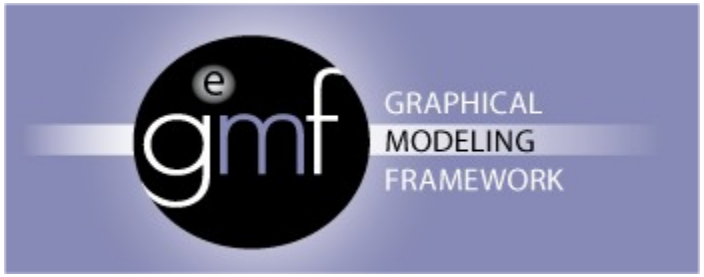
Migrating your Graphical Editor to the Eclipse Graphical Modeling Framework

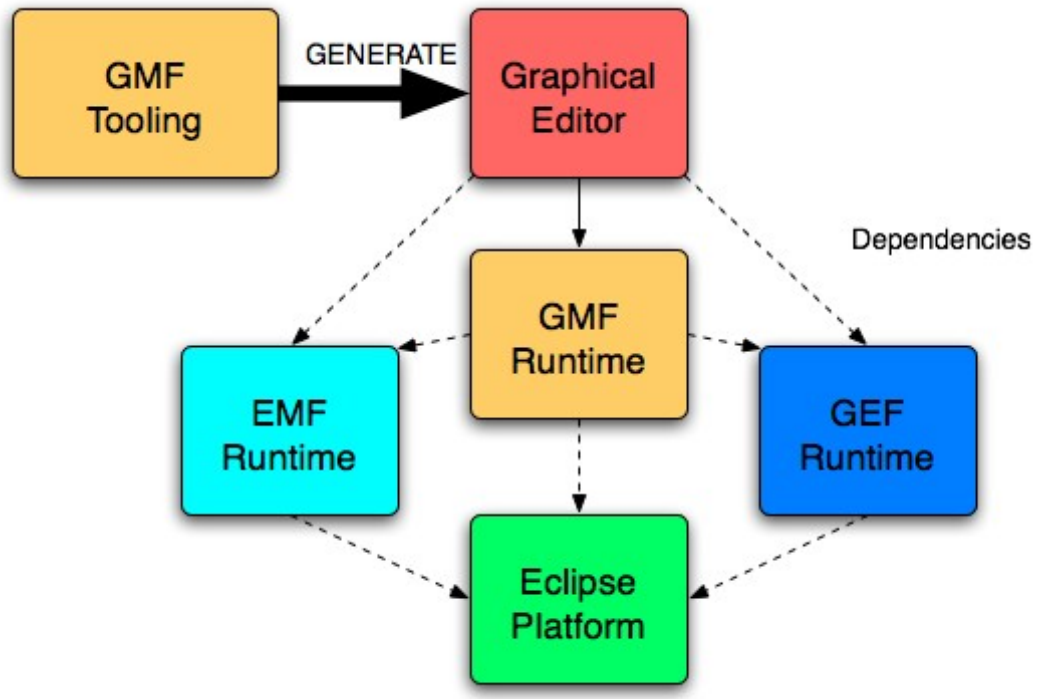


Anthony Hunter

IBM Rational Software

Graphical Modeling Framework Overview

- The Eclipse Graphical Modeling Framework (GMF) provides a generative tooling and runtime infrastructure for developing graphical editors based on EMF and GEF.



Eclipse Modeling Framework (EMF) Review

- Tools
  - Generator for:
    - The Java™ implementation of models described by:
      - XML Schema, IBM™ Rational™ Rose™ Model, Annotated Java
    - Basic Eclipse tree-based editors to edit instances of generated models
  - Transformation engine that supports JSP-like templates (JET)
- Runtime
  - Base classes and utilities for generated models' implementation. Support:
    - Notification mechanism
    - Reflection mechanism
    - Persistence mechanism (XMI by default)
    - Meta-model integration
    - Basic undo/redo support
    - Basic workbench integration
  - Dynamic model creation API

Graphical Editing Framework (GEF) Review

- GEF is an MVC-based framework to create graphical editors
  - GEF editors are integrated within Eclipse and feature:
    - Palette
    - Undo/Redo
    - Overview pane
    - Rulers
    - Guides
    - Snap-To-Grid/Geometry
    - Properties View
    - Etc…
- GEF low-level rendering and layout is called Draw2D
  - Draw2D is a lightweight toolkit built over SWT

Graphical Modeling Framework Overview

- GMF has two main components:
  - GMF Runtime
    - Framework that binds the capabilities of EMF and GEF
    - Provides significant out of the box diagramming capabilities
    - Provides a service layer designed for extensibility
  - GMF Generation Tooling
    - Model driven approach to generate graphical editors
      - Domain model (EMF)
      - Graphical definition model
      - Tooling definition model
      - Mapping definition model
    - Code generation that targets the GMF Runtime

Graphical Modeling Framework Overview

- When should I consider GMF?
  - Your domain requires the need to communicate ideas graphically.
  - You would like to sketch a graphical editor and produce fast results.
  - You need your editor to be extended by third-parties.
  - Your domain is aligned with terms like:
    - [Link | Edge | Node | Shape | Connection | Compartment]
  - You are interested in model driven development.

Graphical Modeling Framework Overview

- When should I not consider GMF?
  - If you have yet another GEF-oriented framework to handle your diagrams
    - *…and can the afford man-years of labor required to support it.*
  - You are not interested in the standard diagram features.
  - Your diagram is form or table based.
    - Where are the [Link | Edge | Connection] ?
  - You do not want to use EMF.
  - You only need static images with nodes and connections.
    - Visualizing the domain data rather than editing it.

