УДК 004.4'2

Д.Ю. Филатов, Л.Н. Лядова

РАЗРАБОТКА РЕДАКТОРА ВИЗУАЛЬНЫХ МОДЕЛЕЙ, ОСНОВАННОГО НА Р-ГРАФАХ

Аннотация: Описаны требования к редактору предметно-ориентированных моделей. Приведено определение формальной модели (граф с полюсами) и описание разработанной на её основе двуслойной объектной модели, использованной для разработки редактора визуальных моделей.

Ключевые слова: визуальная модель, граф с полюсами, объектная модель, предметно-ориентированное моделирование.

Введение

На рынке прикладного программного обеспечения существует множество решений для работы с графическими моделями. Различные CASE-системы, DSM-платформы широко используются при разработке программного обеспечения. Системы имитационного моделирования применяются для решения сложных задач исследования реальных систем и процессов. На данный момент активно ведётся разработка визуальной платформы для моделирования Eclipse Modeling Project, мета-CASE-системы MetaEdit+ [1], DSM-платформы QReal [2] и пр. Такая активность характеризует направленный переход от создания моделей с помощью стандартизованных графических нотаций к созданию собственных языков моделирования, на основе которых создаются предметно-ориентированные визуальные модели.

Возможности систем моделирования во многом определяются математическими моделями, на базе которых разработаны редакторов моделей [3-4].

Редакторы графических моделей в визуальном моделировании обычно не используется независимо от CASE-систем или систем имитационного моделирования. Однако предметно-ориентированное моделирование (Domain Specific Modelling, DSM) предполагает возможность настройки системы на особенности предметной области, потребности пользователей, что обеспечивает преимущества этого подхода. Языковые инструментарии, DSM-платформы обеспечивают возможность разработки собственных визуальных нотаций, поэтому они предъявляют специфические требования к работе с графическими моделями: возможность создания и отображения заданных пользователем визуальных нотаций, возможность использования ранее созданных моделей для разработки новых языков (использования моделей как метамоделей), что обеспечивает их переиспользование, пошаговое уточнение в рамках выделенных предметных областей, решаемых в них задач при создании иерархии моделей [5-6].

На основе анализа графических редакторов, используемых графических нотаций, визуальных языков моделирования выделены следующие дополнительные требования к представлению и построению моделей, принятых в DSM-платформе MetaLanguage:

- Создаваемая модель может иметь иерархическую структуру: модель может содержать элементы, каждый из которых имеет сложную организацию, требующую детализации – расшифровки. Таким образом, математическая модель должна обеспечивать выполнение операции декомпозиции.
- Связи между элементами также могут иметь структуру, требующую описания, детализации в модели.
- Элементы модели могут быть связаны друг с другом в заданных точках, расположение которых может иметь значение, определяющее их семантику. Таким образом, при создании модели необходимо обеспечить не только включение в неё элементов и определение связей между ними, но и выделение точек, через которые связываются элементы.

Для построения визуальных языков в DSM-платформах в качестве математической модели используются различные типы графов. Показано [3-4], что использование ориентированного псевдо-метаграфа даёт значительные преимущества по сравнению с другими предлагаемыми типами графов (гиперграфами, *hi*-графами и др.). Однако существует более мощная формальная модель, которая обеспечивает определение на её основе всех рассматриваемых типов графов, – графы с полюсами, или *P*-графы [7]. Предлагаемый формализм (граф с полюсами) обеспечивает реализацию всех перечисленных выше требований.

Понятие графа с полюсами

Граф с полюсами – это упорядоченная тройка $G = (P, V, W)$, где $P = \{p_1, \dots, p_n\}$ – абстрактное множество – множество внешних полюсов графа, $V = \{v_1, \dots, v_n\}$ – непустое множество вершин, $W = \{w_1, \dots, w_n\}$ – множество связей; при этом:

1. Каждая вершина $v \in V$ – это подмножество множества полюсов P и $\forall\, v_i \forall\, v_j \in V\left[i \neq j \;\rightarrow v_i \cap v_j = \emptyset\right]$, т.е. V – множество взаимно не пересекающихся подмножеств полюсов.

2. В каждой вершине $v \in V$ выделяются два подмножества: $I(v)$ и $O(v)$ входных и выходных полюсов: $\forall v \in V\ \exists I(v) \subset P, \exists O(v) \subset P\ [I(v) \cup O(v) = v]$, причём эти множества могут пересекаться, если эти множества не заданы, будем считать, что $I(v) = O(v) = v$.

