Augmenting graphical modeling workbenches with semantic-aware interactive features

Théo Giraudet*
*Obeo, Univ Rennes, Inria, CNRS, IRISA
theo.giraudet@irisa.fr

Supervisors: Benoît Combemale & Arnaud Blouin

AW06

Introduction

- Graphical modeling workbenches provide a wide range of services to help domain experts to use DSMLs
  - Editors, validators, formatters, …
- Graphical modeling workbenches and DSMLs are made using language workbenches

- However, several studies have identified the usability of modeling workbenches as a key issue [1-3]
  - Modeling tasks are slow, costly to handle, …

[1] Omar Badreddin, Rahad Khandoker, Andrew Forward, Omar Masmali, and Timothy C. Lethbridge. 2018. A Decade of Software Design and Modeling: A Survey to Uncover Trends of the Practice. In Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (Copenhagen, Denmark) (MODELS '18).Association for Computing Machinery, New York, NY, USA, 245–255.
[2] John Hutchinson, Jon Whittle, and Mark Rouncefield. 2014. Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure. Science of Computer Programming 89 (2014),144–161.
[3] Charlotte Verbruggen and Monique Snoeck. 2023. Practitioners' experiences with model-driven engineering: a meta-review. Software and Systems Modeling 22, 1 (Feb 2023), 111–129

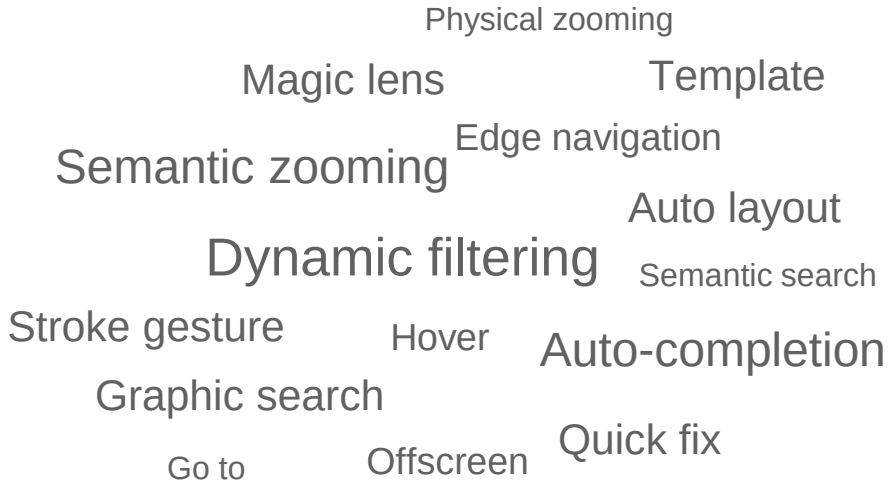
OBEO

Université de Rennes

Semantic-aware interactive features

Semantic-aware interactive features

- Numerous articles in the HCI and SLE communities proposes various interactive features to improve the usability of modeling workbenches

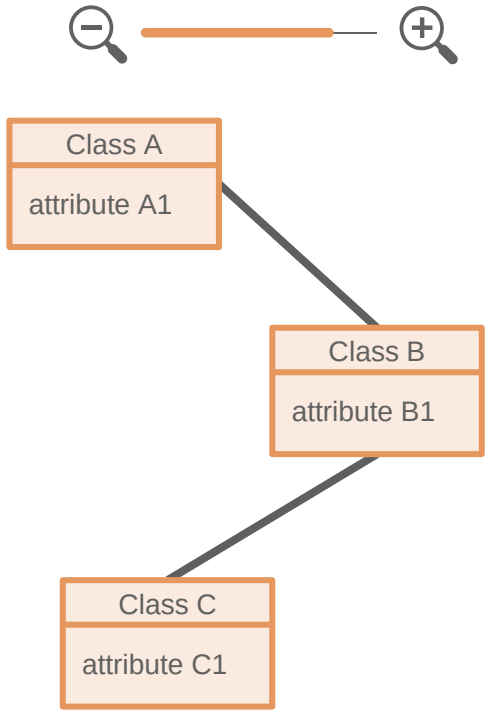


Semantic-aware interactive features

- Numerous articles in the HCI and SLE communities proposes various interactive features to improve the usability of modeling workbenches
- Some of them are domain-specific
  - So… We have to define them for each DSML
- How can we facilitate the development of such semantic-aware interactive features?

