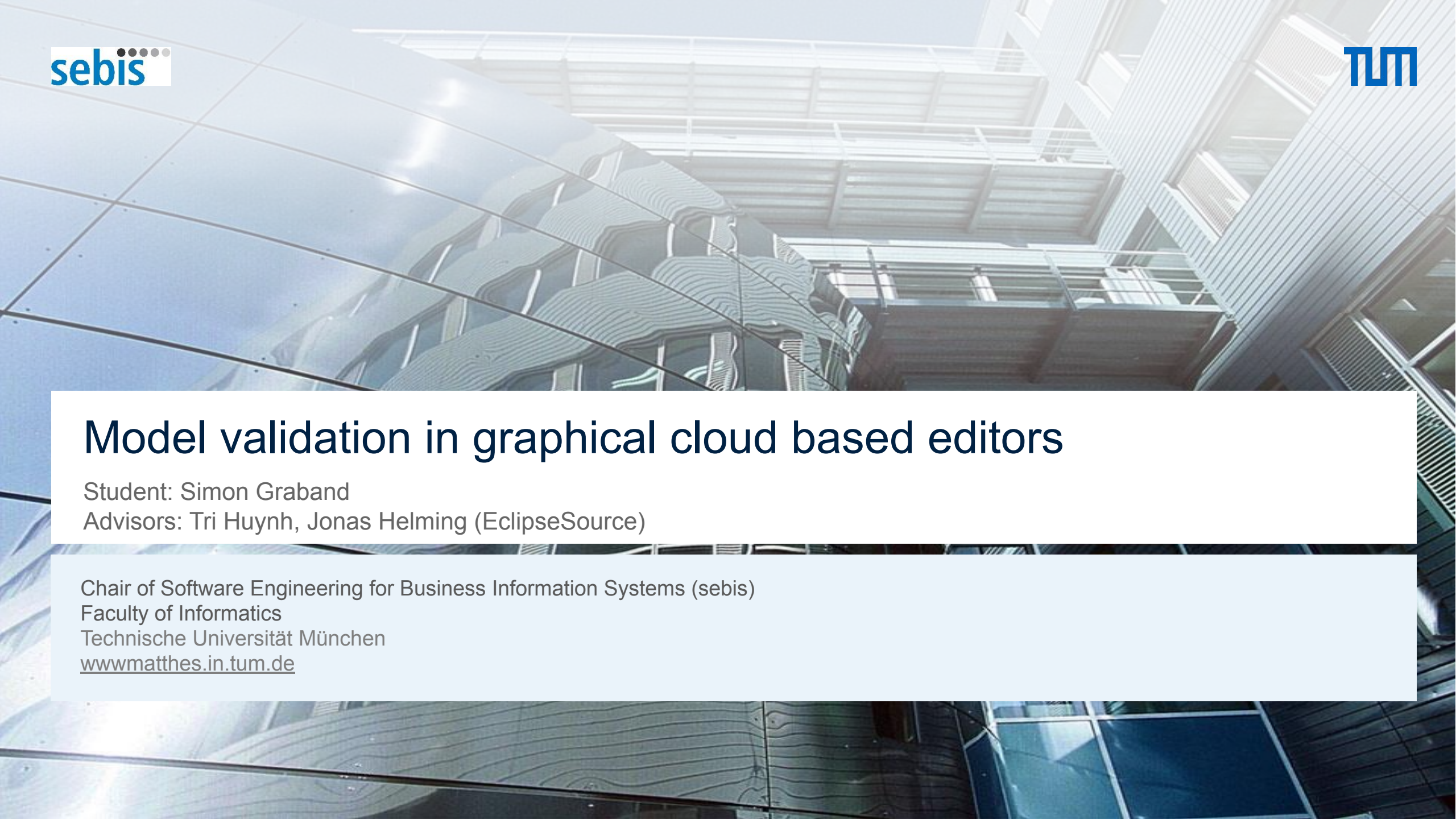


Model validation in graphical cloud based editors

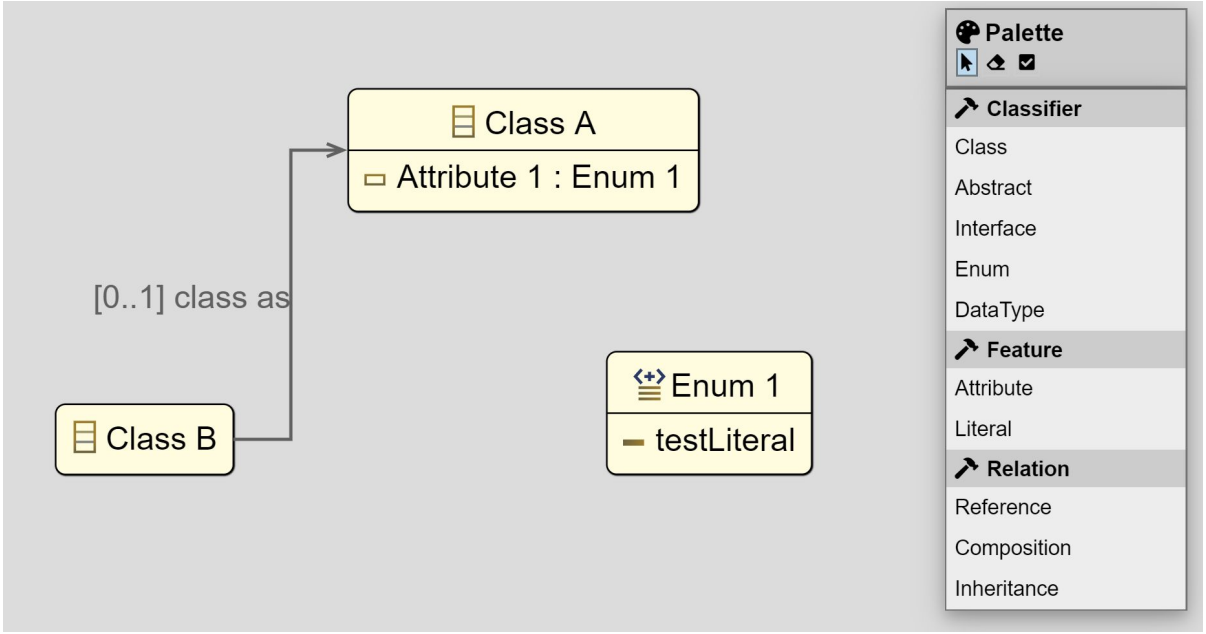
Student: Simon Graband
Advisors: Tri Huynh, Jonas Helming (EclipseSource)

Chair of Software Engineering for Business Information Systems (sebis)
Faculty of Informatics
Technische Universität München
wwwmatthes.in.tum.de

About Me

Simon Graband

- M.Sc. Information Systems (4th semester)
- Master Lab Course:
  - Project: A graphical modelling editor in the web
  - Gained insights into cloud-based modelling tools
- Working Student at EclipseSource



© sebis

Outline

Motivation

Research Questions

Process Description

Expected Outcome

First Results

Timetable

© sebis

Motivation

for cloud-based tools

Tendencies to cloud-based IDEs

Advantages:

