in terms of career.

Who should design modeling tools?

Who should fund tool development?

If modeling is critical, why is sustainable tool development often overlooked?

Why isn’t MDE taught at the bachelor level, despite abstraction being fundamental to computer science?

Is the barrier the paradigm itself, or the complexity of the tools?

Transparency of Tools
Beyond Usability in Modeling Tools

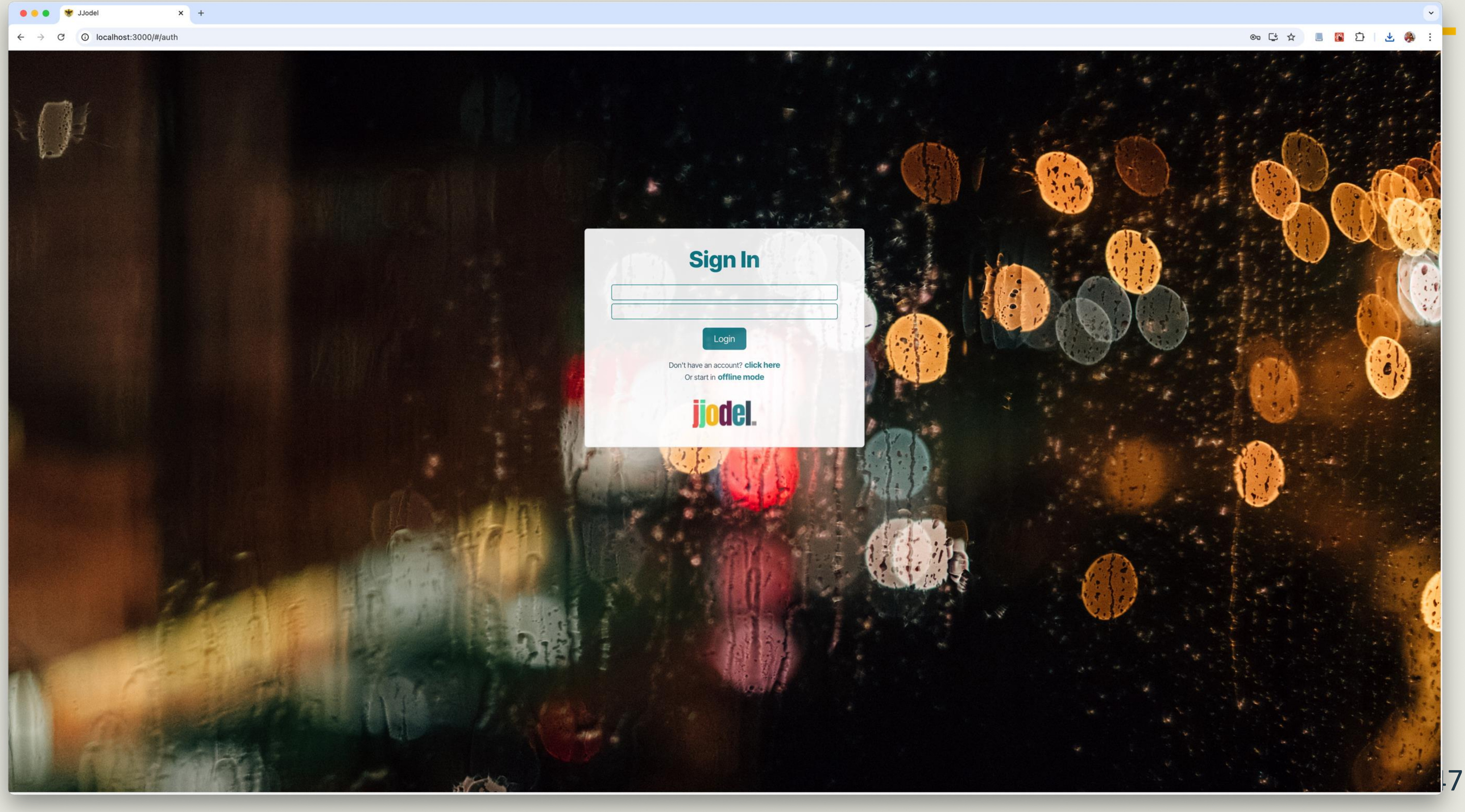
Alfonso Pierantonio
alfonso.pierantonio@univaq.it

jjodel.

