МОДЕЛИРОВАНИЕ И ОПТИМИЗАЦИЯ
MODELING AND OPTIMIZATION

УДК 004.4’22

Д. В. Ошкало, аспирант, e-mail: dmitry.oshkalo@gmail.com,
Московский государственный технический университет им. Н. Э. Баумана, г. Москва

Разработка процессов трансформации моделей с помощью биграфов

Предложен подход к разработке процессов трансформации моделей, основанный на использовании биграфов. Показана взаимосвязь между стандартом MOF и представлением моделей в виде биграфов. Для обеспечения корректности правил трансформации применяются ограничения, накладываемые структурой биграфов, и шаблоны правил. Подход позволяет выполнять проверку свойств процессов трансформации, а также осуществлять их моделирование в условиях внесения изменений в модели.

Ключевые слова: *биграфы, моделеориентированная разработка, модели данных, трансформация моделей, правила трансформации, MOF*

Введение

С точки зрения моделеориентированного подхода артефакты процесса разработки программного обеспечения (программный код, конфигурационные файлы, документация и т. д.) представляются в виде моделей — формальных структурированных описаний на одном из языков моделирования, например *UML* [1].

В целях унификации структуры моделей, упорядочения процесса их разработки, обеспечения возможности использования в различных системах был создан стандарт *MOF* (*Meta-Object Facility*) [2], определяющий отношения между моделями и формирующий закрытую многоуровневую архитектуру моделей. Модель каждого уровня задает структурное описание — набор ограничений, накладываемых на модели уровнем ниже: каждый их элемент должен соответствовать одному из элементов модели вышестоящего (т. е. метауровня). В этом случае модель нижестоящего уровня называется экземпляром модели, расположенной уровнем выше.

Архитектура *MOF* состоит из четырех уровней (рис. 1). На верхнем уровне *M*3 находится модель *MOF*, предоставляющая универсальные средства для описания языков моделирования — метамоделей. По этой причине уровень *M*3 называется уровнем мета-метамоделей. Благодаря наличию этого уровня обеспечивается независимость от различных языков моделирования и возможность унификации работы с моделями.

На уровне *M*2 расположены метамодели, определяющие языки моделирования (такие как *UML*), которые используются для описания различных компонентов разрабатываемой системы в виде моделей уровня *M*1. На последнем уровне *M*0 расположены данные или моделируемые объекты реального мира. На рис. 1 в качестве примера приведен стек моделей, используемых при моделировании реляционного хранилища данных. Штриховыми стрелками на рисунке показаны соответствия между элементами различных уровней.

Отметим важное следствие архитектуры *MOF*. Так как все метамодели формируются согласно единой мета-метамодели, можно составить единообразное описание отображений между метамоделями, которое служит основой процедур трансформации моделей [3] — одной из ключевых активностей процесса моделеориентированной разработки, направленной на выполнение таких задач как созда-

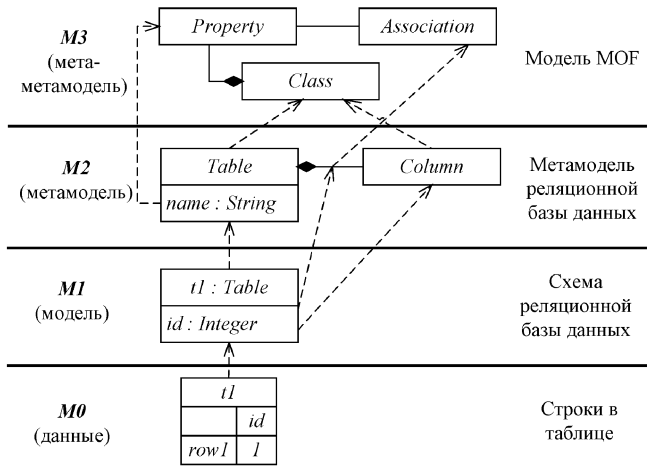


Рис. 1. Уровни архитектуры *MOF*

ние одних моделей на основе других, рефакторинг моделей, перенос изменений между моделями, автоматизированное построение различных компонентов разрабатываемой системы и т. д. Качество выполнения этих задач существенно влияет на качество всего процесса проектирования, а затрачиваемое на их решение время занимает, по разным оценкам, до 30 % времени разработки. Зачастую модели связаны друг с другом: одна из них может быть сгенерирована на основе другой или же обе они могут представлять одну и ту же информацию с разных точек зрения. В этом случае необходимо поддерживать актуальную связь между моделями в условиях внесения в них изменений в процессе разработки.