Graphical Modeling Framework Runtime Architecture

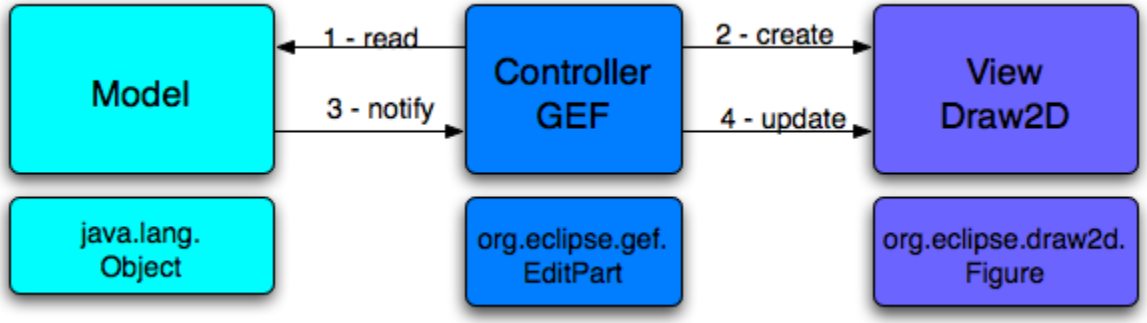
- Component based
  - Separation into layers of function:
    - common, notation, diagram, gef, draw2d, emf
    - IDE versus RCP
- Consistent with architecture of dependent features
  - EMF: EObject dependency
  - GEF: follows MVC pattern
- A standardized model to describe diagram elements
  - Semantic and notational (diagram) models are distinct
- Service based extensibility
  - Enables open and extensible graphical editors

Graphical Modeling Framework Runtime

- History of the GMF Runtime
  - 2000 : Rational XDE™
    - Eclipse based graphical editors : lessons learned.
  - 2002 : IBM WebSphere Studio Application Developer™ version 5
    - First version of a framework for UML diagrams.
  - 2004 : IBM Rational Software Architect™ version 6
    - Second release of the framework: UML, Database, Business diagrams
  - 2006 : IBM Rational Software Architect version 7
    - Moved the framework to Eclipse as part of GMF 1.0.
  - 2007 : GMF 2.0 : GMF Runtime 1.0.x
  - 2008 : GMF 2.1 : GMF Runtime 1.1
- Long history and industry proven architecture.
  - Now leveraged by products from IBM and Borland™ (and many others).

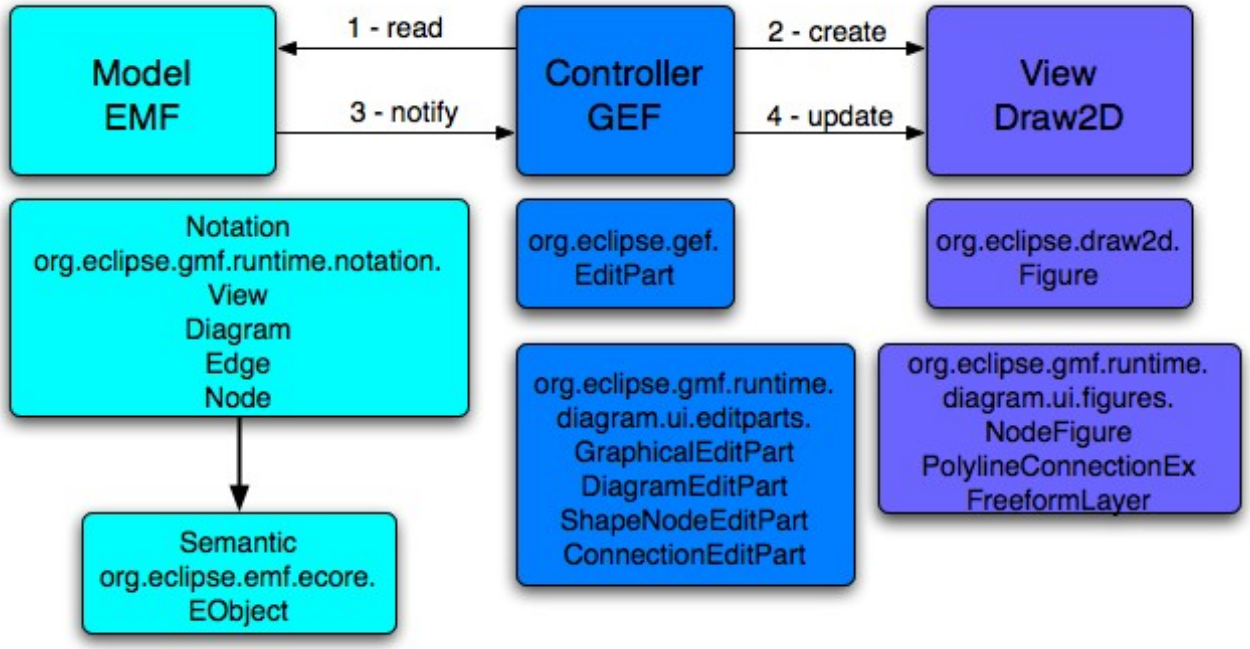
GEF Architecture Review

- Model – View – Controller Architecture



GMF Runtime: Diagram Architecture

- Follows GEF MVC Architecture



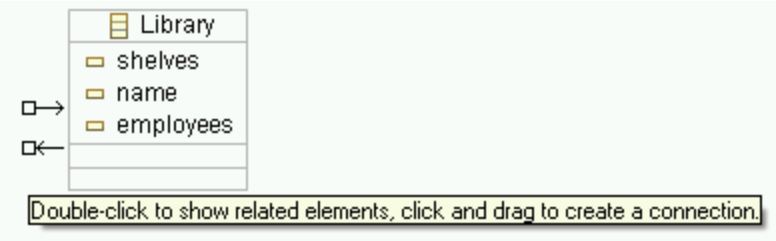
GMF Runtime: Reusable Components

- Direct Text Editing:

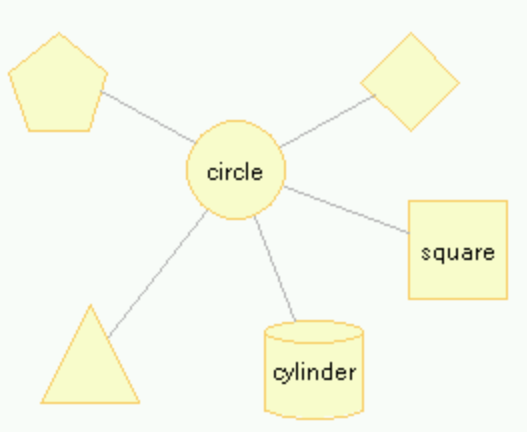
- Action Bars:

- Collapsible Compartments:

- Connection Handles:

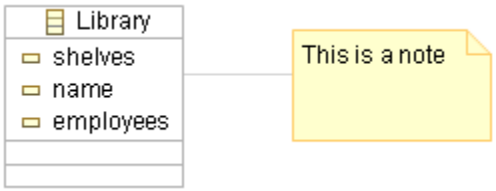


- Geometric Shapes:

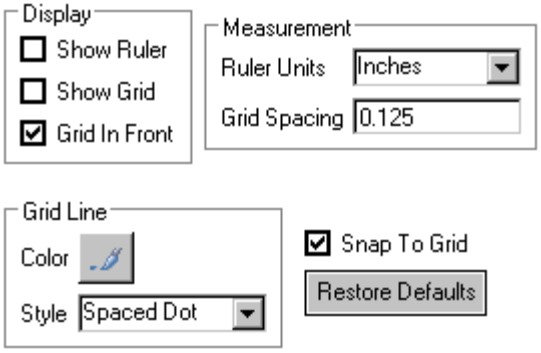


GMF Runtime: Reusable Components

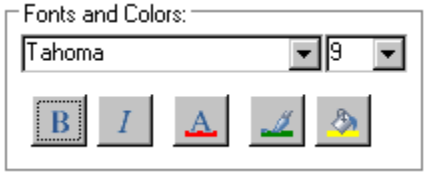
- Text, Note and Note Attachments:



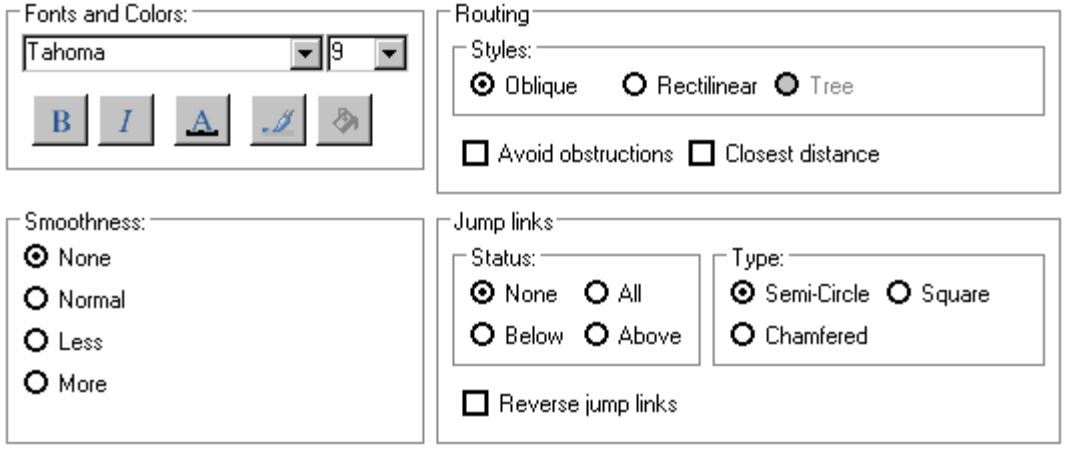
- Ruler and Grid Support:



- Shape Appearance Properties:



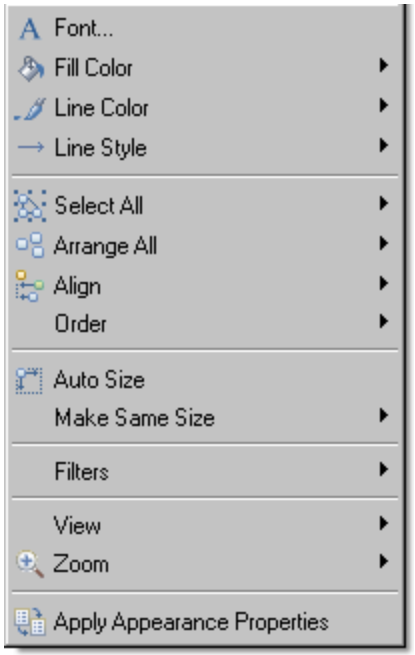
- Connection Appearance Properties:

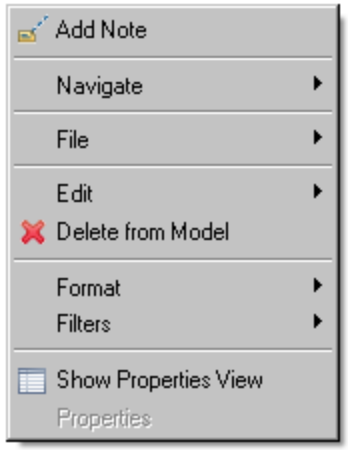


GMF Runtime: Reusable Components

- Common Diagram Toolbar:

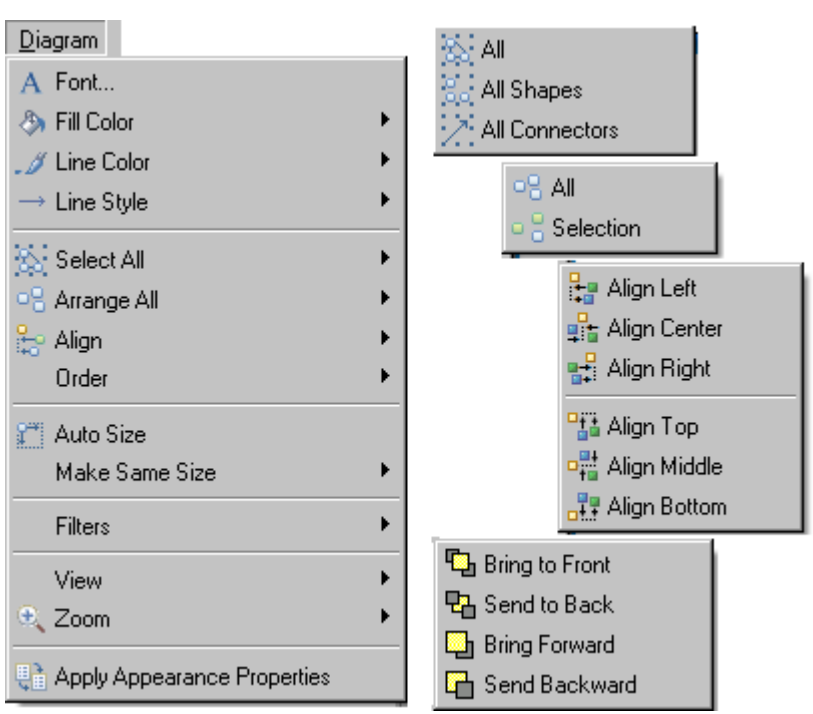
- Common Diagram Menu and Diagram Popup Menu:



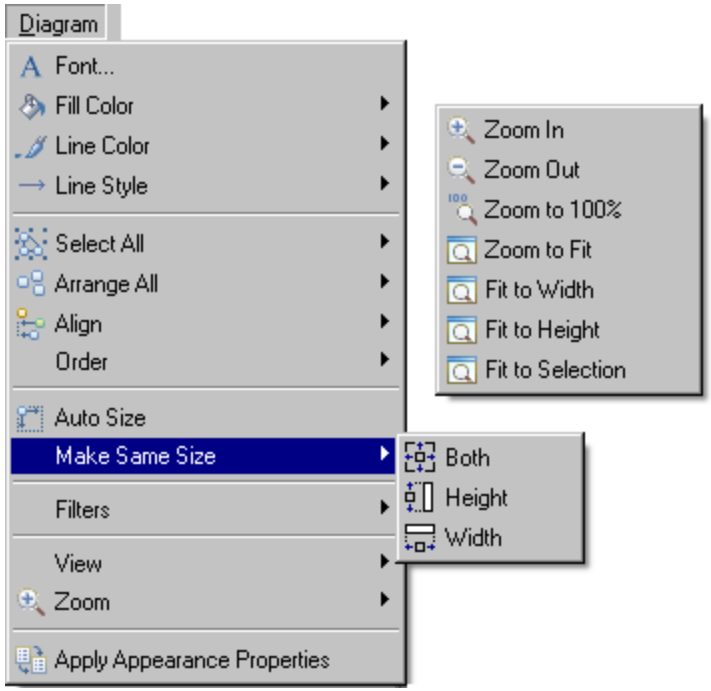


GMF Runtime: Reusable Components

- Select, Arrange, Align, Group:

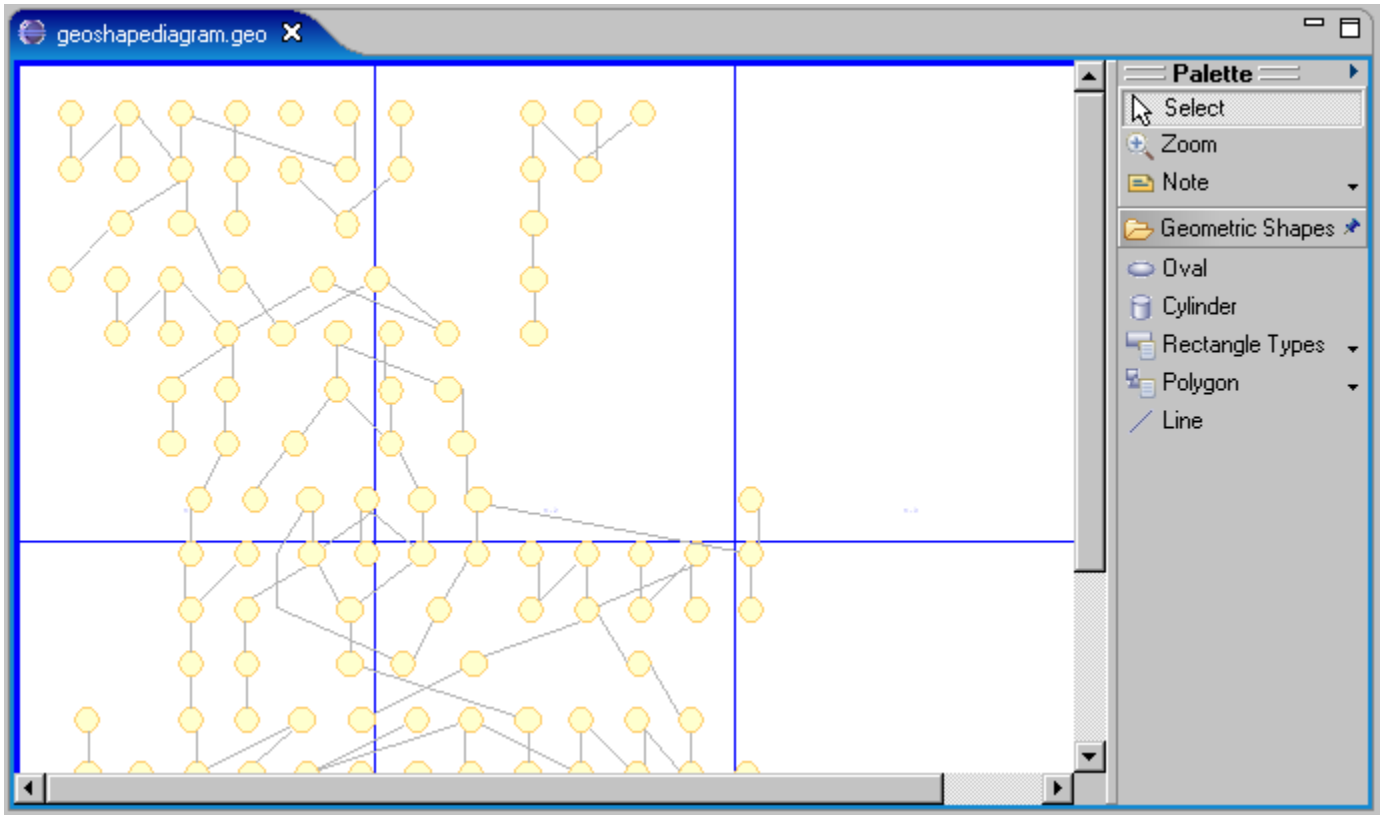


- Size and Animated Zoom:



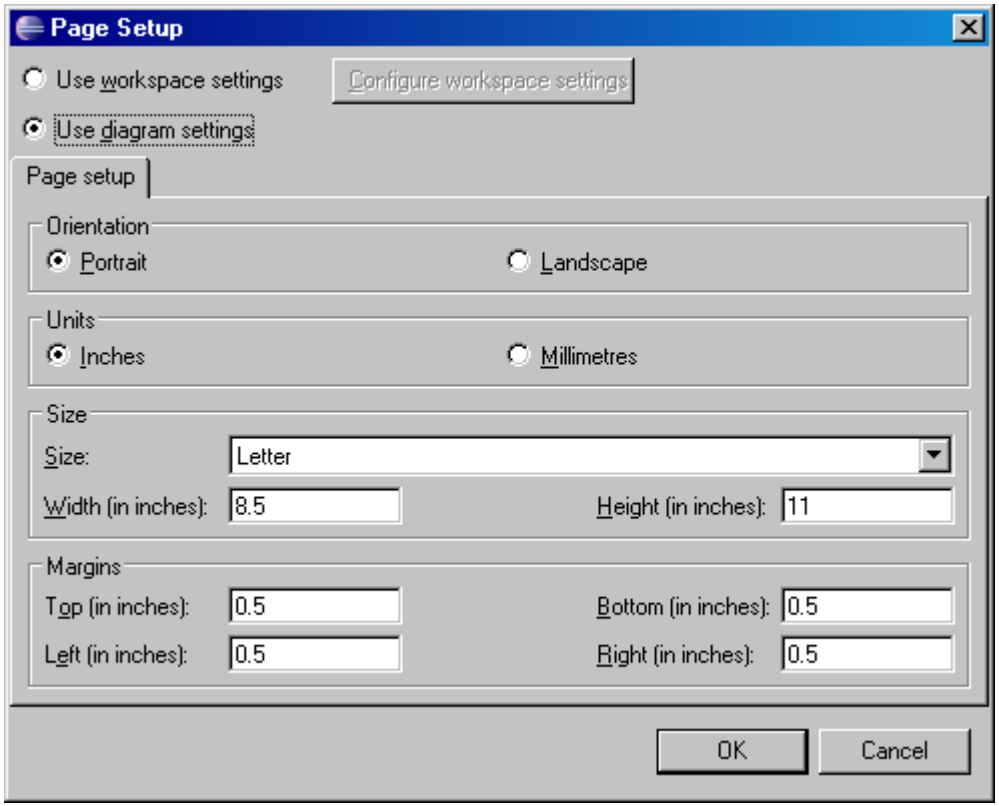
GMF Runtime: Reusable Components

- Printing Support: Show Page Breaks and Print Preview:



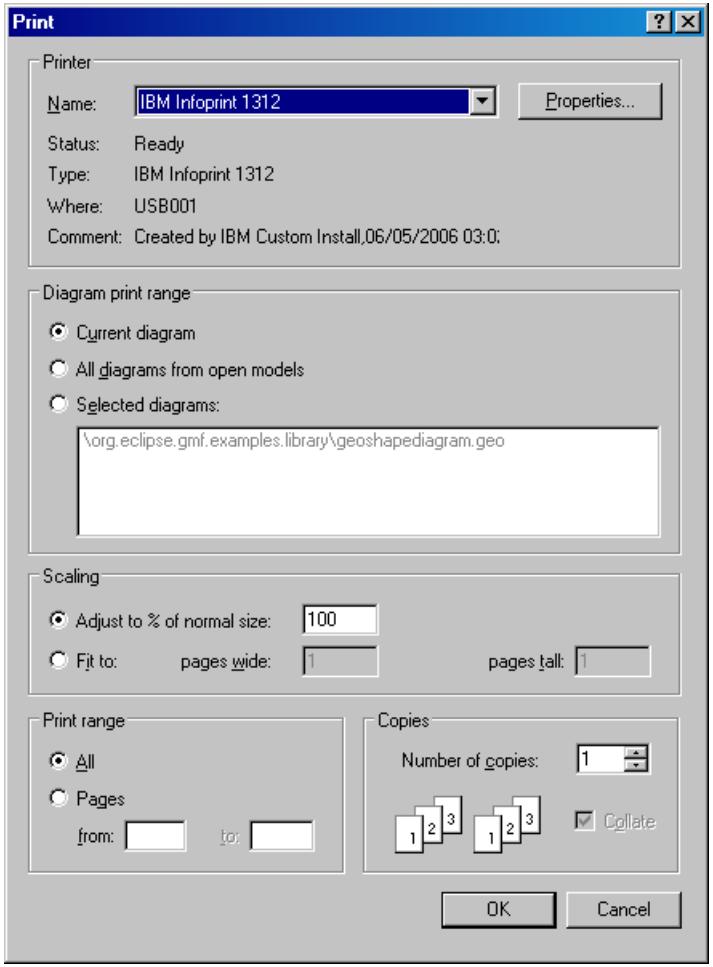
GMF Runtime: Reusable Components

- Printing Support: Page Setup Dialog:



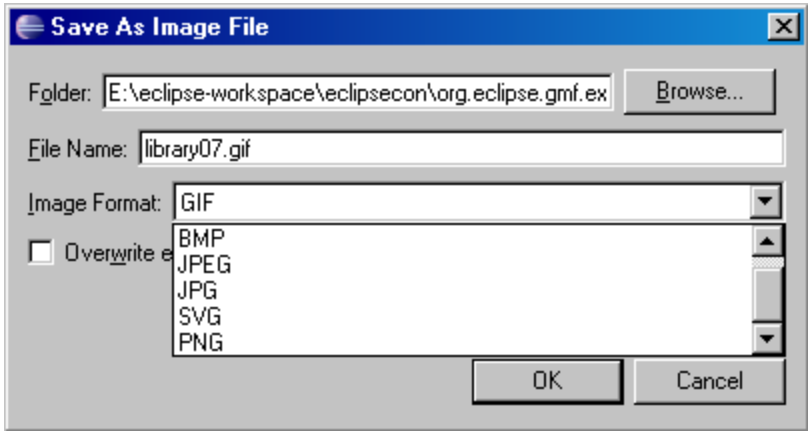
GMF Runtime: Reusable Components

- Printing Support:



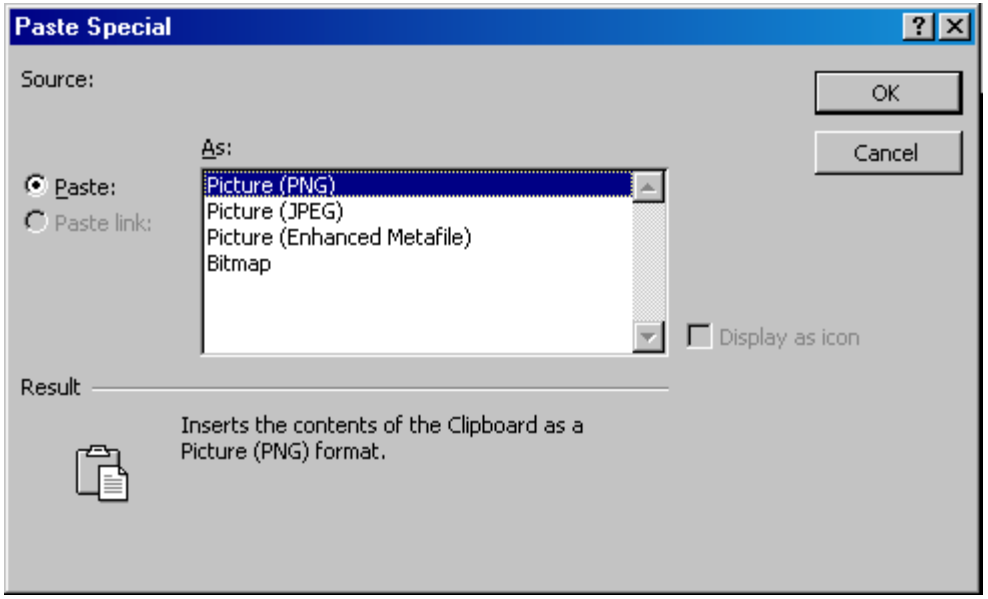
GMF Runtime: Reusable Components

- Save diagram to image file
  - Many formats
  - Export to HTML (split large diagrams)



