Interactr

Iteration 2

Jelle de Coninck, Hannes De Smet, Bruno Vandekerkhove, Shani Vanlerberghe

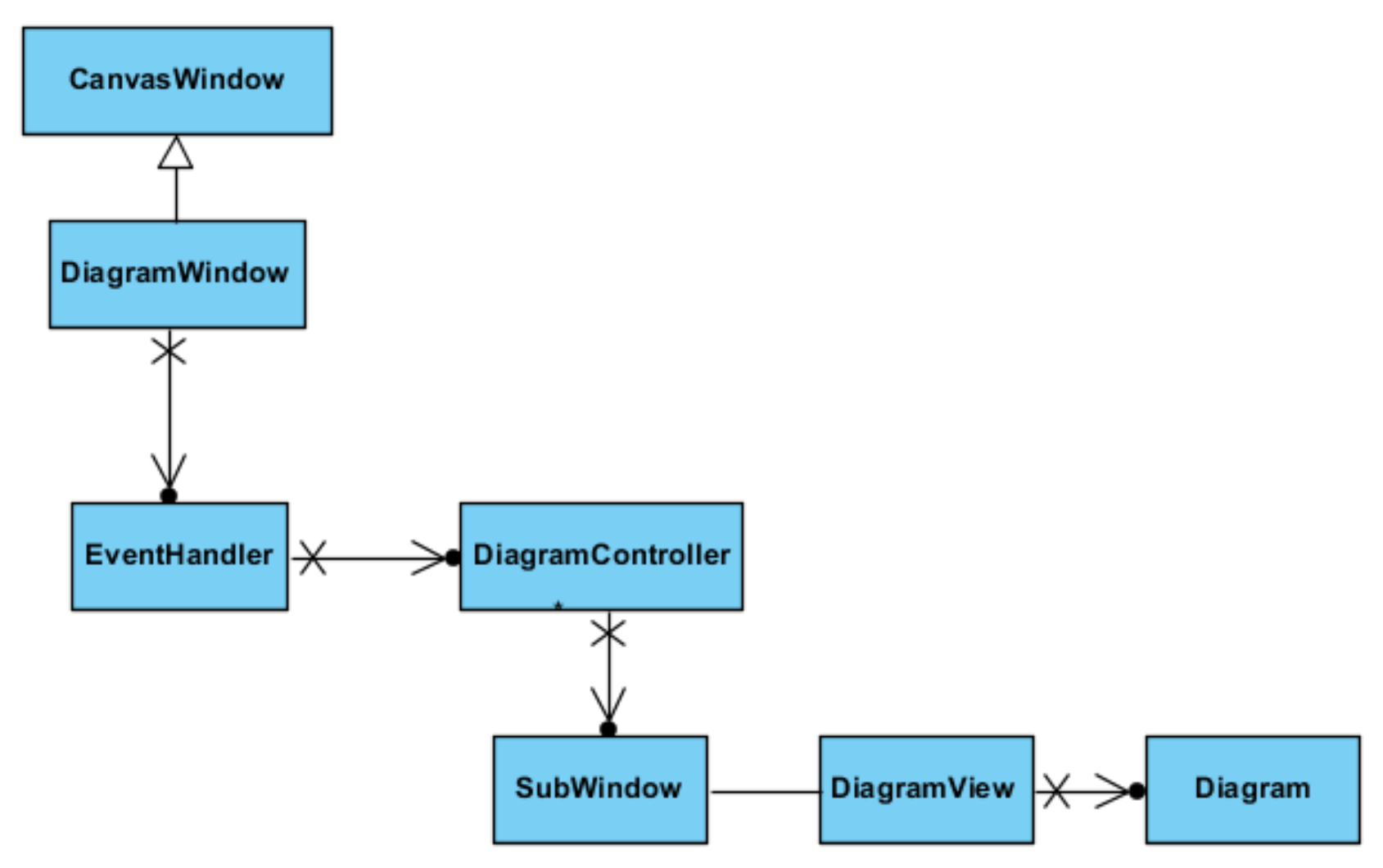
April 25, 2018

KULeuven

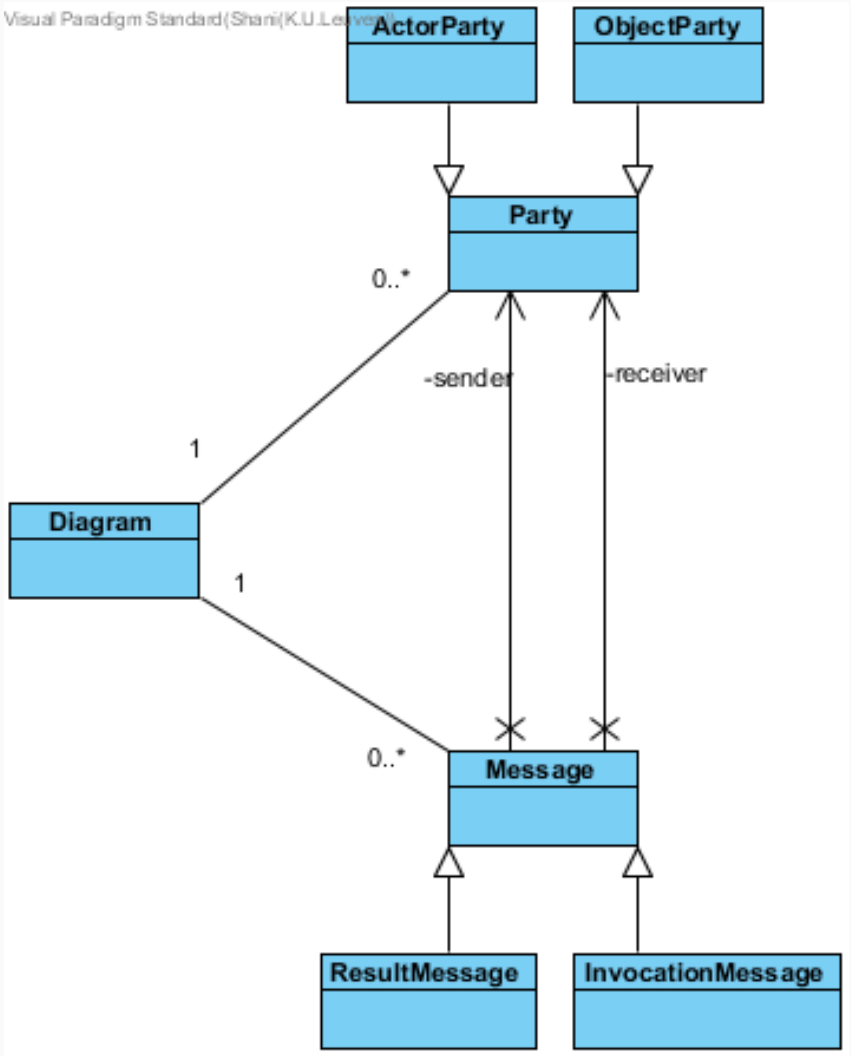
Table of contents

Design

Design - Typical Flow



Design - Domain Model (Mapping)



- Low representational gap
- Virtually nothing changed
- Prefixes fixed

Design - DiagramController

- Used to be quite central (managed `Diagram` itself)
- Has sorted list of subwindows which keep track of diagram views
- Each `DiagramView` manages the `Diagram` directly
- Synchronisation done with Observer pattern (see later)

`Controller - Low Coupling`

Design - Subwindows

`DiagramController` uses sorted list of subwindows (drawn in reverse order)

- Each subwindow has frame
- Responsible for moving and resizing this frame
- Responsible for selection and editing of the labels
- Use of bitflags