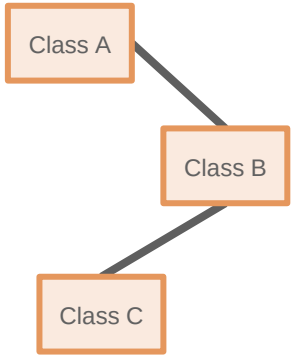
Semantic-aware interactive features

- We focus on three main semantic-aware interactive features:
  - Semantic zooming



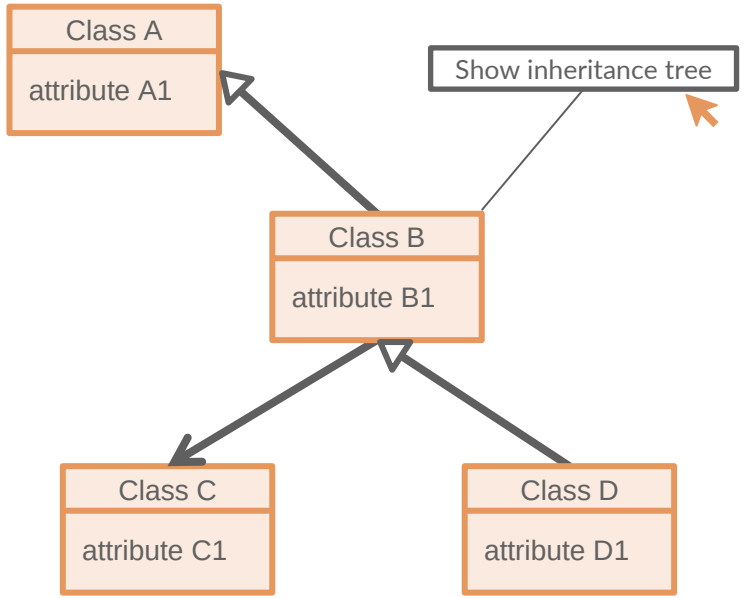
Semantic-aware interactive features

- We focus on three main semantic-aware interactive features:
  - Semantic zooming



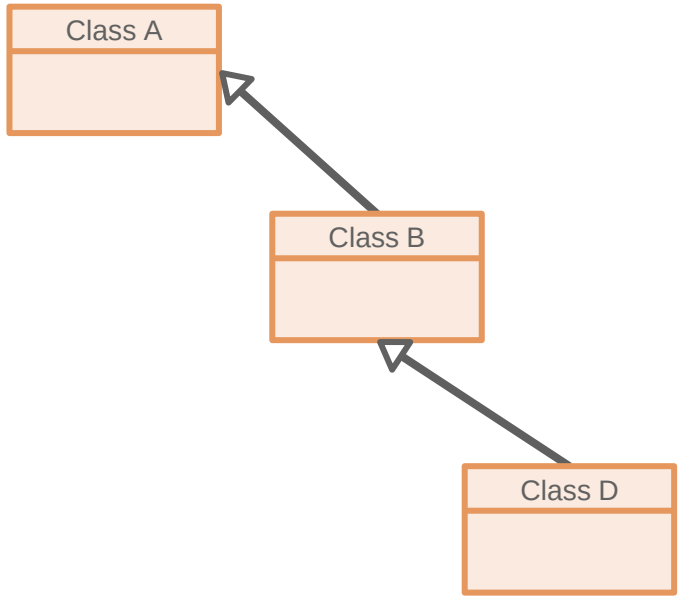
Semantic-aware interactive features

- We focus on three main semantic-aware interactive features:
  - Semantic zooming
  - Dynamic filtering