SWEN
research group

UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

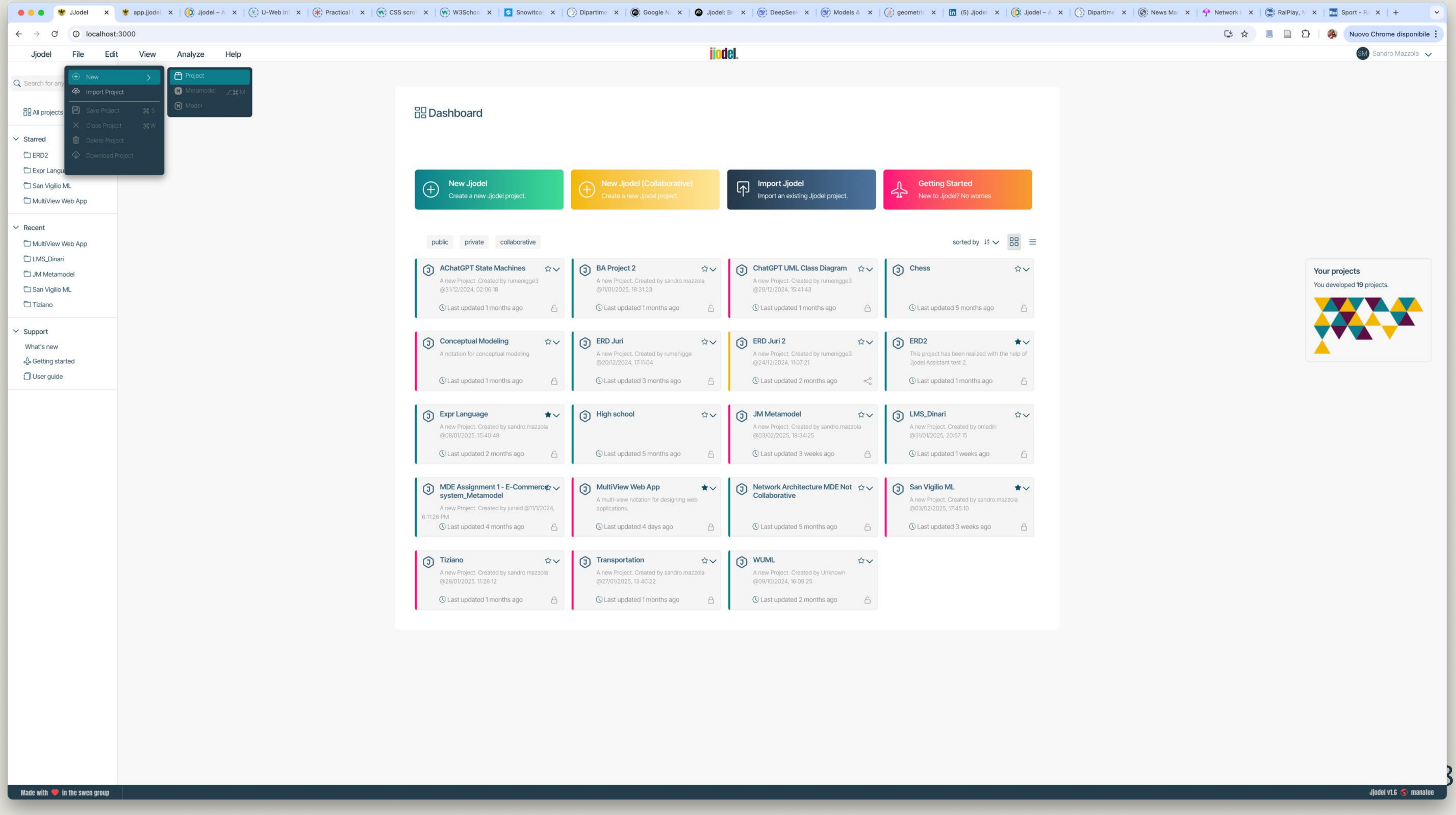
Feature	React	Angular	Vue.js
Released	2013	2010 - 2016	2014
Type	Library	Full Framework	Progressive Framework
Language	JSX	TypeScript	JavaScript
Data Binding	One-way	Two-way	Two-way
Learning Curve	Moderate	Steep	Easy-Moderate
Performance	High (Virtual DOM)	Moderate	High (Reactivity System)