Для решения данной проблемы применяют движки трансформации моделей, использующие описанный некоторым образом набор правил трансформации и применяющие эти правила к элементам моделей в определенной последовательности (рис. 2). Таким образом, правила трансформации формулируются на уровне *M*2, архитектуры *MOF*, а процесс их применения затрагивает уровень моделей *M*1.

В движке трансформаций описан механизм применения правил к элементам в моделях, поэтому проектирование процессов трансформации сводится к разработке необходимого набора правил для конкретного сценария трансформации моделей.

В настоящее время широкое распространение получили подходы, основанные на графовых грамматиках [4—6], оперирующих моделями в терминах типизированных графов, в которых типы вершин и ребер, а также соотношения между ними регламентируются используемым языком моделирования. Популярность таких подходов обусловлена удобством практического применения, выражаемом в наглядности, возможности использования графических редакторов при проектировании правил трансформации, более прозрачных и эффективных методах контроля и отладки разработанных трансформаций. Примеры программных средств, в которых реализован этот подход, представлены в работе [5]. Выделяют две методики разработки правил [6]:

- создание в редакторе моделей для конкретного сценария трансформации, задание вручную соответствий между их элементами и отладка на тестовых примерах;

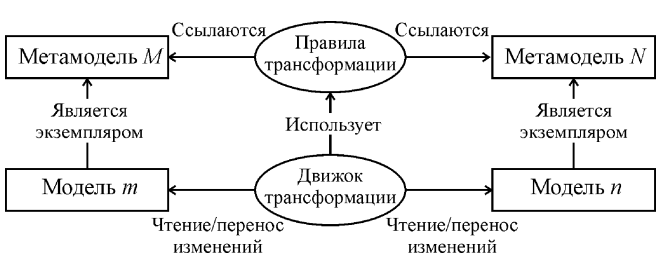


Рис. 2 Основные компоненты процесса трансформации моделей

- использование готовых примеров моделей, на основе которых автоматически будут сгенерированы правила трансформации.

Предлагаемые методики не могут считаться достаточными, поскольку не гарантируют построения корректного набора правил трансформации, не позволяют проверить свойства [3, 7] построенного набора правил и его работоспособность в условиях одновременного внесения изменений несколькими пользователями.

Для создания набора правил, удовлетворяющего этим требованиям, предлагается представить трансформацию моделей в виде системы, состоящей из нескольких параллельно взаимодействующих компонентов. Проектирование таких систем включает этап моделирования, задачей которого является анализ структуры, свойств и поведения как системы целиком, так и отдельных ее частей. Основой для формального описания и анализа в этом случае является аппарат теории процессов. Наиболее важный класс задач, для решения которых предназначена теория процессов, связан с проблемой верификации процессов, которая заключается в построении формального доказательства того, что анализируемый процесс обладает заданными свойствами [8].

Существует множество разновидностей исчислений процессов, в основе которых лежат те или иные аспекты моделируемых систем: обмен сообщениями, работа с поточными данными, мобильность и изменение конфигурации компонентов системы и т. д. Особенностью рассматриваемого случая является использование иерархических структур данных, которыми являются модели, в то время как исчисления процессов оперируют только с простыми представлениями данных. По этой причине для моделирования системы трансформации моделей предлагается использовать теорию биграфов, разработанную Робином Милнером [9—11]. Биграфы (здесь возможна путаница с двудольными графами, но автор, зная об этом, не стал менять названия) и построенные на их основе биграфовые реагирующие системы [10, 12] являются теоретическим базисом для моделирования систем, в которых в процессе взаимодействия между элементами может меняться их расположение и связи между ними. Кроме того, биграфы являются обобщающей концепцией для многих исчислений процессов, таких как π-исчисления и сети Петри [10, 13].

В данной статье рассматриваются аппарат теории биграфов, их статические и динамические компоненты, возможности представления моделей в виде биграфов и использование этого представления при проектировании процессов трансформации моделей на примере трансформации между *UML*-диаграммой классов и соответствующей ей схемой реляционного хранилища данных, а также при моделировании этих процессов и проверке их свойств.