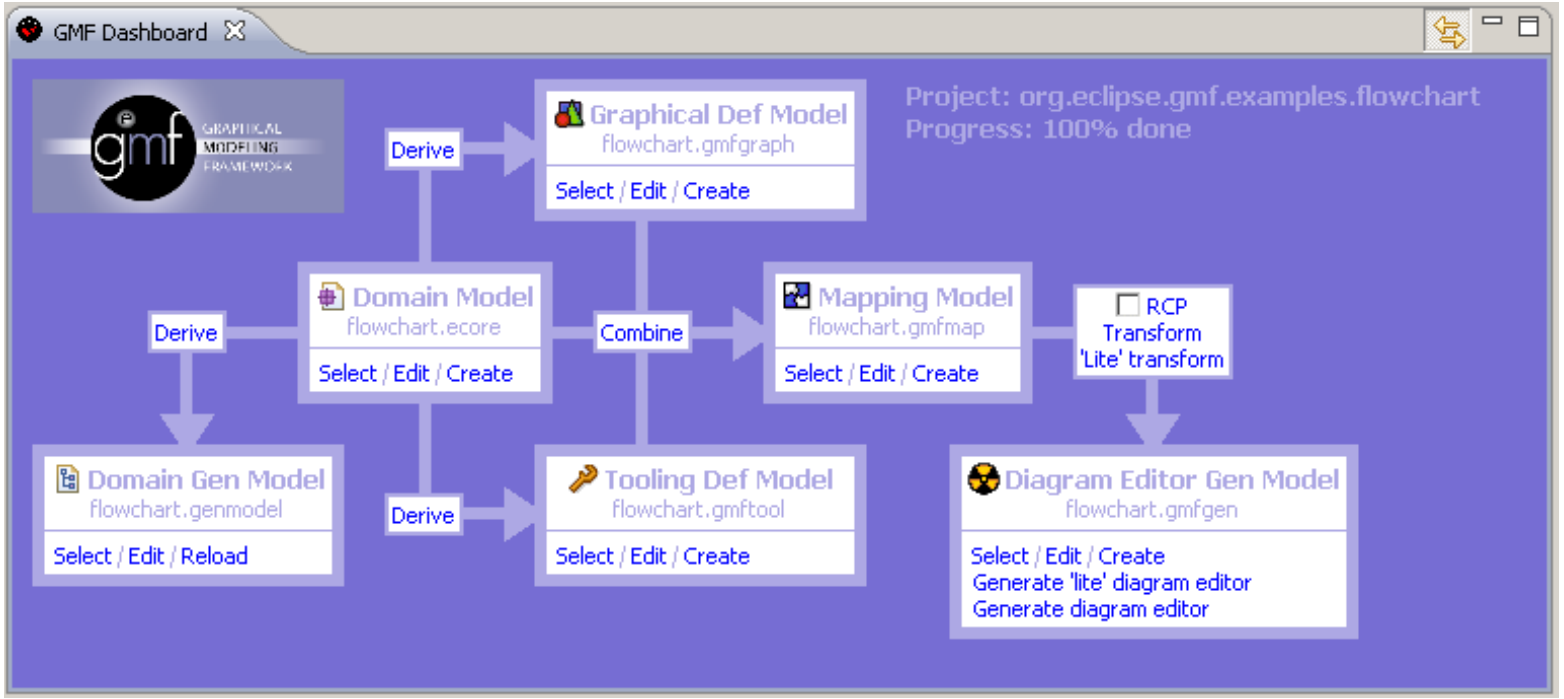
GMF Runtime: Reusable Components

- Copy and paste to clipboard; system clipboard formats:



GMF Generation Tooling Overview

- Provides a model-driven approach to generating graphical editors.
- GMF Dashboard:



GMF Generation Tooling: Domain Model

- EMF model for your domain.
- Captures non-graphical aspect of the diagram.
- Metamodel – EMF's ECore
- Many ways to define
  - Import with EMF (Rose, annotated Java, UML2)
  - Design with standard EMF tree like editor.
  - Design with the GMF Ecore diagram editor.
    - available as part of the GMF SDK

GMF Generation Tooling: Graphical Definition Model

- A model to define graphical elements:
  - Figures
  - Nodes
  - Connections
  - Compartments
  - Labels
- Figure Galleries allow for reuse
  - Share common shapes, labels, connections, etc.
  - Custom figures to allow any GEF figure
- Wizard provides starter definition from the domain model.
- GMF SDK provides a gmfgraph diagram editor.