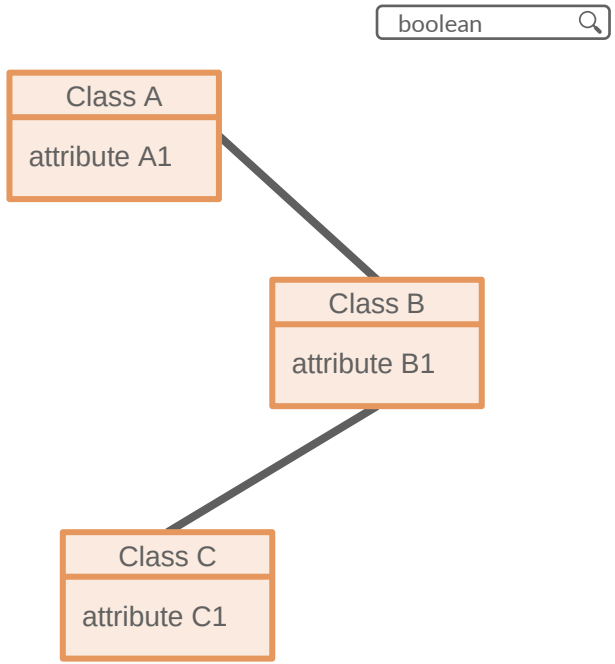
Semantic-aware interactive features

- We focus on three main semantic-aware interactive features:
  - Semantic zooming
  - Dynamic filtering



Semantic-aware interactive features

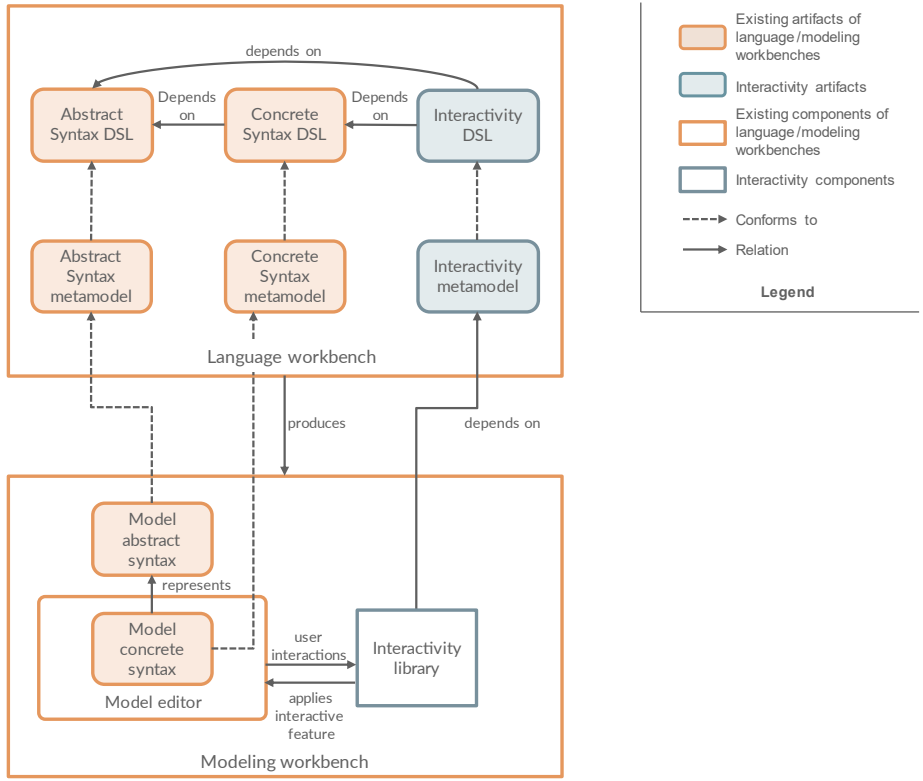
- We focus on three main semantic-aware interactive features:
  - Semantic zooming
  - Dynamic filtering
  - Semantic search