- Zero Installation time
- Portability
- Scalability

Especially beneficial when:

- Involvement of several (external) parties
- Usage time is short (e.g. “review”)

for model validation

Essential benefit of modelling is reduced costs

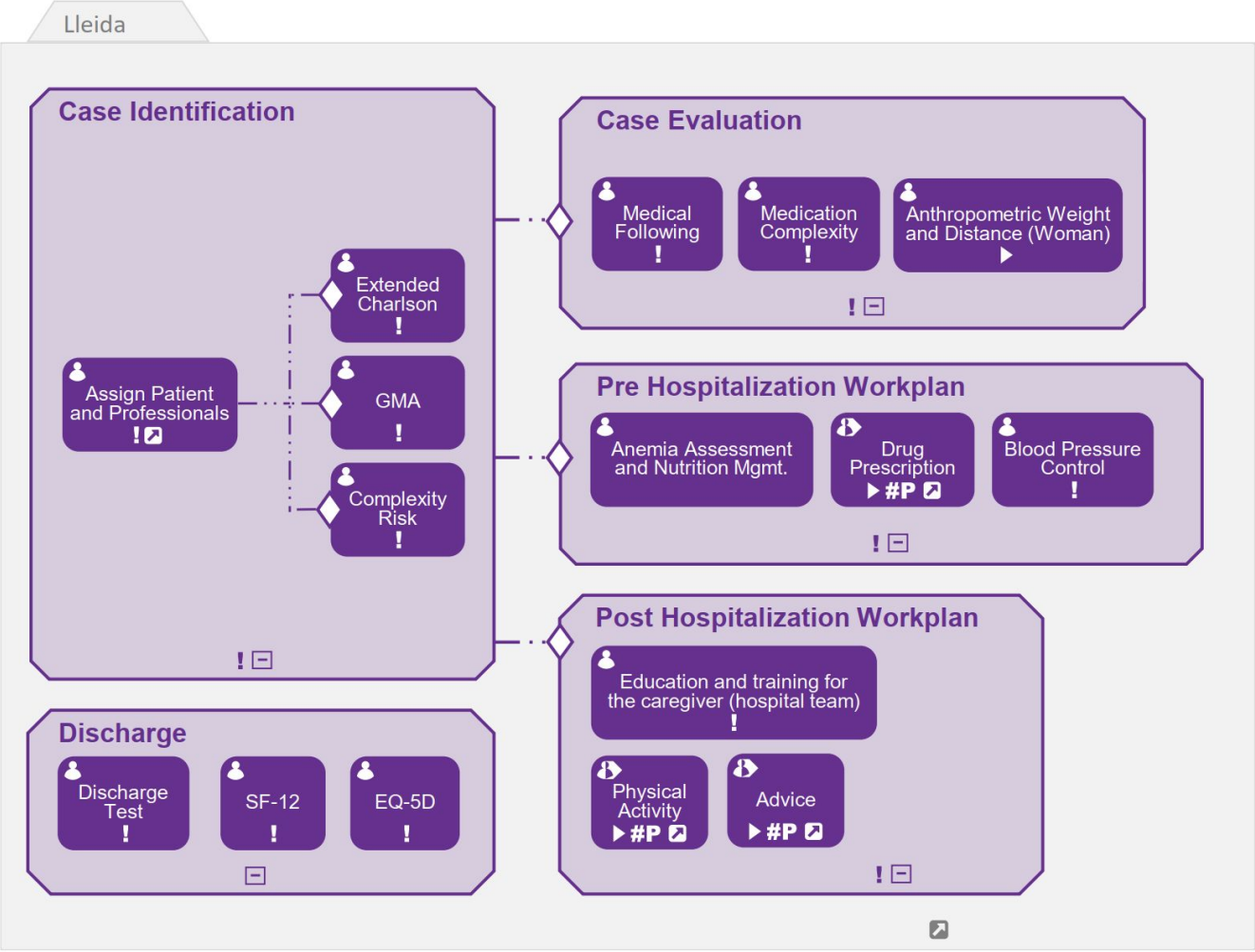
Advantages:

- prevent errors
- less errors
- faster error fixing
- more consistent models

➡ A solution to enable model validation in the cloud

© sebis

Example



Simplified Treatment Plan for chronic diseases

Stage - Treatment Phases

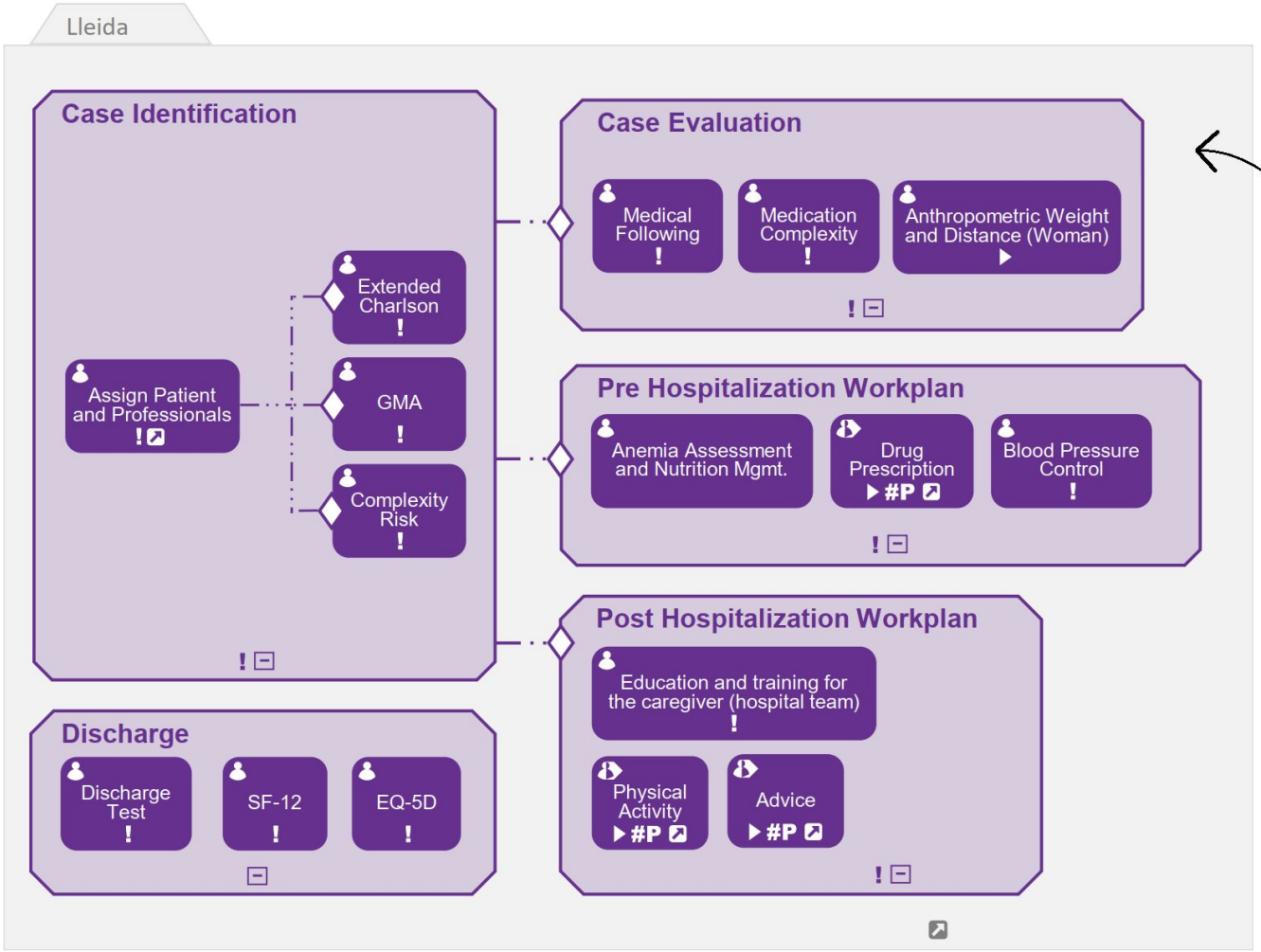
Task -Treatment activities

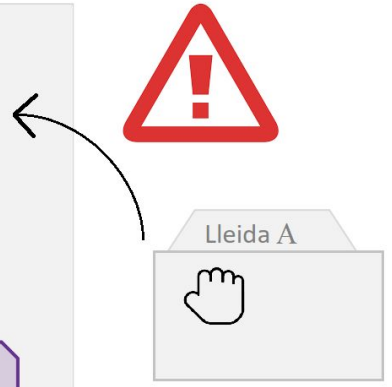
Sentry - Preconditions

HTTP Hook - HTTP request trigger

© sebis

Example



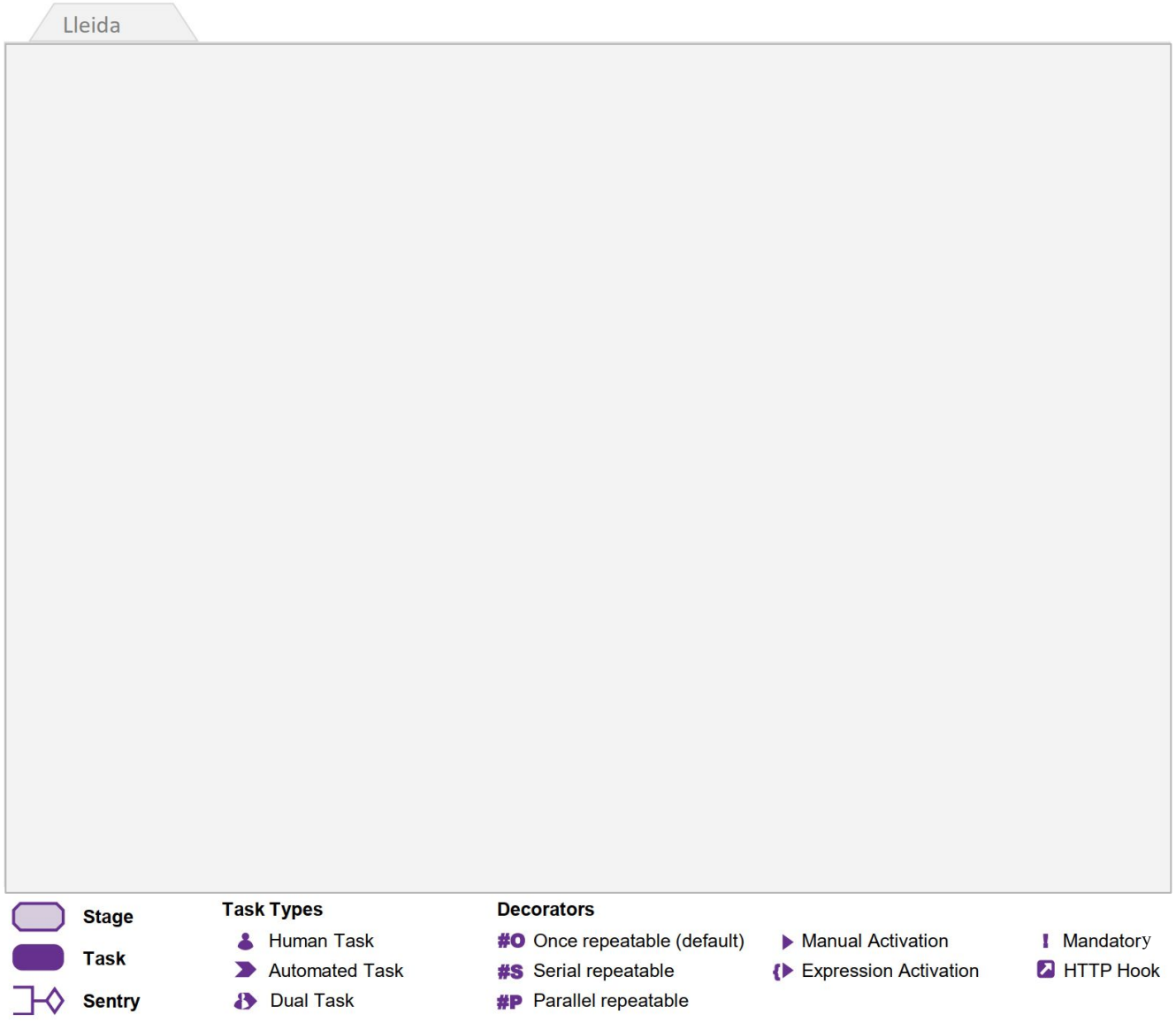


Invalid Nesting

A case cannot be nested within another case

© sebis

Example



Empty Case

A case cannot be empty

© sebis

Outline

Motivation

Research Questions

Process Description

Expected Outcome

First Results

Timetable

© sebis

Research Questions

1. What are the existing solutions for constraint validation?

2. How can those solutions be applied to a cloud-based tool?

    2.1. How to execute and visualize model validation in a client server architecture?

3. What user benefits can be measured with a case study?

© sebis

Outline

Motivation

Research Questions

Process Description

Expected Outcome

First Results

Timetable

© sebis

Process Description

Research

- Literature research on the topic
- What model validation solutions exist?
- How can validation constraints be categorized?
- What standard approaches can be derived?
- What are the problems with current validation techniques?

© sebis

Process Description

CMMN Editor (Proof of Concept)

- Based on GLSP&Theia
- PoC for the offered solution
- Stack:
  - Typescript (Client), Java (Server)

Validation (Framework)

- Methods for
  - integrating/implementing constraints
  - splitting validation methods between Server & Client
  - error prevention
  - error detection
  - improving the usability (e.g. “quick-fixes”)

© sebis

Process Description

Case Study

- Conduct tests/interviews about the validation
- Test how the framework improves usage time & usability
- What are the benefits for the user?
- Which limitations are part of the offered solution?

