Towards Declaratively Defined Graphical Editors

O. Beaudouix, F. Jouault, ESEO/TRAME
Outline

● Anatomy of a Graphical Editor
● Interactors
● Prototype 1: Loa-JS
● Prototype 2: TCSVG
● Prototype 3: AOF Diagram
● Perspectives
Anatomy of a Graphical Editor

- Model (i.e., abstract syntax)
- Views
  - Templates (i.e., “shape” groups)
    - Transformed from model
  - Geometrical constraints (on shapes using model)
  - Auto-disposition (i.e., layout, routing)
- Interactors: interactions to actions
  - Actions: changes to model
  - Interactions: user events on views

E.g., geometric constraints from semantics (e.g., align elements having a specific relationship)
Interactors

● Maturity
  ○ Defined in the ‘80 by Brad Myers
  ○ Transform an interaction into an action (on a domain object)
  ○ Especially useful in the scope of diagram authoring apps

● Implementations
  ○ Loa prototype provides simple and predefined interactors
    ■ Users parameterize interactors by leveraging active operations

● Planned
  ○ Support modern interactors (e.g., touch-based ones) - easy improvement

● Extensions to Consider
  ○ Support user definition of interactors/interactions
Prototype 1: Loa-JS
Web-based Graphical Editors

- Model: custom (JavaScript-based, EMF-like)
- Views
  - Templates: XML (e.g., HTML, SVG)
    - incrementally transformed with active operations
  - Geometrical constraints: recently used Cassowary
  - Auto-disposition: not supported
- Interactors: library of predefined and parameterizable interactors
  - Actions: in JavaScript
Loa-JS Demo

Graf

Details on flyer:
 prototype 2: tcsvg
web-based graphical viewers

- model: none
- views
  - templates: in svg
  - geometrical constraints: using cassowary
  - auto-disposition: not supported
- interactors
  - actions: not applicable
  - interactions: only drag-drop-move

TCSVG (Trame CSVG) is a prototype reimplementation of CSVG (SVG + constraints) in Javascript, and based on Cassowary. It is not intended to be compatible with the original CSVG.
TCSVG Demos

- Graf
- GridBagLayout
- UML Diagrams
  - Composite Structure
  - Sequence: 1, 2, 3
  - Activity: 1, 2, 3, 4
Prototype 3: AOF Diagram
Graphical Editors in Java

- **Model**: EMF
- **Views**
  - Templates: in Java
    - incrementally transformed with active operations
  - Geometrical constraints: using AOF
  - Auto-disposition: not supported
- **Interactors**: a few predefined interactors
  - Actions: in Java
AOF Diagrams Demo

- Graf
- UML Diagrams
  - Sequence
  - State (with actual backing EMF model)
Some Wild Ideas

- Web-based interactive viewers
  - e.g., for Papyrus exports, letting users change layout outside of the tool
- Web-based Papyrus (SVG-based)
- Projectional editors
  - e.g., for textual elements in UML syntax such as Features, Transition labels, etc.
  - to provide document-like views on models