Our approach

Our approach

- We propose a new meta-DSL, “Interactivity DSL”, to describe semantic-aware interactive features
- This meta-DSL is aware of the concrete and abstract syntaxes of the developed DSML
- It is integrated inside language workbenches



Our approach

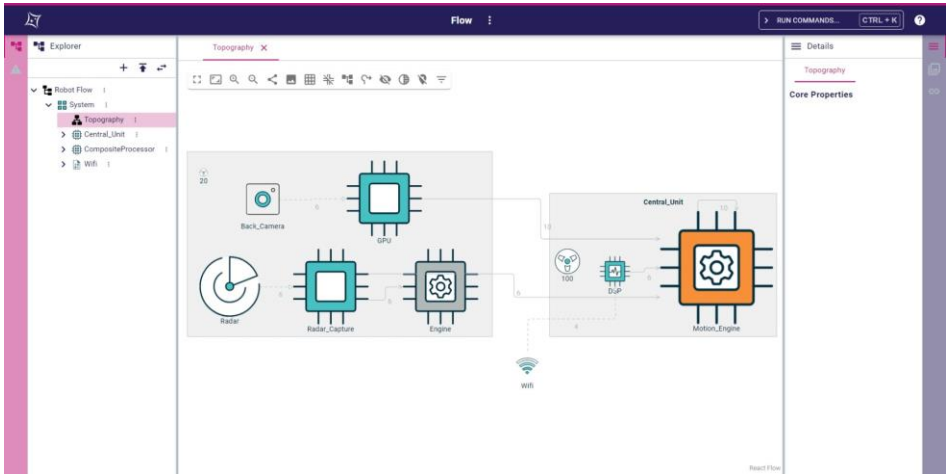
- We propose a new meta-DSL, “Interactivity DSL”, to describe semantic-aware interactive features
- This meta-DSL is aware of the concrete and abstract syntaxes of the developed DSML
- It is integrated inside language workbenches

```
import abstract-syntax ‘uml-as.ecore' as umlas
import concrete-syntax ‘uml-cs.ecore' as umlcs
search: [umlas.packages.classes.name, umlas.packages.name, ‘*’]
dynamic-filter:
    name: inheritance
    focus: umlas.packages.classes
    radius: [1..*]
    filter: show-inheritance
semantic-zoom:
    [0%-75%[:
        filter: without-attributes
    [150%-200%[:
        filter: without-packages
filters:
    show-inheritance:
        show: umlas.packages.classes
        show: umlas.package.class.superclasses
    without-attributes:
        hide: umlas.packages.classes.attributes
        setstyle umlas.packages.classes.name:
            font-size 200%
    without-package:
        hide: umlas.packages
```