Биграфы

Рассмотрим основные понятия, связанные со структурой биграфов, в том объеме, в котором это необходимо для представления моделей. Отметим, что устоявшегося перевода положений теории биграфов на момент написания статьи нет, поэтому используемые в статье термины приведены вместе с оригинальным написанием. Наиболее полная информация доступна в работах [9—13].

Биграф — это структура, объединяющая в себе такие особенности моделируемой системы, как взаимное расположение ее элементов и связи между ними, представленные в виде двух графов: графа расположения вершин биграфа (*place graph*) и графа связей (*link graph*). На рис. 3 изображены примеры биграфов и их компоненты по отдельности.

Граф расположения (*place graph*) представляет собой лес с именованными корнями и ребрами, описывающими иерархию вершин: вершины, располагающиеся под текущей вершиной в графе и связанные с ней ребром, являются дочерними по отношению к ней (т. е. находятся внутри нее на изображении биграфа). Например, на рис. 3, *а* вершина v_5 на графе расположения ниже вершины v_4, поэтому на биграфе v_5 внутри v_4. Корни графа расположения называют корнями (*roots*), а листья — положениями (*sites*). Первые являются участками, внутри которых находятся вершины биграфа, обозначаются штриховым четырехугольником со скругленными краями; вторые служат в качестве позиций, внутри которых могут быть расположены другие биграфы, обозначаются сплошным четырехугольником с номером.

Граф связей (*link graph*) — это гиперграф, в котором ребра могут связывать произвольное число вершин. Корни графа связей называют внешними именами биграфа, листья — внутренними. Множество имен биграфа образует связи, к которым извне могут быть подсоединены другие биграфы. Таким образом обеспечивается механизм образования новых связей между элементами.

Точки, в которых ребра соединяются с вершинами, называются портами. Каждая из вершин биграфа имеет собственный тип, определяющий число ее портов (например, на рис. 3, *а* вершины v_1, v_3, v_4 имеют тип K с двумя портами). Сигнатурой биграфа называется $\Sigma = (K, ar)$, где K — множество типов, $ar : K \to N$ — функция, для каждого типа устанавливающая число портов.

Для моделирования сложных структур, состоящих из нескольких компонентов, вводится понятие внешних и внутренних интерфейсов, формируемых графами расположения и связей. Та часть интерфейса биграфа на рис. 3, *а*, которая формируется графом расположения, обозначается следующим образом: G^P: 0 → 3, где 0 — количество положений (часть внутреннего интерфейса); 3 — число корней (часть внешнего интерфейса). Для биграфа на рис. 3, *б*: $H^P : 3 \to 2$. Аналогично часть интерфейса биграфов на рис. 3, *а* и 3, *б*, формируемая графом связей, будет иметь следующий вид: $G^L : \varnothing \to \{xy\}$ и $H^L : \{xy\} \to \varnothing$. Сам же биграф при этом характеризуется совокупностью его внутренних и внешних интерфейсов:

$$G = \langle G^P,\ G^L\rangle : \varepsilon \to \langle 3,\ \{xy\}\rangle, \text{ где } \varepsilon = \langle 0,\ \varnothing\rangle;$$

$$H = \langle H^P,\ H^L\rangle : \langle 3,\ \{xy\}\rangle \to \langle 2,\ \varnothing\rangle.$$

Рассмотренных понятий достаточно, чтобы привести формальное определение биграфа. Биграф с сигнатурой Σ представляется как $G = (V, E, P, ctrl, prnt, link) : \langle m, X\rangle \to \langle n, Y\rangle$, где m и n — число положений и корней соответственно; X, Y — множества внутренних и внешних имен; V — множество вершин; E — множество ребер; $P = \{(v, i) | i \in ar(\mathrm{ctrl}(v))\}$ — множество портов, где v — вершина графа; i — номер порта; $\mathrm{ctrl} : V \to K$ — функция, ставящая в соответствие каждой вершине ее тип; $prtn : V \cup m \to V \cup n$ — представление графа расположений; $link : P \cup X \to E \cup Y$ — представление графа связей; $\langle m, X\rangle$ — внутренний интерфейс; $\langle n, Y\rangle$ — внешний интерфейс. Для биграфа, изображенного на рис. 3, *а*:

$$\Sigma = (\{K, L\}, \{(K, 2), (L, 1)\});$$

$$V = \{v_1, v_3, v_4, v_5\} E = \{e_1, e_2\};$$

$$P = \{p(v_1, 1), p(v_1, 2), p(v_3, 1), p(v_3, 2), p(v_4, 1), p(v_4, 2), p(v_5, 1)\};$$