3. Множество рёбер, описывающих связи между вершинами, определяется как подмножество множества всех пар полюсов, причём $W \subset P \times P \setminus diag(P \times P), \forall v \in V\ \forall p \in v\ \forall r \in v[(p, r) \notin W, (r, p) \notin W]$, т.е. полюс не может быть соединён сам с собой и связи не могут быть установлены между полюсами, принадлежащими одной вершине.

Для графа с полюсами определены операции добавления/удаления полюсов, вершин, рёбер, а также операции расшифровки вершин графами, что позволяет строить иерархические модели сложных систем и процессов, выполняя их пошаговую детализацию. Эти операции могут быть использованы в качестве основы как для разработки графических моделей, так и для их преобразований (трансформаций).

На основе данного определения построена объектная модель графического редактора.

Разработка объектной модели редактора

Основные объекты модели разделены на два слоя:

– описание графа модели – описание *P*-графа, не содержащее информации о визуальном представлении модели, её элементов, их расположения;
– описание представления графа модели в редакторе, с которым работает пользователь – разработчик модели.

Первый слой позволяет выполнять основные операции над элементами иерархической модели, интерпретировать модель, анализировать её свойства, не оперируя избыточной информацией о её визуализации.

Второй слой даёт возможность визуализировать модели при их разработке в конкретном редакторе моделей, в созданной предметно-ориентированной среде.

Диаграмма классов для первого слоя – графа модели – показана на рис. 1. Объектная модель графа с полюсами включает следующие объекты: граф с полюсами (PGraph); вершина (Node); полюс (Pole); ребро (Link); тип ребра (LinkType).

Граф с полюсами обладает следующими свойствами: список внешних полюсов, список вершин, список рёбер. Полюс, вершина и ребро содержат уникальные идентификаторы для поиска их в графе. Для вершины графа определяются только полюсы, однако вершина может быть расшифрована графом; тогда внешние полюсы этого графа должны быть сопоставлены внутренним полюсам этой вершины. Тип вершины используется для удобной их классификации в процессе трансформации и визуализации моделей.