Sign In

Don't have an account? **click here**

Or start in **offline mode**



Dashboard

New Jjodel
Create a new Jjodel project.

New Jjodel (Collaborative)
Create a new Jjodel project.

Import Jjodel
Import an existing Jjodel project.

Getting Started
New to Jjodel? No worries

AChatGPT State Machines
A new Project. Created by rumenigge3 @31/12/2024, 02:06:18
Last updated 1 months ago

BA Project 2
A new Project. Created by sandro.mazzola @11/01/2025, 18:31:23
Last updated 1 months ago

ChatGPT UML Class Diagram
A new Project. Created by rumenigge3 @28/12/2024, 15:41:43
Last updated 1 months ago

Chess
Last updated 5 months ago

Conceptual Modeling
A notation for conceptual modeling
Last updated 1 months ago

ERD Juri
A new Project. Created by rumenigge @20/12/2024, 17:11:04
Last updated 3 months ago

ERD Juri 2
A new Project. Created by rumenigge3 @24/12/2024, 11:07:21
Last updated 2 months ago

ERD2
This project has been realized with the help of Jjodel Assistant test 2.
Last updated 1 months ago

Expr Language
A new Project. Created by sandro.mazzola @08/01/2025, 15:40:48
Last updated 2 months ago

High school
Last updated 5 months ago

JM Metamodel
A new Project. Created by sandro.mazzola @03/02/2025, 18:34:25
Last updated 3 weeks ago

LMS_Dinari
A new Project. Created by omadin @31/01/2025, 20:57:15
Last updated 1 weeks ago

MDE Assignment 1 - E-Commerce system_Metamodel
A new Project. Created by junaid @11/1/2024, 6:11:26 PM
Last updated 4 months ago

MultiView Web App
A multi-view notation for designing web applications.
Last updated 4 days ago