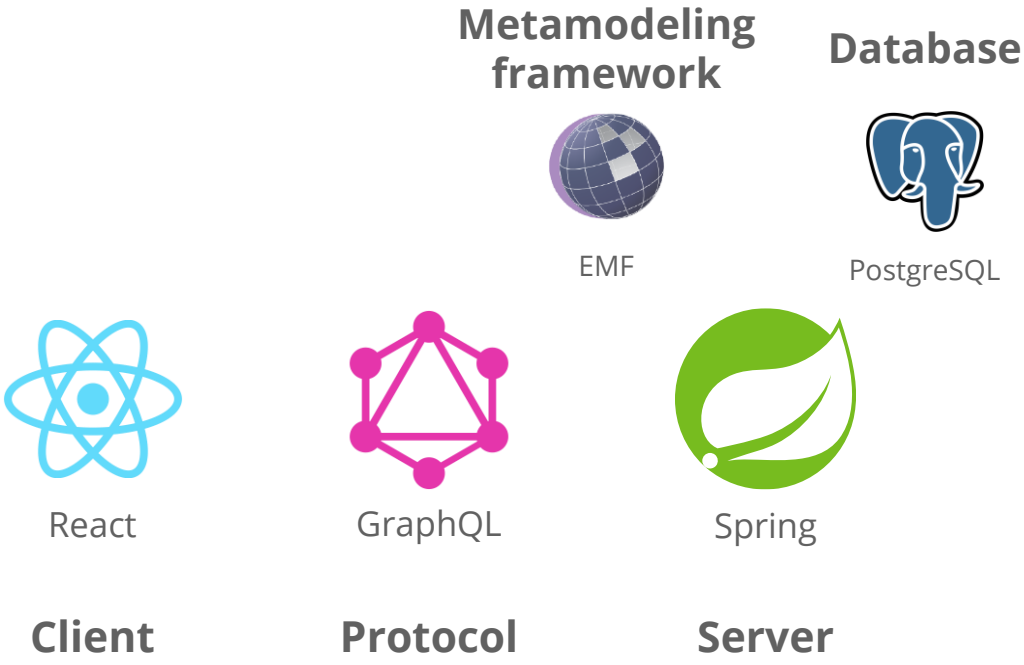
Implementation

Implementation

- We implemented our Interactivity DSL inside the Sirius Web graphical language workbench
- Open-source language workbench, hosted by the Eclipse Foundation
- Cloud-native architecture



Implementation



Most computations are performed on the server side

Implementation

Demonstration

Conclusion

Conclusion

- We proposed a new meta-DSL to describe semantic-aware interactive features for graphical DSML
- It currently supports three semantic-aware interactive features: semantic zooming, dynamic filtering, and semantic search
- We implemented it inside the Sirius Web language workbench

Future work

- Generify the approach to facilitate the addition of new semantic-aware interactive features without modifying the meta-DSL itself
- Implement our approach in GLSP

Augmenting graphical modeling workbenches with semantic-aware interactive features

Speaker: Théo Giraudet
Contact: theo.giraudet@irisa.fr

- We proposed a new meta-DSL to describe semantic-aware interactive features for graphical DSML
- It currently supports three semantic-aware interactive features: semantic zooming, dynamic filtering, and semantic search
- We implemented it inside the Sirius Web language workbench