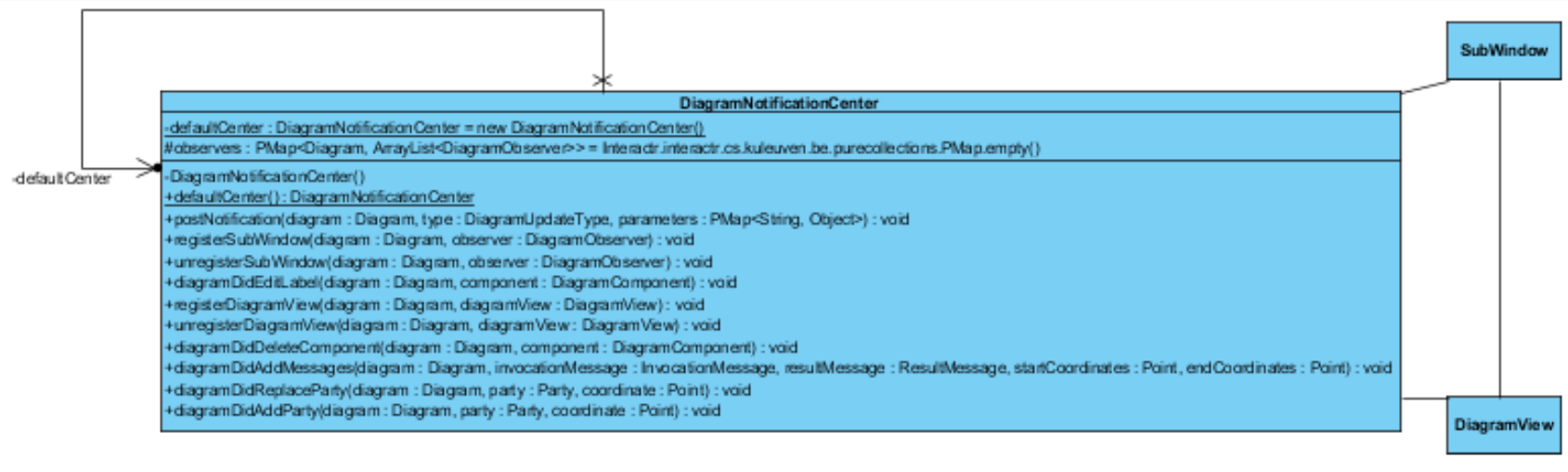
$\Rightarrow$ Sequence Diagram (Resize Subwindow)

Design - Drawing

- `DiagramController`, `SubWindow` and `DiagramView` draw themselves.
- Use of clipping rectangle set on `PaintBoard`

`Information Expert - Polymorphism`

Design - Synchronisation



Pure Fabrication - Observer (GoF)

Design - GoF Patterns

4 patterns :

- `PaintBoard` : Facade
- `proposedFigure()` : Factory Method
- `DiagramNotificationCenter` : Singleton & Observer

Design - Coding

- Refactored here and there (eg. `Extract Method` in response to `Duplication`)
- Defensive programming - use of exceptions
- Everything is documented (informally)

Design - Edit Label

$\Rightarrow$ Sequence Diagram (Edit Label)

Extensibility

Extensibility - Observer

3 methods considered :

- `DiagramObserver` interface with one method `diagramDidUpdate`, passing along the update type and associated parameters (in a dictionary)
- Keeping `SubWindow` and `DiagramView` in separate lists as observers
- `DiagramObserver` interface with methods with `default` implementations

Second method currently in use, last two methods similar in terms of extensibility. First method causes messy code with lots of `instanceof`.

Testing

Testing - Methods

Combination of :

- Recordings
- Step-by-step recordings
- Unit tests

Testing - Coverage

98% classes, 87% lines covered in package 'be'

Element	Class, %	Method, %	Line, %
domain	100% (8/8)	98% (53/54)	98% (187/189)
exceptions	100% (10/10)	100% (3/3)	100% (15/15)
purecollections	100% (27/27)	86% (137/158)	88% (367/416)
resources			
ui	96% (32/33)	87% (251/286)	85% (1067/1253)

Project Management

Project Management - Overview

- +- 50 hours per person
- Started early with design
- Code was nearly finished early on
- Last weeks were spent on refactoring and testing

Demonstration