Network Architecture MDE Not Collaborative
Last updated 5 months ago

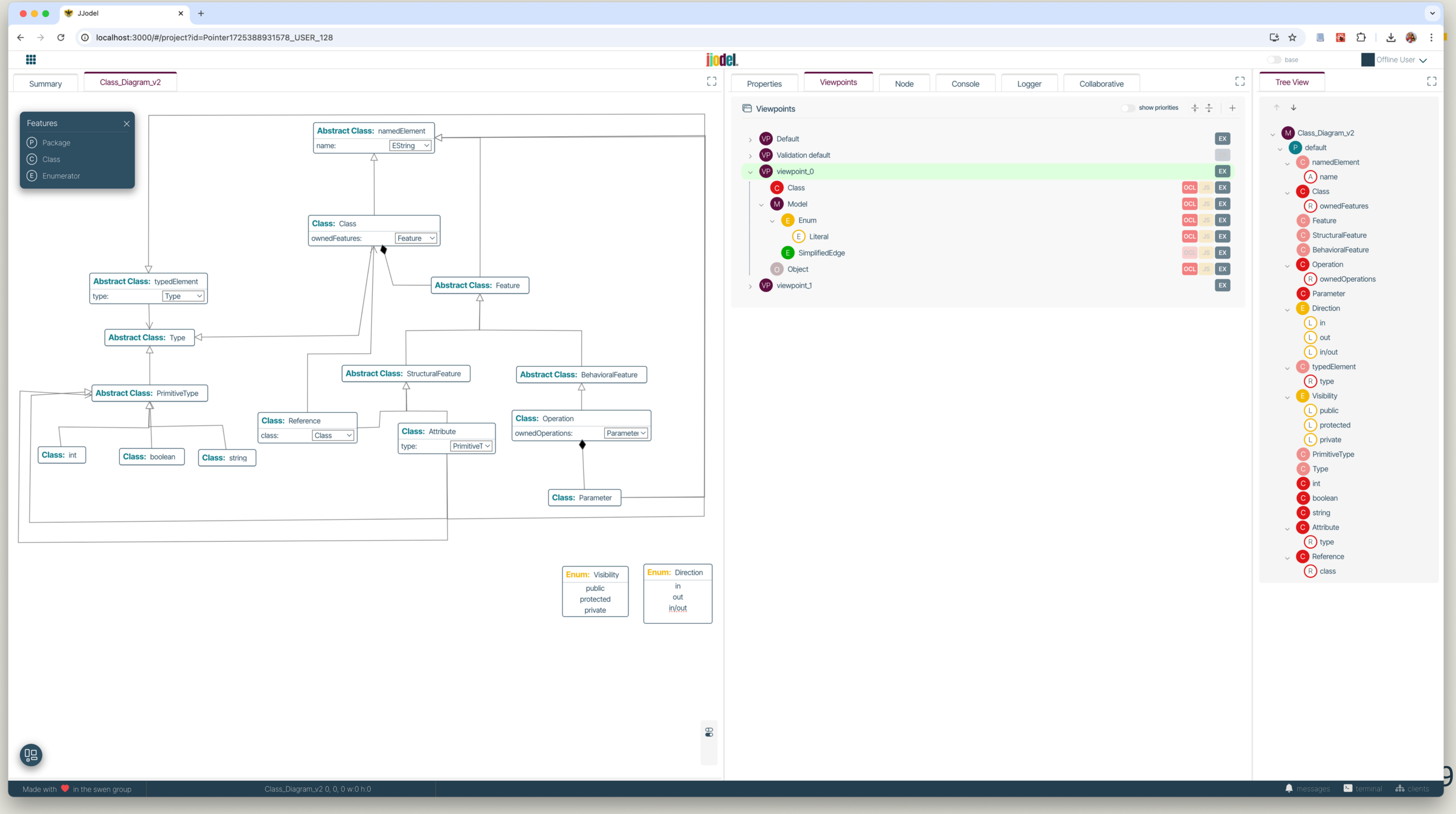
San Vigilio ML
A new Project. Created by sandro.mazzola @03/02/2025, 17:45:10
Last updated 3 weeks ago