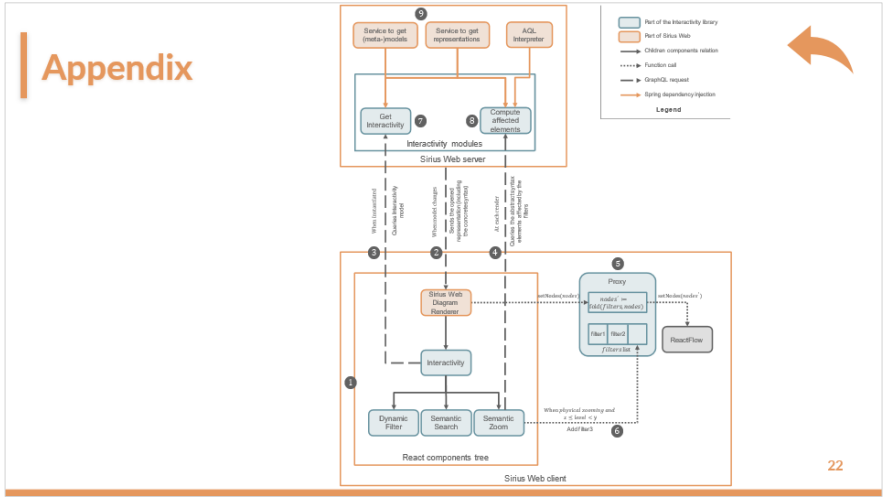
Question to the audience

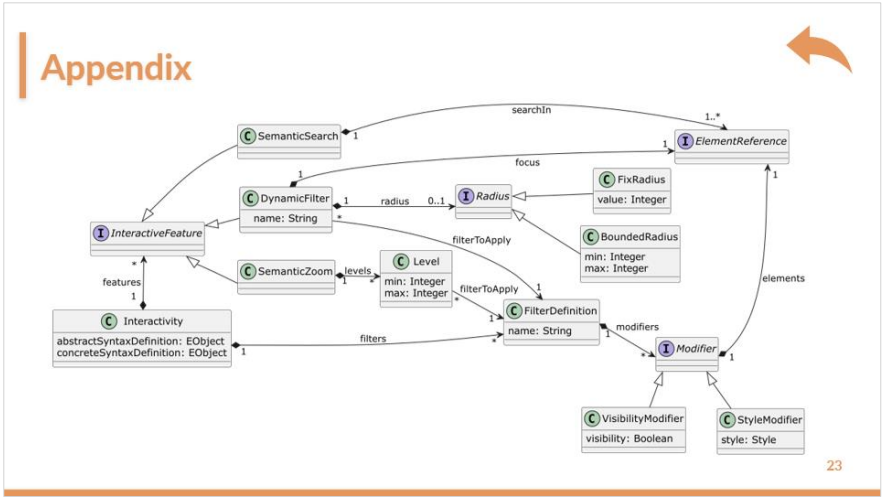
- Are you aware of other semantic-aware interactive features (*e.g.*, to edit models)?

