Gruppe: HK-07-4
Gruppenleiter: Stanley Hillner 07.05.2007

Entwurfsbeschreibung

1. Allgemeines

Die drei Editoren wurden von uns mittels Eclipse GMF erstellt und für deren Nutzung ist die Eclipse IDE vorausgesetzt. Die Editoren können als Plug-Ins in Eclipse aufgerufen werden.
Mit dem ersten Editor (Petrinetze) ist es möglich, eine leichtere Version der bekannten Petrinetze zu modellieren.
Mit den beiden anderen kann man jeweils ein Rechteck auf die Zeichenfläche des Editors zeichnen. Die Editoren führen dann intern verschiedene Operationen aus.

2. Produktübersicht

2.1. Editor Petrinetze

Dieser Editor bietet eine in Eclipse integrierte Benutzeroberfläche, welche aus einer Zeichenfläche und einer Werkzeugpalette besteht.
Die Werkzeugpalette enthält Zeichenwerkzeuge für Stellen, Transitionen und Übergänge zwischen den Stellen und Transitionen, wobei für den jeweiligen Übergang (z.B. Stelle zu Transition) ein eigenes Zeichenwerkzeug existiert.

Der Editor bietet folgende Funktionalität:

- Zeichnen von beliebig vielen Stellen auf der Zeichenfläche.
- Zeichnen von beliebig vielen Transitionen auf der Zeichenfläche.
- Zeichnen von Übergängen von Stellen zu Transitionn.
- Zeichnen von Übergängen von Transitionen zu Stellen.
- Es ist nicht zulässig (und somit auch nicht möglich), Übergänge zwischen zwei Transitionen oder zwischen zwei Stellen zu zeichnen.
- Es ist ebenfalls nicht möglich, von einer Stelle ausgehend mehrere Übergänge zu zeichnen.
- Mehrere abgehende Übergänge von Transitionen sind möglich, aber mehrere eingehende Übergänge bei Stellen sind nicht erlaubt.

2.2. *Editor Rechteck*

Diese beiden Editoren sind im wesentlichen dem Petrinetz-Editor ähnlich, jedoch ist es nur möglich, ein Rechteck auf die Zeichenfläche zu zeichnen. Bei dem ersten der beiden Editoren wird intern eine Rectangle EClass instanziiert.
Bei dem zweiten Editor ist es möglich, die Farbe des Rechtecks nicht nur im Editor-View, sondern auch im Modell zu ändern. Es werden also die Änderungen des Views in des Datenmodell übernommen.

3. Grundsätzliche Struktur und Entwurfsprinzipien

3.1. *Editor-Petrinetze (4.1.)*

Ecore-Modell:
In dem Ecore-Modell sind die einzelnen EClasses zu den Graphischen Objekten des Editors definiert.
So verfügt das Modell über des Package „Petri“, welches sämtliche erforderlichen EClasses umfasst. Der Container, welcher dann alle Elemente des Editors aufnimmt, ist eine EClass namens „PetriNetz“, welche beliebig viele Instanzen der EClass „Form“ enthalten kann.
Die EClass „Form“ stellt eine Superklasse der grafischen Objekte der Zeichenfläche dar und enthält das Attribut „name“ vom Typ EString.
Die beiden grafischen Elemente „Transition“ und „Stelle“ wurden von der EClass „Form“ abgeleitet und können somit dem Container „PetriNetz“ hinzugefügt werden. Des Weiteren verfügt die EClass „Stelle“ über eine EReferenz namens „UebergangNachT“, welche vom Typ Transition ist und die mögliche Verbindung von Stellen zu Transitionen definiert. Stellen können maximal eine Referenz dieses Typs besitzen. Die EClass „Transition“ verfügt über eine EReferenz namens „UebergangNachS“, welche vom Typ Stelle ist und die Verbindungen von Transitionen zu Stellen definiert. Transitionen können beliebig viele Referenzen dieses Typs besitzen.
GMF-Graph:
Das Root-Element der GMF-Graph ist eine Canvas, welche die Zeichenfläche des Editors darstellt. Diese hat ein Child vom Typ FigureGallery namens „Default“, welches die weiteren Kinder (Rectangle namens „TransitionFigureFrame“ oder auch Ellipse namens „StelleFigure“ und die beiden PolylineConnections) enthält.
RechteckTransitionFigure enthält noch weitere Kinder, wie z.B. MaximumSize, MinimumSize oder auch Background und BorderLayout, die die optische Erscheinung des Objekts definieren. Es sind auch zu den anderen Elementen der FigureGallery noch Kinder spezifiziert. Des Weiteren Enthält die Canvas auch noch zwei Knoten, die mit Stelle und Transition der FigureGallery assoziiert sind. Es gibt auch noch Connections,

welche mit den beiden PolylineConnections assoziiert sind und dann sind noch zwei DiagramLabels vorhanden, welche die beiden Labels der Knoten identifizieren.