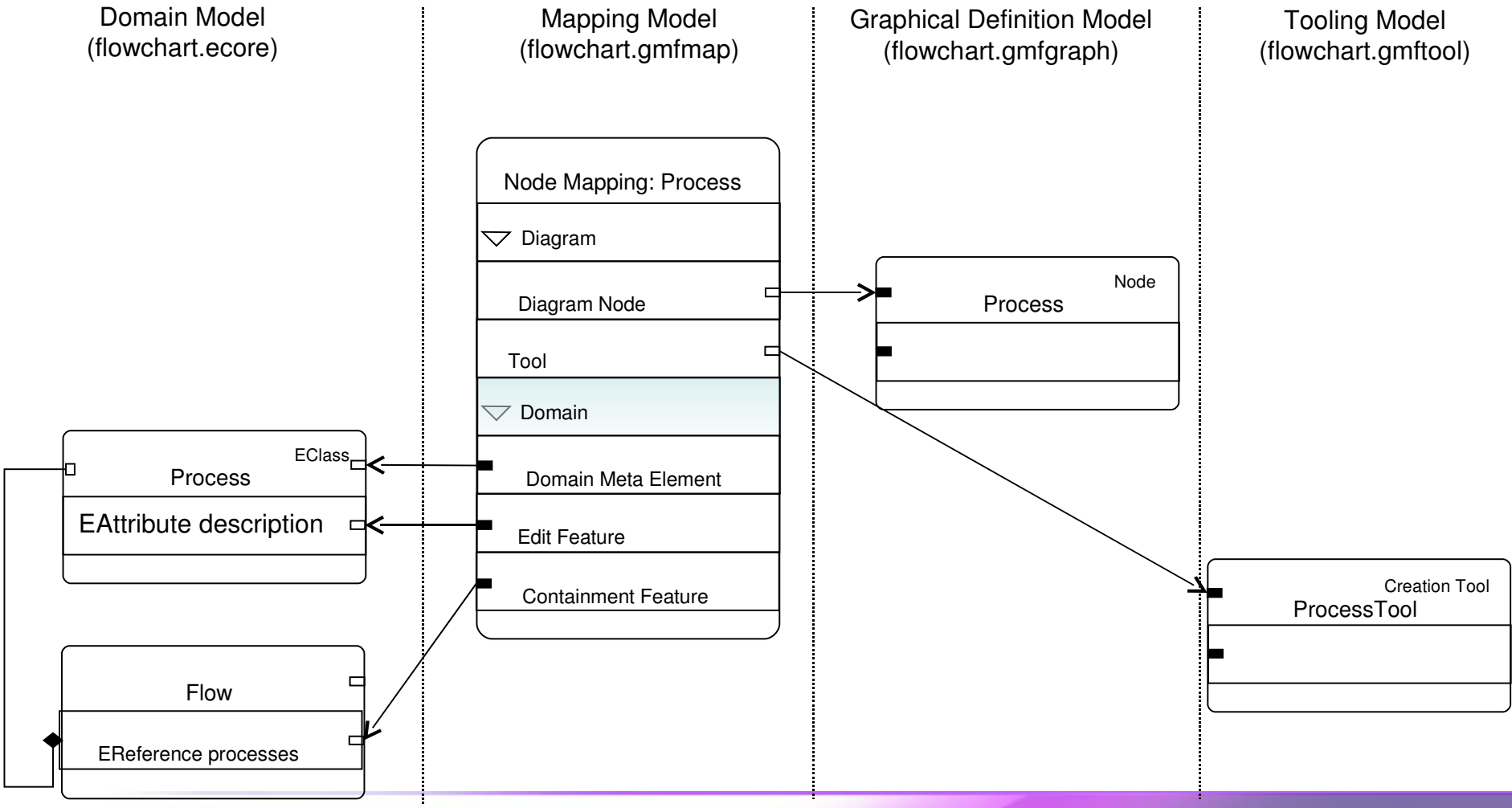
GMF Generation Tooling: Tooling Definition Model

- A model to define as part of tool registry:
  - Palette
  - Palette Tools
  - Menus; context, main, popup
  - Actions
  - Toolbars
- Wizard provides starter definition the domain model

GMF Generation Tooling: Mapping Model

- A model to specify the relationship between:
  - Domain elements (*.ecore)
  - Graphical elements (*.gmfgraph)
  - Tooling elements (*.gmftool)
- Allows for audit & metric definitions (both domain and diagram)
  - Leverages EMF Validation
- Domain can be constrained and initialized
  - Leverages EMF OCL
- Wizard provides starter mapping from the domain, graphical, and tooling models

GMF Generation Tooling: Mapping Model Example

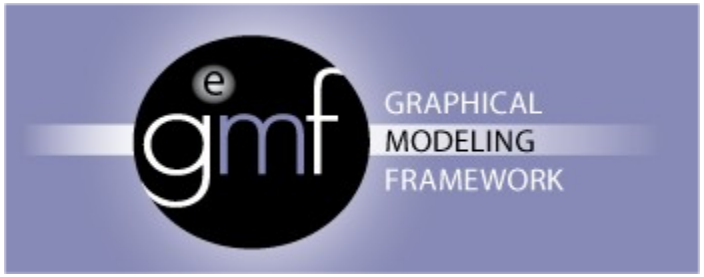


GMF Generation Tooling: GMF Generation Model

- A model used to specify code generation parameters
  - Similar to EMF *.genmodel
  - Result of transformed mapping model
- Code generation using Xpand
  - Substitute provided templates with your own
- Alter properties for generation to suit your taste
  - Plug-in provider name, ID, package namespace, etc.
- Specify runtime options
  - Print support, validation support, file extension, etc.
  - Diagram persistence (one file for diagram and domain files)
  - Tabbed Properties View
  - Project Explorer support

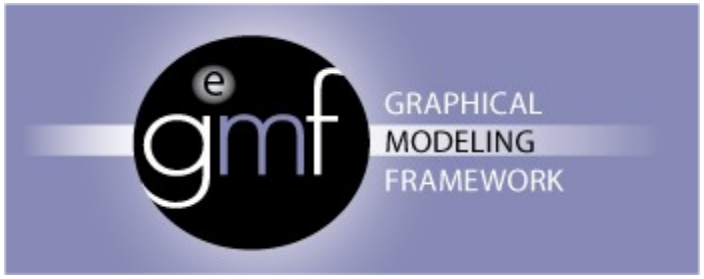
Migrating your Graphical Editor to the Eclipse Graphical Modeling Framework

- Demonstration
  - Creating a graphical editor



Migrating your Graphical Editor to the Eclipse Graphical Modeling Framework

- Demonstration:
  - Custom shapes for the graphical editor



Legal Notices