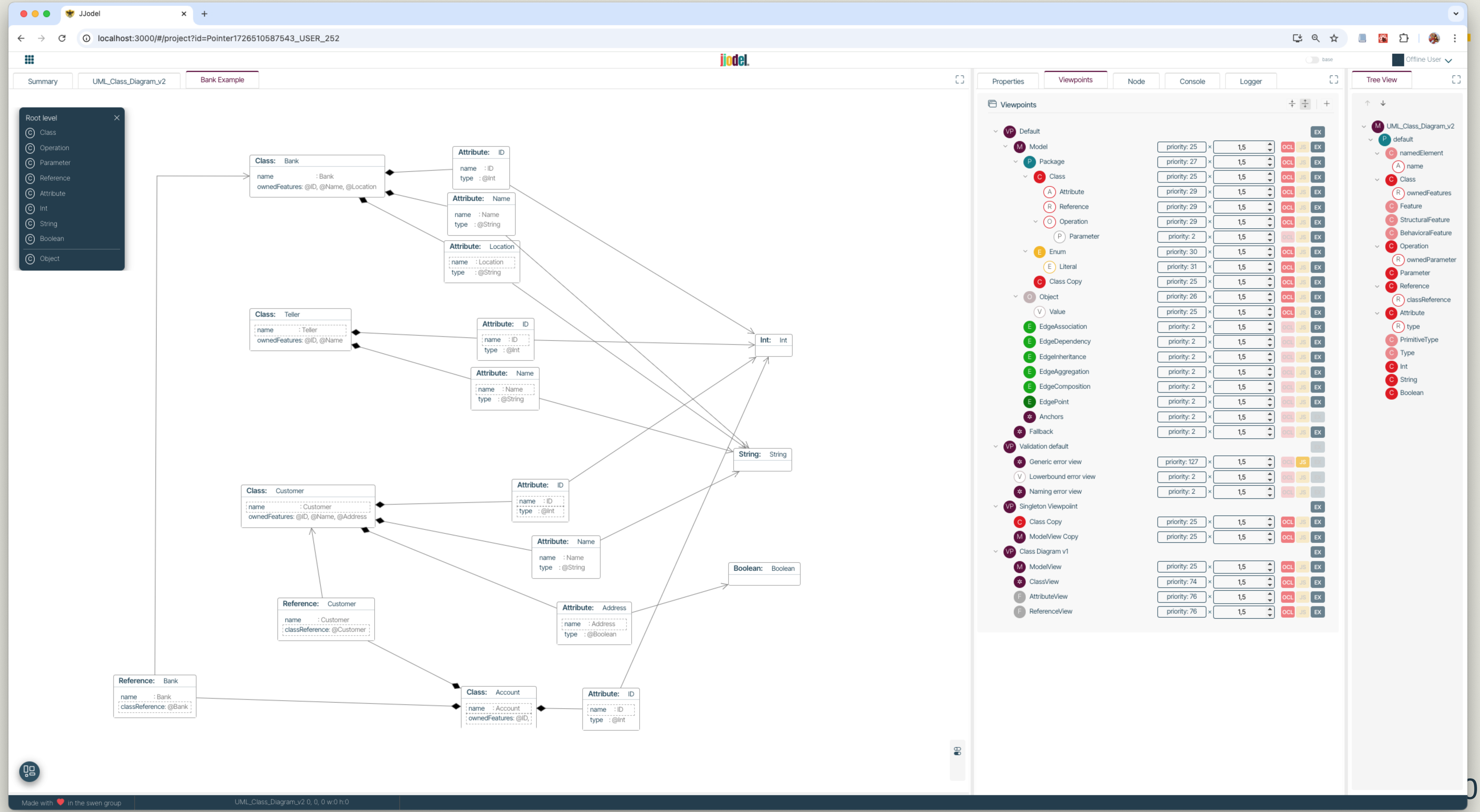
Tiziano
A new Project. Created by sandro.mazzola @28/01/2025, 11:26:12
Last updated 1 months ago

Transportation
A new Project. Created by sandro.mazzola @27/01/2025, 13:40:22
Last updated 1 months ago

WUML
A new Project. Created by Unknown @09/10/2024, 16:09:25
Last updated 2 months ago

Your projects
You developed **19** projects.