AW06

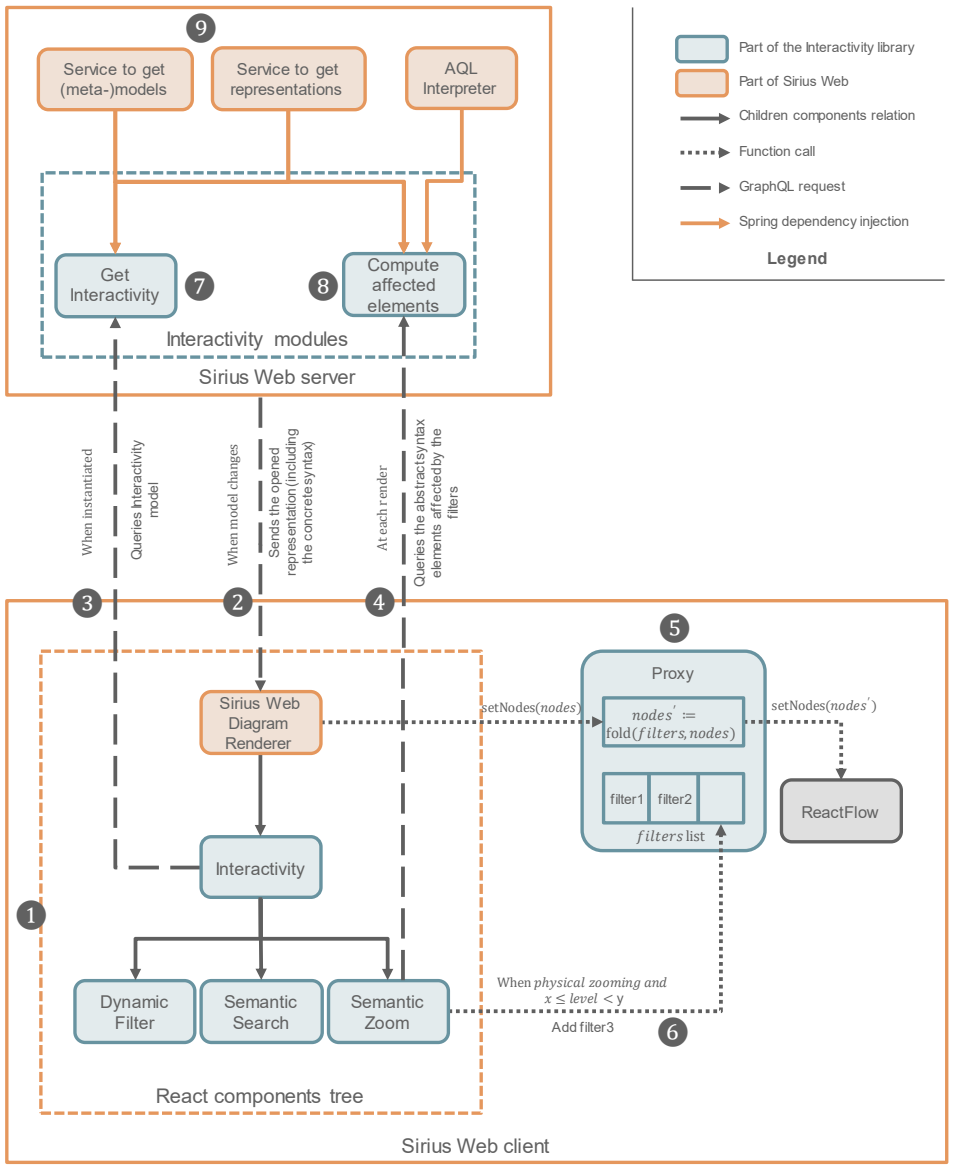
ProxyHands

Appendix

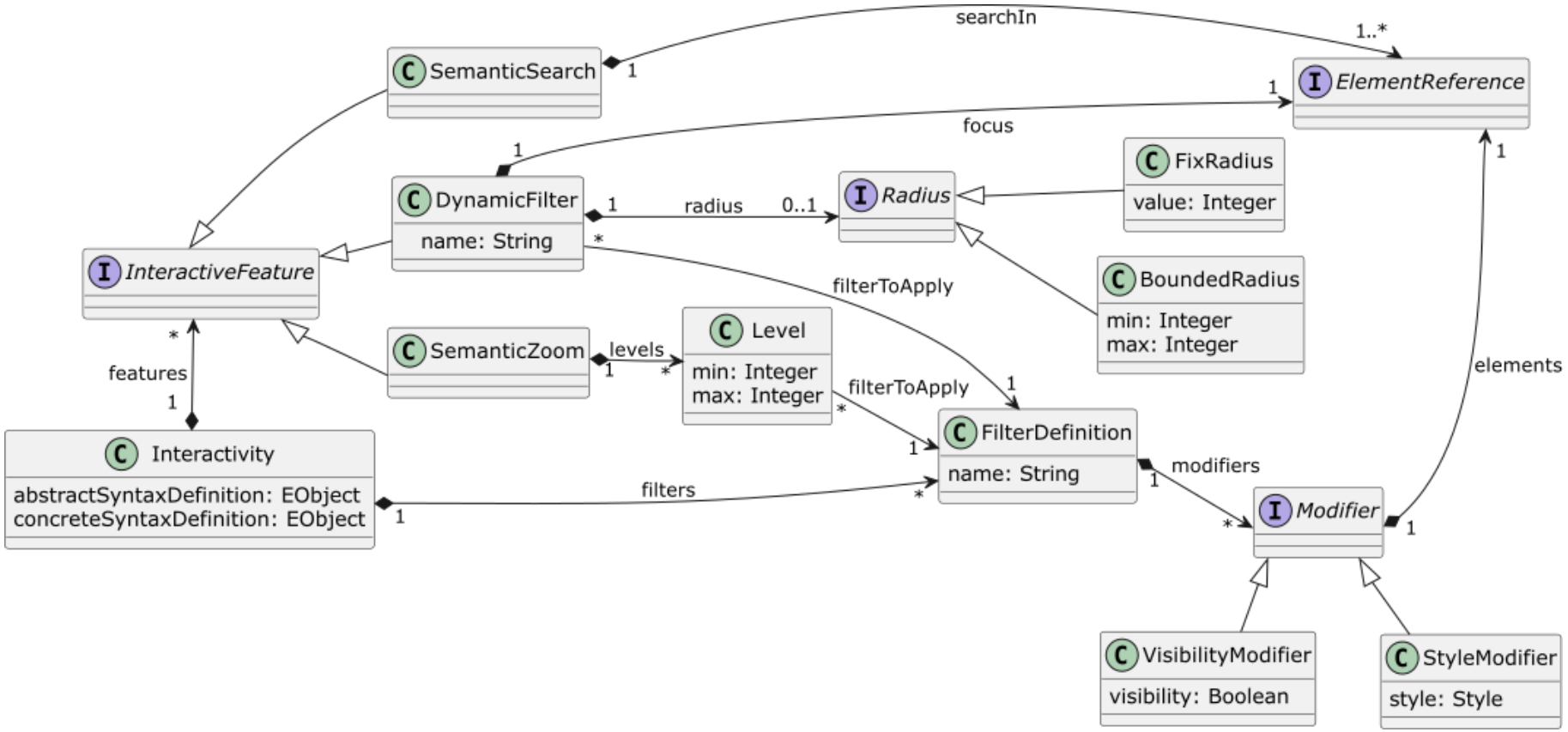

Appendix
Service to get (meta-)models
Service to get representations
AQL Interpreter
Get Interactivity
Compute affected elements
Interactivity modules
Sirius Web server
Part of the Interactivity library
Part of Sirius Web
Children components relation
Function call
GraphQL request
Spring dependency injection
Legend
Sirius Web Diagram Renderer
Proxy
ReactFlow
Interactivity
Dynamic Filter
Semantic Search
Semantic Zoom
React components tree
Sirius Web client
22


Appendix
SemanticSearch
searchIn
ElementReference
focus
DynamicFilter
name: String
radius
Radius
FixRadius
value: Integer
InteractiveFeature
filterToApply
BoundedRadius
min: Integer
max: Integer
SemanticZoom
levels
Level
min: Integer
max: Integer
features
elements
Interactivity
abstractSyntaxDefinition: EObject
concreteSyntaxDefinition: EObject
filters
FilterDefinition
name: String
modifiers
Modifier
VisibilityModifier
visibility: Boolean
StyleModifier
style: Style
23

Appendix


9
Service to get (meta-)models
Service to get representations
AQL Interpreter
Get Interactivity
7
8
Compute affected elements
Interactivity modules