© sebis

Outline

Motivation

Research Questions

Process Description

Expected Outcome

First Results

Timetable

© sebis

Expected Outcome

Validation

- Can detect/prevent language-specific and user-defined constraints
- Usable error prevention/handling of the framework
- PoC for validation in a cloud-based CMMN Editor

Case Study

- Assessment of usability & functionality of the framework & editor
- Limitations of the proposed solution
- Feedback for future improvements

Thesis

- Motivation for the topic
- Research on existing solutions
- Categorization of constraints
- Solution for implementing model validation into a cloud-based tool

© sebis

Outline

Motivation

Research Questions

Process Description

Expected Outcome

First Results

Timetable

© sebis

First Results - Dimensions of constraints

Time of validation [1]

- When to validate?
- Live vs. batch
- Pre-enter, pre-confirm, pre-save, after-save

Scope [2]

- What is needed to validate?
- Local to an attribute
- Local to a class
- Global (e.g. Circle Detection)

Feedback [3]

- How to enforce/detect the constraint?
- Blocking the input
- Showing the error
- *Severity (errors vs. warnings)*

Location

- Where to validate?
- Server-side handling
- Client-side handling

© sebis

First Results - Dimensions of constraints

Time of validation [1]

- When to validate?
- Live vs. batch
- Pre-enter, pre-confirm, pre-save, after-save

Scope [2]

- What is needed to validate?
- Local to an attribute
- Local to a class
- Global (e.g. Circle Detection)

Feedback [3]

- How to enforce/detect the constraint?
- Blocking the input
- Showing the error
- *Severity (errors vs. warnings)*

Location

- Where to validate?
- Server-side handling
- Client-side handling

© sebis

First Results - Dimensions of constraints

Time of validation [1]

- When to validate?
- Live vs. batch
- Pre-enter, pre-confirm, pre-save, after-save

Scope [2]

- What is needed to validate?
- Local to an attribute
- Local to a class
- Global (e.g. Circle Detection)

Feedback [3]

- How to enforce/detect the constraint?
- Blocking the input
- Showing the error
- *Severity (errors vs. warnings)*

Location

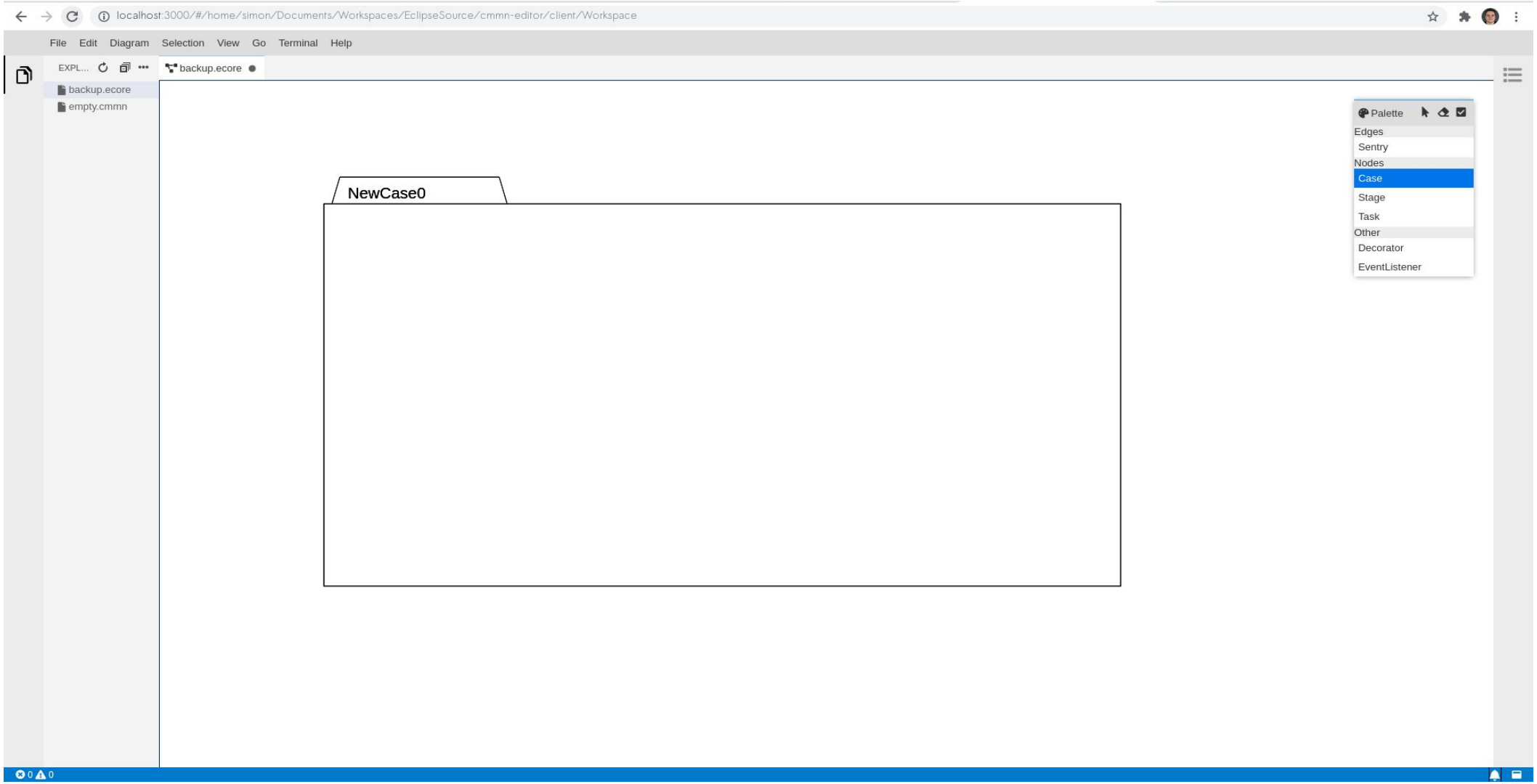
- Where to validate?
- Server-side handling
- Client-side handling