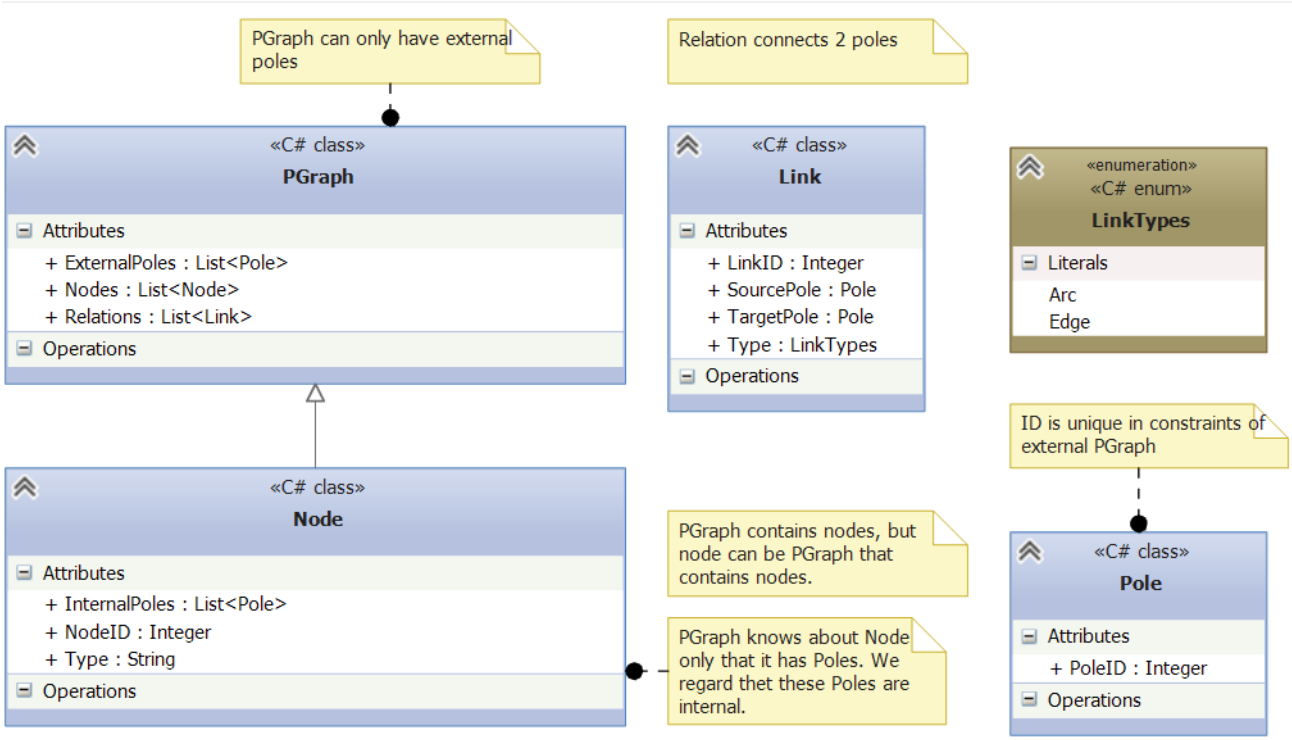

Рис. 1. Упрощённая UML-диаграмма классов графа с полюсами

Чтобы обеспечить иерархию моделей используется наследование от вершины к графу.

Ребро соединяет два полюса, принадлежащие двум разным вершинам в соответствии с приведённым выше определением, следовательно, соответствующий объект должен содержать выходной и входной полюсы и тип ребра (ориентированное (Arc) или неориентированное (Edge)).

При построении визуальной модели систем или процессов будем оперировать терминами сущность, связь и атрибут [3-6].

Под сущностью подразумевается объект предметной области. Чаще всего сущность в графических редакторах изображается в виде простых геометрических фигур, однако может иметь вид произвольной фигуры или изображения для повышения наглядности модели при разработке предметно-ориентированных языков (Domain Specific Language, DSL) и моделей.

Связи показывают отношения между объектами (сущностями), влияние одной сущности на другую и т.п. и изображаются обычно в виде различных типов линий, соответствующих различным видам отношений.

Атрибут обозначает свойство объекта или связи.

На основе описанной модели графа с полюсами строится объектная модель, определяющая представление графа – визуальной модели, её объектов.

При разработке редактора предлагается представлять как сущности не только объекты визуальной модели, но и их отношения, что позволяет единообразно обрабатывать визуальные образы всех элементов модели в редакторе.

Представление модели (View) содержит следующие свойства: порядок отображения элементов; цвет фона модели; состояние, определяющее видимость сетки; высота видимой части модели; ширина видимой части модели; позиция видимой части модели.

Настройки модели включают в себя следующие свойства: название модели; признак, определяющий, могут ли связи пересекаться; признак, определяющий, была ли изменена модель; признак, определяющий, может ли сущность иметь более одной связи; признак, определяющий возможность изменения модели; признак, определяющий возможность изменения подписей элементов; признак, определяющий необходимость каскадного удаления элементов.

Методы, отвечающие за изменение, получение, сохранение и восстановления свойств экземпляров классов, считаются очевидными для реализации и не требуют описания. Комплексные методы, возвращающие нетривиальные результаты включаются в описания классов.

Объектная модель графического редактора показана на рис. 2.

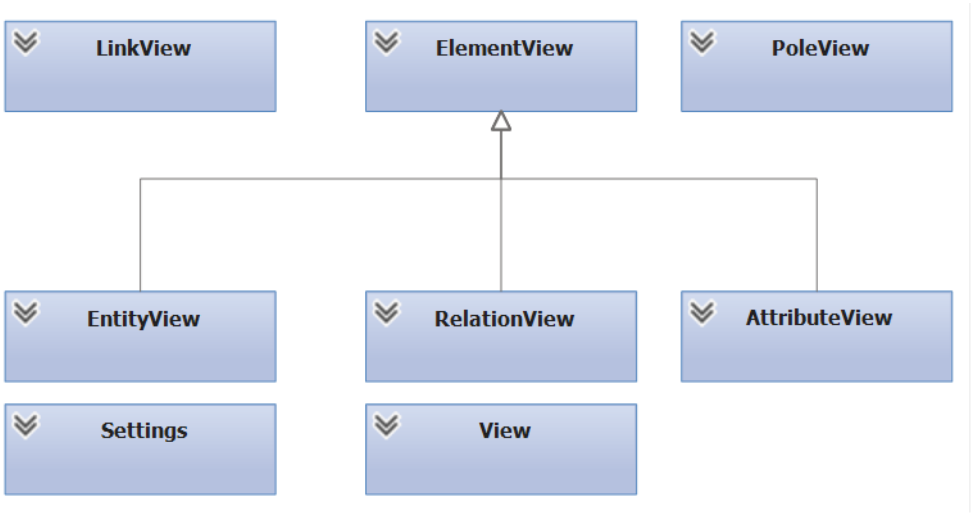

Рис. 2. Упрощённая UML-диаграмма классов основных элементов визуального представления модели

Сущность (EntityView – её представление в модели) может иметь сколько угодно атрибутов также, как и связь.