Sirius Web server
When instantiated
Queries Interactivity model
When model changes
Sends the opened representation (including the concrete syntax)
At each render
Queries the abstract syntax elements affected by the filters
3
2
4
5
Proxy
Sirius Web Diagram Renderer
setNodes(nodes)
nodes' := fold(filters, nodes)
setNodes(nodes')
filter1
filter2
filters list
ReactFlow
Interactivity
1
Dynamic Filter
Semantic Search
Semantic Zoom
When physical zooming and x ≤ level < y
Add filter3
6
React components tree
Sirius Web client
Part of the Interactivity library
Part of Sirius Web
Children components relation
Function call
GraphQL request
Spring dependency injection
Legend

Appendix


searchIn
1..*
SemanticSearch
1
ElementReference
1
focus
1
DynamicFilter
name: String
1
radius
0..1
Radius
FixRadius
value: Integer
BoundedRadius
min: Integer
max: Integer
InteractiveFeature
*
filterToApply
1
SemanticZoom
levels
1
*
Level
min: Integer
max: Integer
filterToApply
*
1
FilterDefinition
name: String
1
modifiers
*
Modifier
1
elements
1
features
*
1
Interactivity
abstractSyntaxDefinition: EObject
concreteSyntaxDefinition: EObject
1
filters
*
VisibilityModifier
visibility: Boolean
StyleModifier
style: Style