© sebis

(Demo) First Prototype of the CMMN Editor

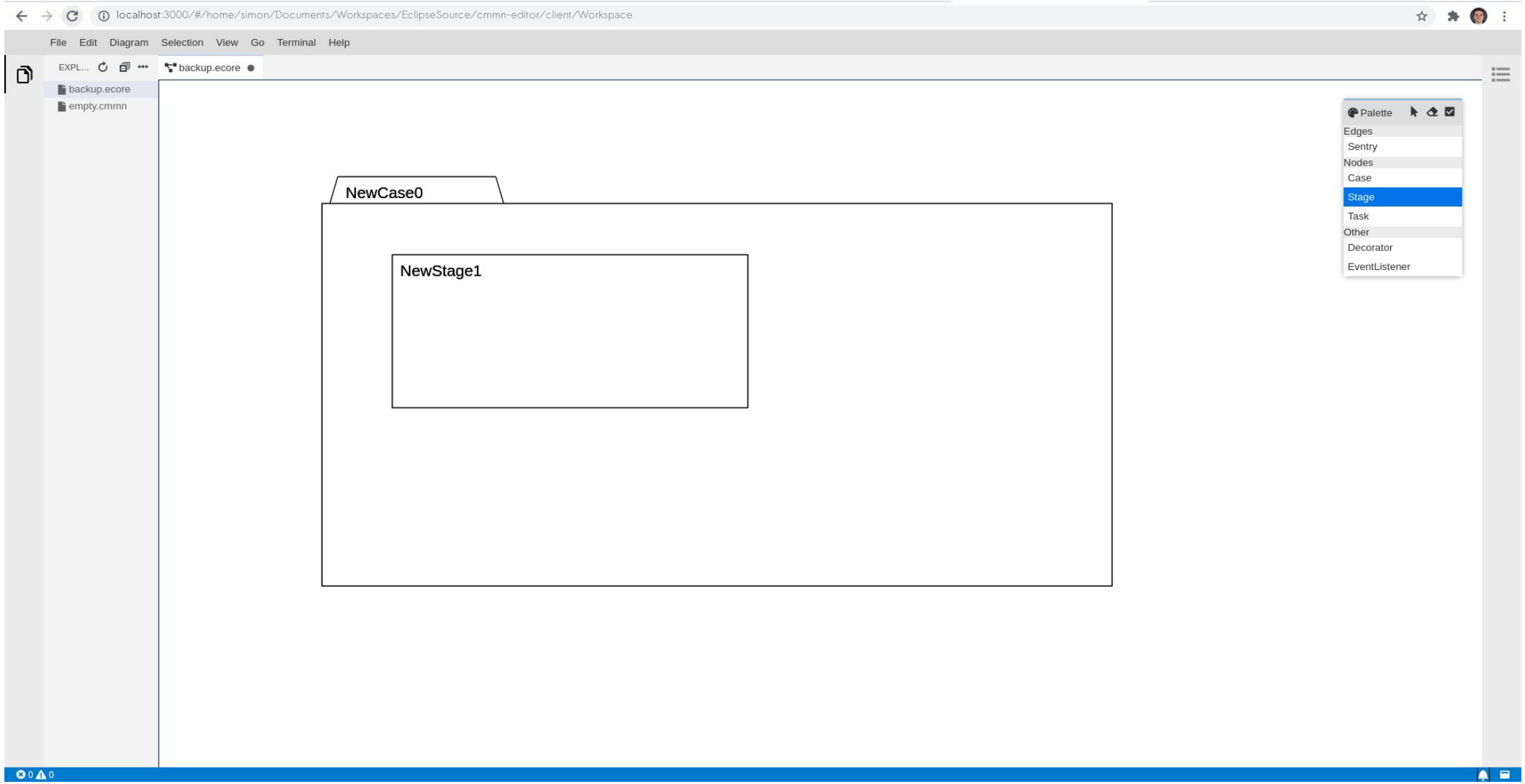
© sebis

Creation of Elements



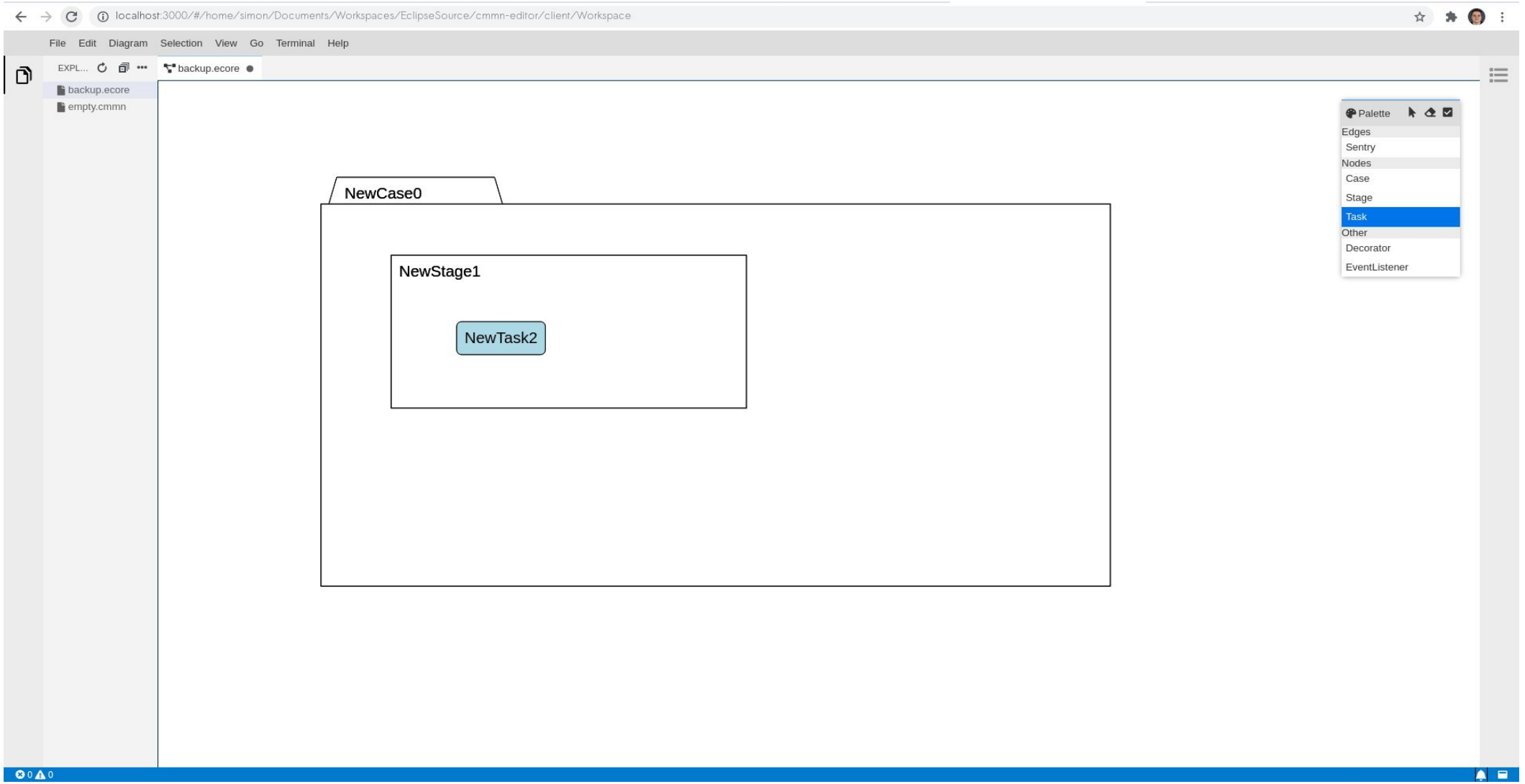
© sebis

Creation of Elements



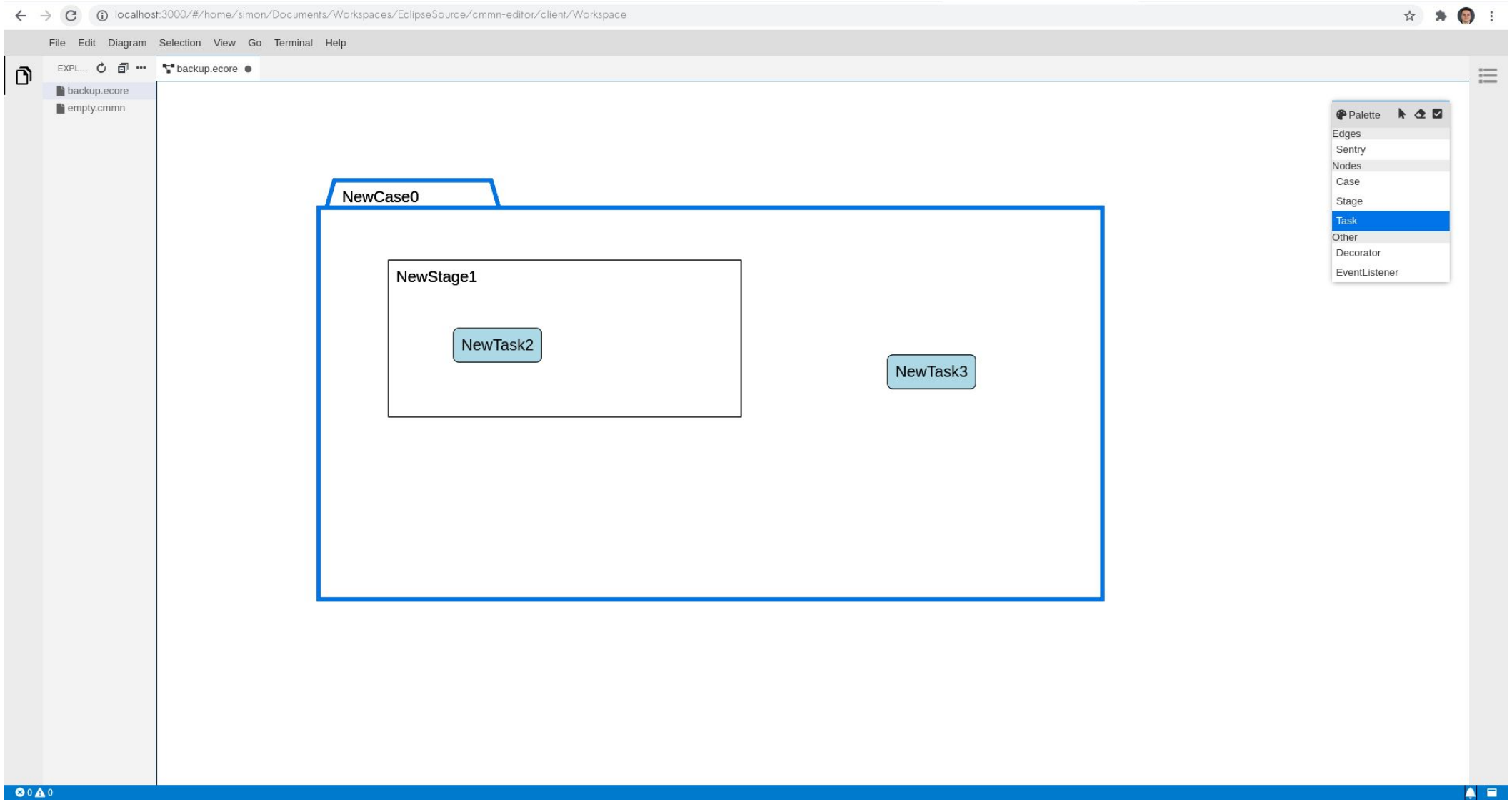
© sebis

Creation of Elements



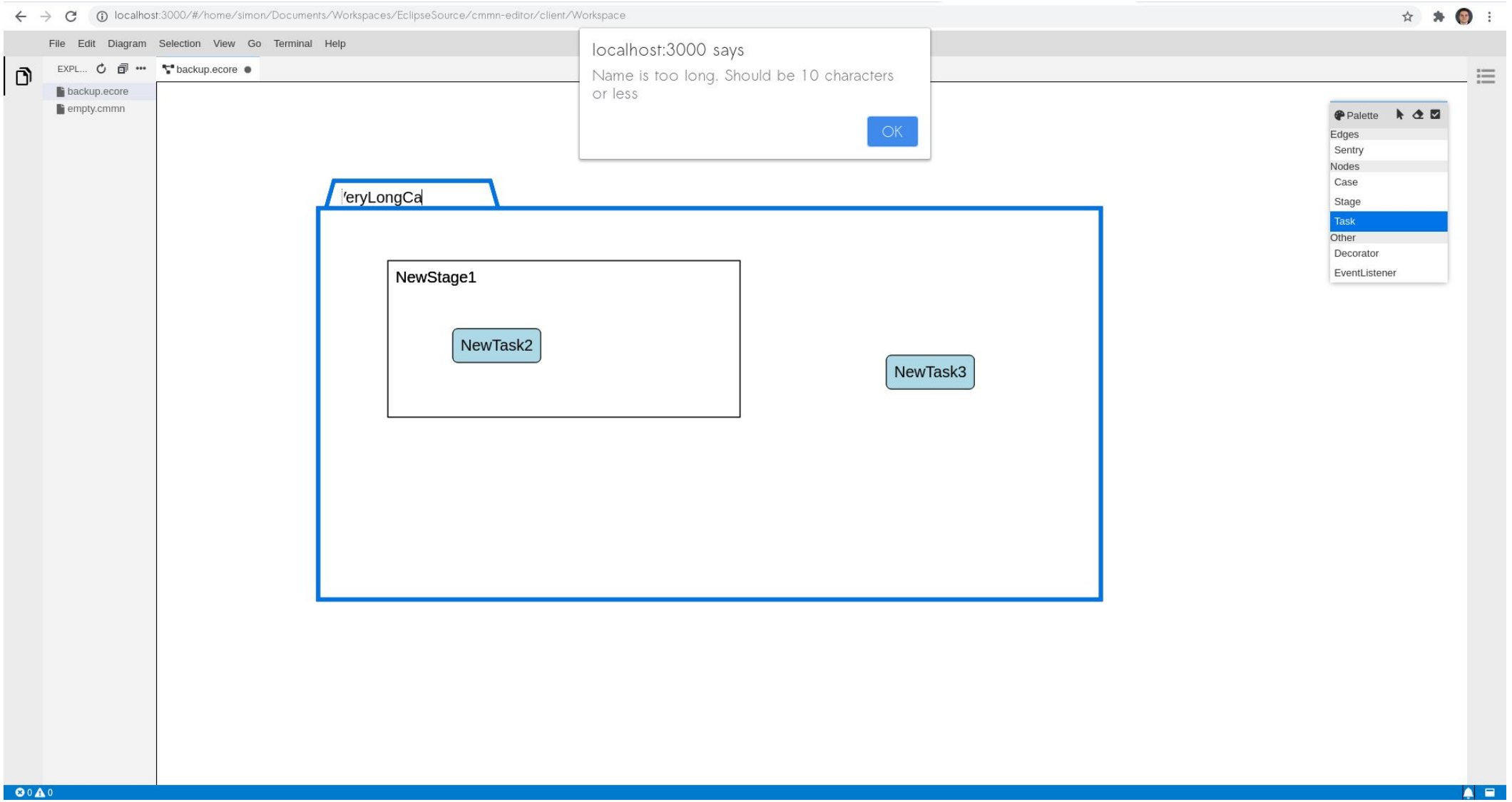
© sebis

Creation of Elements

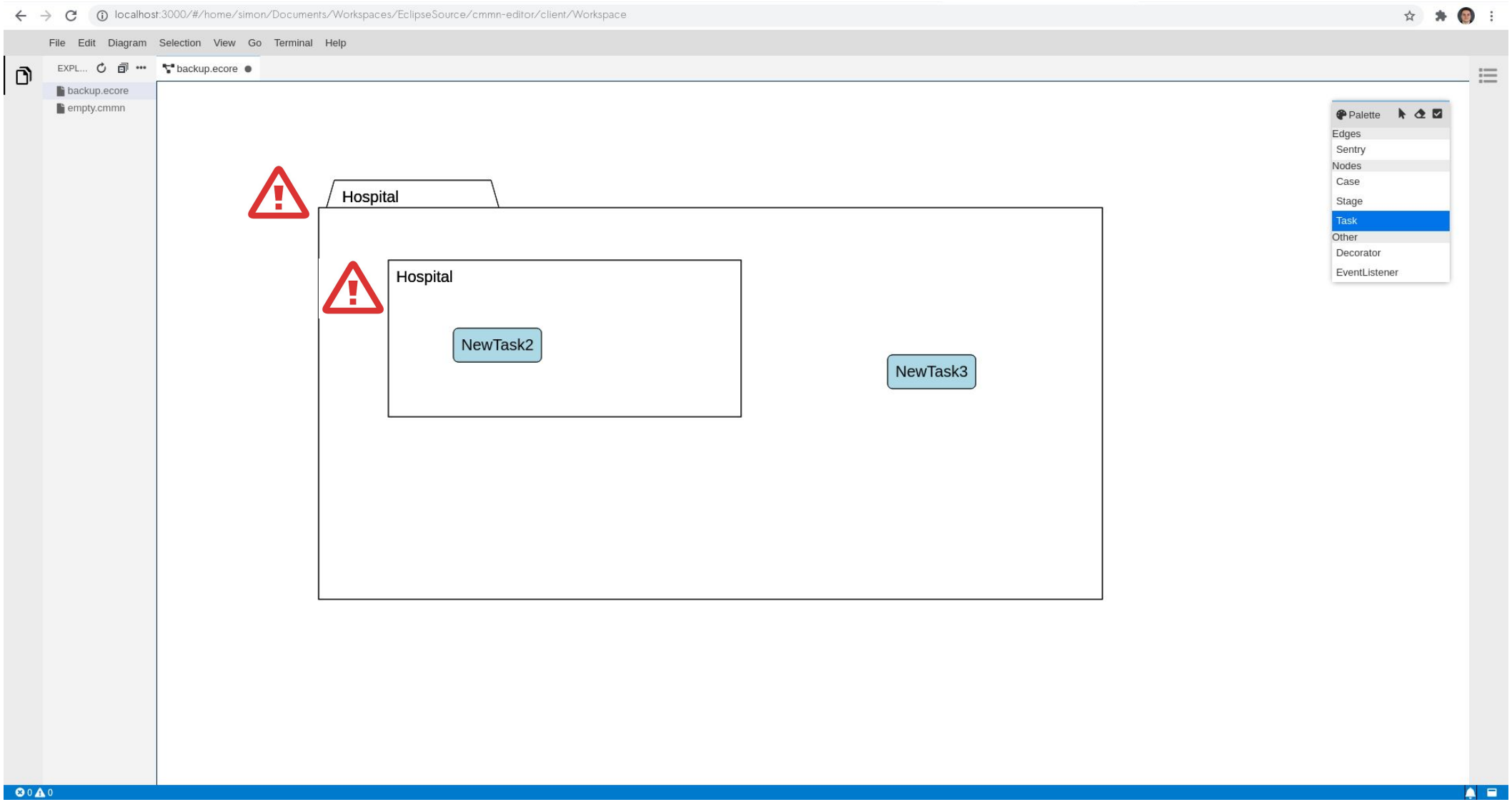


© sebis

Input Validation



Global Validation



© sebis

Outline

Motivation

Research Questions

Process Description

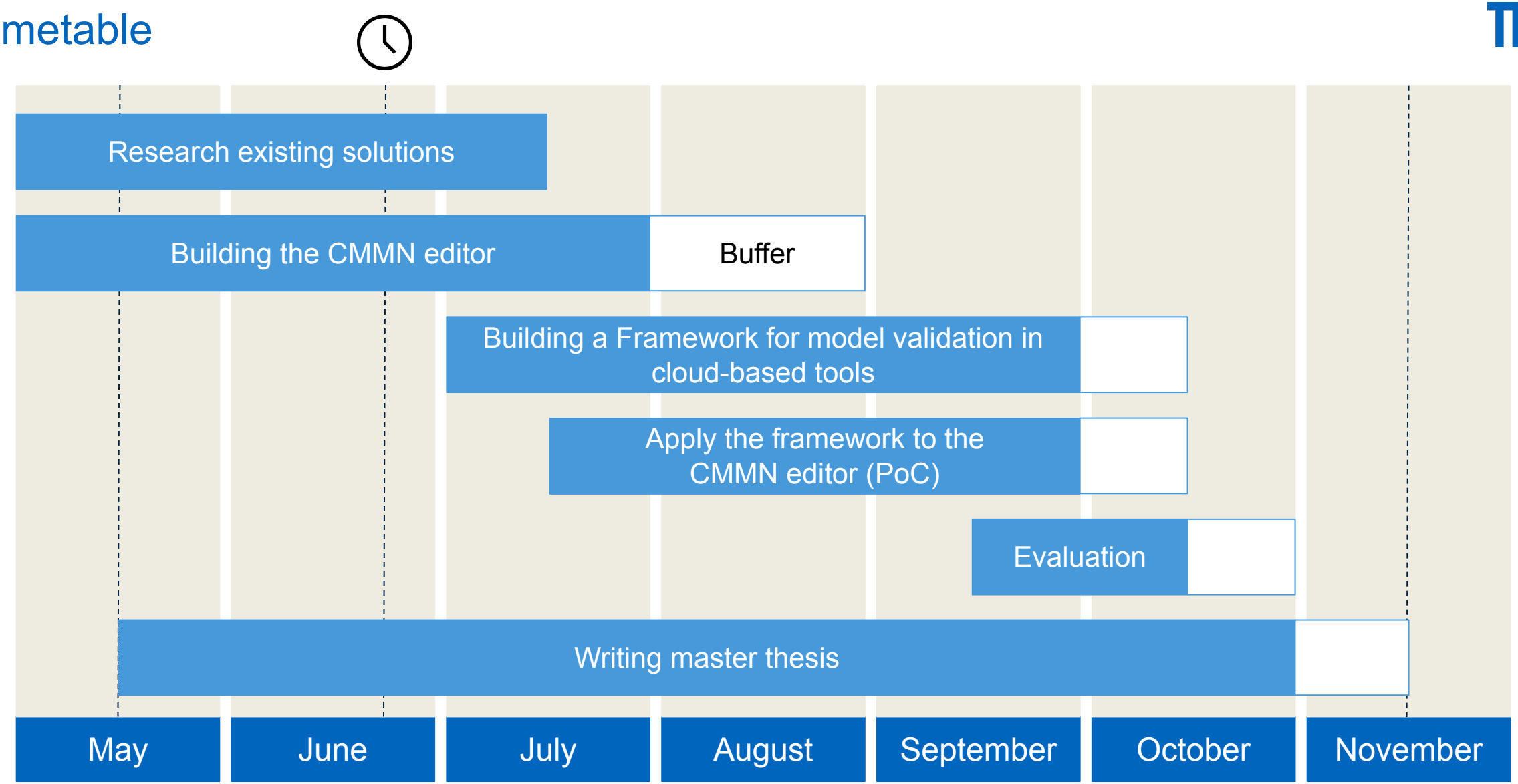
Expected Outcome

First Results

Timetable

© sebis

Timetable



© sebis

References