GMF-Tool:

Die GMF-Tool verfügt über eine Tool-Registry, welche die Tool-Einträge des Editors registriert, also später die Werkzeuge des Editors enthält. Diese enthält ein Kind vom Typ Palette, welche dann wieder die ToolGroup „Petri“ als Kind enthält, welche zur Gruppierung der Werkzeuge dient. Weiter enthält die ToolGroup vier CreationTools zur Erstellung der Stellen, Transitionen und Übergänge.

GMF-Map:

Die GMF-Map verfügt für die Knoten Stelle und Transition über eine TopNodeReference, welche jeweils ein Kind vom Typ NodeMapping enthält, das dann das entsprechende CreationTool „referenziert“. Dieses verfügt noch über ein Kind vom Typ LabelMapping, das das zu dem Knoten gehörende Label identifiziert.
Die beiden LinkMappings gehören zu den Übergängen (Connections) aus der GMF-Graph und stellen für diese die CreationTools bereit. Das CanvasMapping identifizier noch die Zeichenfläche des Editors, die Werkzeugpalette und die Modelle, die dem Ganzen zugrund liegen.

3.2. Editor-Rechteck (4.2.)

GMF-Graph:

Das Root-Element der GMF-Graph ist eine Canvas, welche die Zeichenfläche des Editors darstellt. Diese hat ein Child vom Typ FigureGallery namens „Default“, welches ein Kind vom Typ Rectangle enthält.Dieses stellt das zu zeichnende Rechteck dar und ist mit dem Knoten „Rectangle“ referenziert.

GMF-Tool:

Die GMF-Tool verfügt über eine Tool-Registry, welche die Tool-Einträge des Editors registriert, also später die Werkzeuge des Editors enthält. Diese enthält ein Kind vom Typ Palette, welche die ToolGroup für unser Werkzeug zur Erstellung des Rechtecks enthält. In der ToolGroup befindet sich das CreationTool namens „Rectangle“ zur Erstellung des Rechtecks im Editor.

GMF-Map:

Die Map verfügt wieder über ein TopNodeMapping und ein NodeMapping für den Knoten Rectangle, welches diesen Knoten aus der GMF-Graph mit dem CreationTool aus der GMF-Tool verbindet.
Das CanvasMapping übernimmt wieder dieselbe Aufgabe wie zuvor!

3.3. Editor-Rechteck (4.3.)

Dieser Editor ist in seinem Aufbau dem Editor aus Aufgabenstellung 4.2. gleich, jedoch wurden zur Änderung der Farbe im Semantischen Modell des Editors noch einige Änderungen am Quellcode des Editors vorgenommen.

Im Package org.eclipse.gmf.examples.rechteck.diagramm.part sind sechs neue Klassen (RechteckBlau, ...) hinzugekommen, die bei Auswahl der neuen Farbe des Rechtecks im Kontextmenü die Farbe des View-Objektes und die des Semantischen Modells zu diesem Objekt anpassen. Registriert sind die Klassen in der plugin.xml im extension-point "org.eclipse.ui.popupMenus". Dieser extension-point ist mit dem View-Objekt des Rechtecks verbunden und registriert das Aufrufen eines Unterpunktes des Menüs „Farbe ändern“ im Kontextmenü. Beim Aufruf eines vorgegebenen Farbwertes wird die run-Methode der entsprechenden Klasse gestartet und die Farbe im View und Modell des Rechtecks geändert. Dazu wurde im Paket „org.eclipse.gmf.examples.rechteck.impl“ noch eine Klasse CustomColor deklariert, welche von der Klasse RGBColorImpl abgeleitet ist und einen öffentlichen Konstruktor zur Estellung eines neuen Farbwertes für das semantische Modell des Rechtecks bietet.