Вопреки схожести терминов связь (отношение) в области визуального моделирования и ребро в теории графов связь в модели может принимать совершенно разные визуальные представления в модели. По этой причине представление связи (RelationView) наследуется от вершины, а не от ребра. Такое наследование позволяет более гибко настраивать отображение связи в модели, расшифровывать её и т.д. Связь имеет следующие свойства: список атрибутов, форма конца связи и форма начала связи (из-за частого использования различных типов наконечников связей в визуальном моделировании было решено добавить отдельные свойства для них).

Атрибут также может иметь своё представление (AttributeView), включающее соответствующее изображение, позицию относительно соответствующего элемента модели, содержащего атрибут, а также текст, описывающий атрибут.

Представление полюса (PoleView) обозначает точку соединения элементов модели.

Разработка программной модели редактора

За основу при разработке архитектуры приложения – исследовательского прототипа редактора визуальных моделей, принят шаблон проектирования MVC (Model-View-Controller), который включает в себя три основных компонента: модель, представление и контроллер.

Модель (Model) предоставляет данные представлению и реагирует на команды контроллера, изменяя своё состояние. Представление (View) отвечает за отображение данных для пользователя, реагируя на изменения модели. Контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений. Таким образом, MVC позволяет изменять каждый компонент с минимальным изменением связанных компонентов.

В качестве компонента представления для реализации исследовательского прототипа редактора было решено использовать графическую подсистему WPF, так как она входит в состав платформы .Net Framework и позволяет отображать сложные объекты. Представление взаимодействует с контроллером с помощью набора программных интерфейсов DirectX. Для того чтобы представление отображало всегда актуальные данные, реализованы несколько

программных интерфейсов: IUpdate – для модификации данных с помощью контроллера, IModelSubscriber – для подписки об изменении данных и INotify – для оповещения представления об изменении данных.

Заключение

На основе описанных объектных моделей были реализованы базовые функции редактора визуальных моделей. Разработанное приложение – исследовательский прототип редактора – показало практическую значимость предложенной графовой модели и созданной на её основе двухслойной объектной модели.

Реализация исследовательского прототипа редактора моделей показала, что предложенная модель может быть основой для создания языкового инструментария (DSM-платформы), обладающего широкими возможностями для разработки визуальных языков и моделей, их трансформации, обеспечивающего гибкие средства их визуализации для создания удобной предметно-ориентированной среды моделирования.

Библиографический список

1. *Tolvanen J.-P.* Model-Driven Development Challenges and Solutions: Experiences with Domain-Specific Modelling in Industry / *J.-P. Tolvanen, S. Kelly* // Proceedings of MODELSWARD 2016, 4th International Conference on Model-Driven Engineering and Software Development. – Rome: SCITEPRESS. 2016.
2. *Брыксин Т.А.* Платформа для создания специализированных визуальных сред разработки программного обеспечения / *Т.А. Брыксин, А.Н. Терехов* // СПб.: Санкт-Петербургский государственный университет, 2016.
3. *Сухов А.О.* Анализ формализмов описания визуальных языков моделирования / *А.О. Сухов* // Современные проблемы науки и образования. – 2012. – № 2. – С. 1-9.
4. *Сухов А.О.* Теоретические основы разработки DSL-инструментария с использованием графовых грамматик / *А.О. Сухов* // Информатизация и связь. – 2011. – № 3. – С. 35-37.
5. *Лядова Л.Н.* Языковой инструментарий системы MetaLanguage / *Л.Н. Лядова, А.О. Сухов* // Математика программных систем: межвуз. сб. науч. ст. Пермь: Изд-во Перм. гос. ун-та, 2008. – Вып. 5. – С. 40-51.
6. *Сухов А.О.* Среда разработки визуальных предметно-ориентированных языков моделирования / *А.О. Сухов* // Математика программных систем: межвуз. сб. науч. ст. Пермь: Изд-во Перм. гос. ун-та, 2008. – Вып. 5. – С. 84-94.
7. *Миков А.И.* Автоматизация синтеза микропроцессорных управляющих систем / *А.И. Миков* // Иркутск: Изд-во Иркут. Ун-та, 1987.

D.Yu. Filatov, L.N. Lyadova

DEVELOPMENT OF THE VISUAL MODELS EDITOR BASED ON P-GRAPHS

Abstract: Requirements imposed to the domain specific models editor are described. Definition of the formal model (a graph with poles) and the description of the two-layer object model developed on its basis used for development of the visual models editor is given.

Keywords: visual model, graph with poles, object model, domain specific modelling.