JJodel
localhost:3000/#/project?id=Pointer1725388931578_USER_128
jjodel
base
Offline User
Summary
Class_Diagram_v2
Features
Package
Class
Enumerator
Abstract Class: namedElement
name: EString
Class: Class
ownedFeatures: Feature
Abstract Class: typedElement
type: Type
Abstract Class: Feature
Abstract Class: Type
Abstract Class: StructuralFeature
Abstract Class: BehavioralFeature
Abstract Class: PrimitiveType
Class: Reference
class: Class
Class: Attribute
type: PrimitiveT
Class: Operation
ownedOperations: Parameter
Class: int
Class: boolean
Class: string
Class: Parameter
Enum: Visibility
public
protected
private
Enum: Direction
in
out
in/out
Properties
Viewpoints
Node
Console
Logger
Collaborative
Viewpoints
show priorities
Default
Validation default
viewpoint_0
Class
Model
Enum
Literal
SimplifiedEdge
Object
viewpoint_1
EX
OCL
JS
Tree View
Class_Diagram_v2
default
namedElement
name
Class
ownedFeatures
Feature
StructuralFeature
BehavioralFeature
Operation
ownedOperations
Parameter
Direction
in
out
in/out
typedElement
type
Visibility
public
protected
private
PrimitiveType
Type
int
boolean
string
Attribute
type
Reference
class
Made with ♥ in the swen group
Class_Diagram_v2 0, 0, 0 w:0 h:0
messages
terminal
clients



Class: Bank
name : Bank
ownedFeatures: @ID, @Name, @Location

Attribute: ID
name : ID
type : @Int

Attribute: Name
name : Name
type : @String

Attribute: Location
name : Location
type : @String

Class: Teller
name : Teller
ownedFeatures: @ID, @Name

Attribute: ID
name : ID
type : @Int

Attribute: Name
name : Name
type : @String

Int: Int

String: String

Class: Customer
name : Customer
ownedFeatures: @ID, @Name, @Address

Attribute: ID
name : ID
type : @Int

Attribute: Name
name : Name
type : @String

Boolean: Boolean

Reference: Customer
name : Customer
classReference: @Customer

Attribute: Address
name : Address
type : @Boolean

Reference: Bank
name : Bank
classReference: @Bank

Class: Account
name : Account
ownedFeatures: @ID,

Attribute: ID
name : ID
type : @Int

Viewpoints

```
VP Default                                         EX
   M Model                  priority: 25  × 1,5  OCL JS EX
      P Package             priority: 27  × 1,5  OCL JS EX
         C Class            priority: 25  × 1,5  OCL JS EX
            A Attribute     priority: 29  × 1,5  OCL JS EX
            R Reference     priority: 29  × 1,5  OCL JS EX
            O Operation     priority: 29  × 1,5  OCL JS EX
               P Parameter  priority: 2   × 1,5  OCL JS EX
         E Enum             priority: 30  × 1,5  OCL JS EX
            E Literal       priority: 31  × 1,5  OCL JS EX
         C Class Copy       priority: 25  × 1,5  OCL JS EX
      O Object              priority: 26  × 1,5  OCL JS EX
         V Value            priority: 25  × 1,5  OCL JS EX
      E EdgeAssociation     priority: 2   × 1,5  OCL JS EX
      E EdgeDependency      priority: 2   × 1,5  OCL JS EX
      E EdgeInheritance     priority: 2   × 1,5  OCL JS EX
      E EdgeAggregation     priority: 2   × 1,5  OCL JS EX
      E EdgeComposition     priority: 2   × 1,5  OCL JS EX
      E EdgePoint           priority: 2   × 1,5  OCL JS EX
      ✲ Anchors             priority: 2   × 1,5  OCL JS EX
   ✲ Fallback               priority: 2   × 1,5  OCL JS EX
VP Validation default                              EX
   ✲ Generic error view     priority: 127 × 1,5  OCL JS EX
   V Lowerbound error view  priority: 2   × 1,5  OCL JS EX
   ✲ Naming error view      priority: 2   × 1,5  OCL JS EX
VP Singleton Viewpoiint                            EX
   C Class Copy             priority: 25  × 1,5  OCL JS EX
   M ModelView Copy         priority: 25  × 1,5  OCL JS EX
VP Class Diagram v1                                EX
   M ModelView              priority: 25  × 1,5  OCL JS EX
   ✲ ClassView              priority: 74  × 1,5  OCL JS EX
   F AttributeView          priority: 76  × 1,5  OCL JS EX
   F ReferenceView          priority: 76  × 1,5  OCL JS EX
```