[1] Haibing, Li, Zhu Ning, Lei Yonglin, Li Xiaobo, and Zhu Yifan. “EMF Based Validation Methods of the Static Semantics of Models.” In *2015 2nd International Conference on Information Science and Control Engineering*, 207–11, 2015. https://doi.org/10.1109/ICISCE.2015.53.

[2] Gogolla, Martin, and Frank Hilken. “Model Validation and Verification Options in a Contemporary UML and OCL Analysis Tool,” n.d., 16.

[3] Nielsen. “10 Usability Heuristics for User Interface Design”. 1994.

https://www.nngroup.com/articles/ten-usability-heuristics/.

© sebis

B.Sc.
Simon Graband

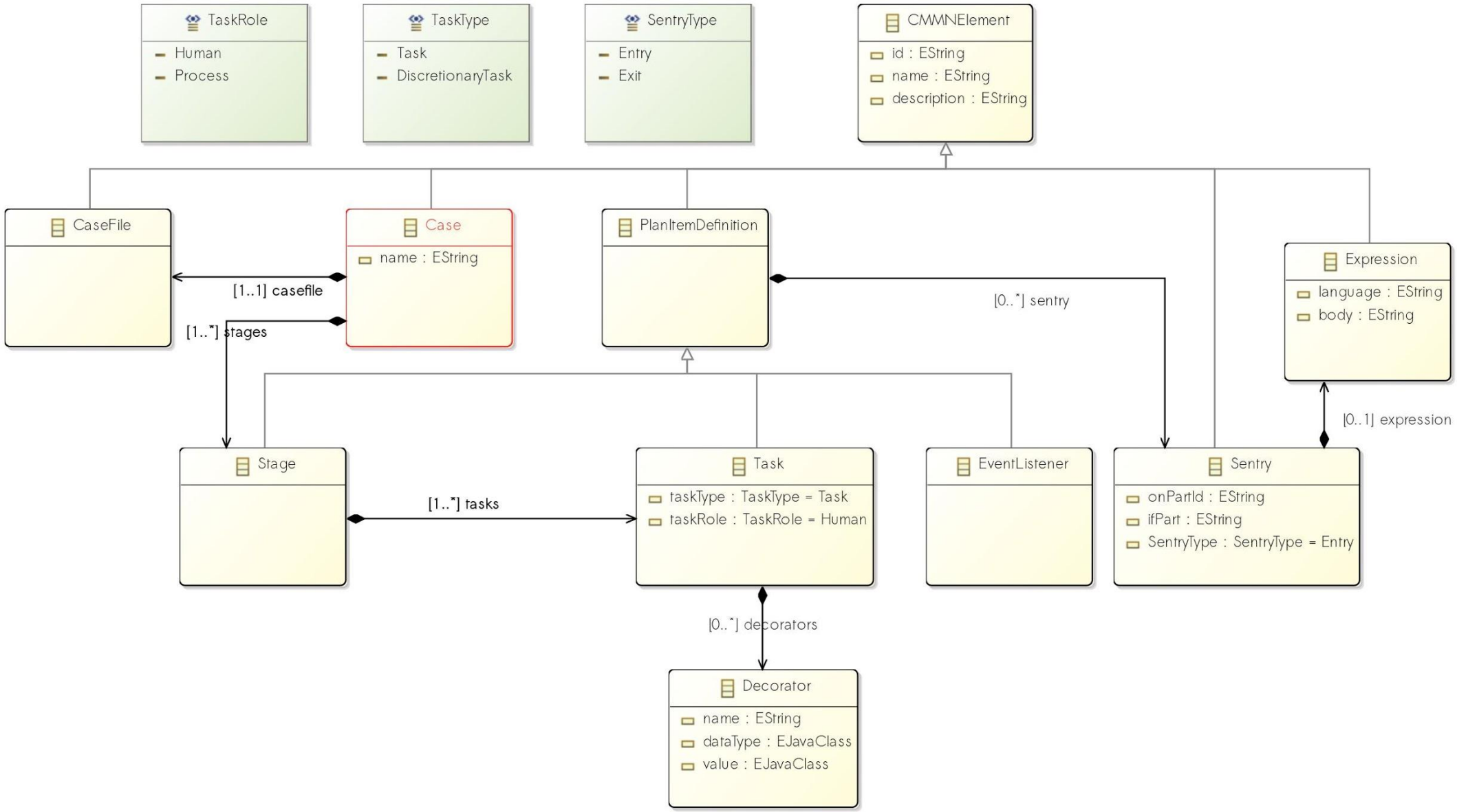
Technische Universität München
Faculty of Informatics
Chair of Software Engineering for Business Information Systems

Boltzmannstraße 3
85748 Garching bei München

Tel +49.89.289.17132
Fax +49.89.289.17136

simon.graband@tum.de
wwwmatthes.in.tum.de

CMMN Metamodel



Scope of constraints

Language-specific

- Invalid Nesting
  - Obj. cannot be nested within Obj.
- Invalid Hierarchy
  - cannot include Case inside a Stage
- Limited number of subelements
  - 1 Case has only 1 CaseFileItem
- Invalid Data Type
  - Syntax/Grammar of the Element
- Length of value
  - Limit name to X chars
- Minimum Objects
  - Stage cannot be Empty
  - Case must have a Stage
- Unused Objects
- Cyclic Process
- Reference to non-existing objects

User-defined

- Invalid data type of user-defined elements
  - validate data type within in the user-defined element (decorator, task type)
- Violation of rules in user-defined setting
  - sentry (edge) can only be connected to stages, not to tasks
- Cases must have an exit criterion

© sebis

Nielsen’s 10 Usability Heuristics

Especially relevant for validation:

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition other than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose and recover from errors
- Help and Documentation

© sebis