$$ctrl = \{(v_1, K), (v_3, K), (v_4, K), (v_5, L)\};$$

$$prnt = \{(v_1, 0), (v_3, 1), (v_4, 2), (v_5, v_4)\};$$

$$link = \begin{Bmatrix} (p(v_1, 1), x), (p(v_1, 2), e_1), (p(v_3, 1), e_1), \\ (p(v_3, 2), e_2), (p(v_4, 1)e_2), \\ (p(v_4, 2), y), (p(v_5, 1), e_2); \end{Bmatrix}$$

$$m = 0,\ n = 3,\ X = \varnothing,\ Y = \{x, y\}.$$

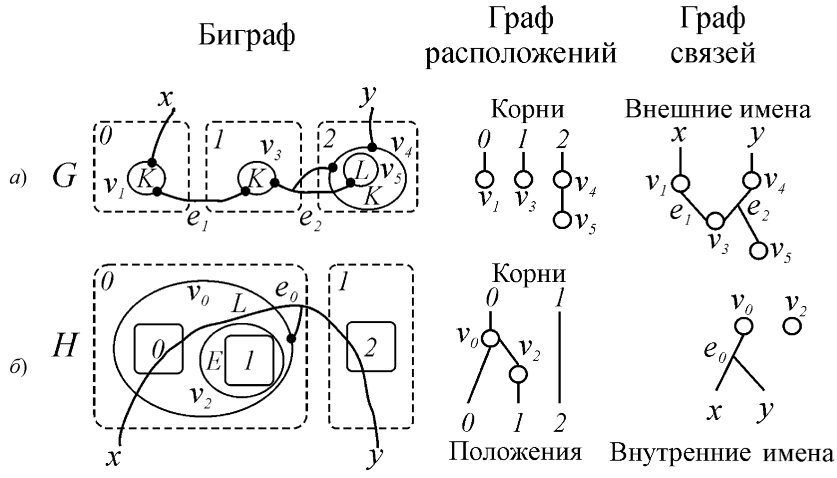


Рис. 3. Биграфы и их компоненты

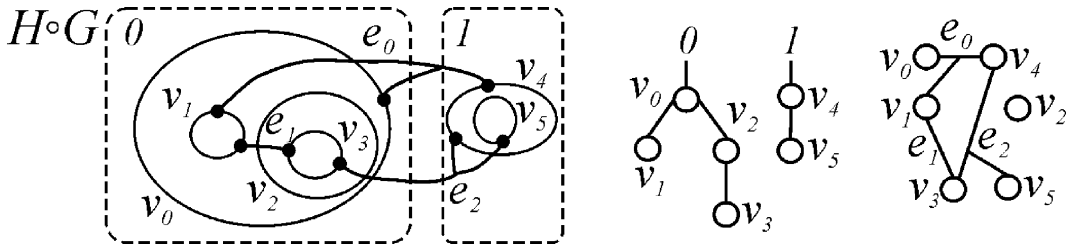


Рис. 4. Результат композиции биграфов

Используя понятие интерфейсов, определим операцию композиции биграфов. Пусть даны два биграфа $G: \langle l, X\rangle \to \langle m, Y\rangle$ и $H: \langle m, Y\rangle \to \langle n, Z\rangle$. Их композиция $H \circ G$ получается путем подстановки корней G со всей внутренней структурой на место соответствующих (по номерам) положений в H, а также соединения связей внешнего интерфейса G с соответствующими (по именам) связями внутреннего интерфейса H. Результат композиции биграфов рис. 3 изображен на рис. 4 (типы вершин не показаны).

Операция композиции биграфов позволяет не только составлять сложные системы из более простых, но и также, используя определение интерфейсов, задавать ограничения, накладываемые на биграфы, описывающие компоненты моделируемой системы.

Представление моделей в виде биграфов

Поскольку трансформация моделей охватывает уровни *M*2 и *M*1 архитектуры *MOF*, для ее описания с помощью биграфов необходимо установить соответствие между биграфами и уровнями *MOF* таким образом, чтобы биграфовое представление моделей и метамоделей сохраняло отношение между этими уровнями, накладываемое *MOF*. Кроме того, чтобы описывать трансформации моделей независимо от используемых языков моделирования, нужно установить связь между аппаратом биграфов и средствами описания языков моделирования, предоставляемыми уровнем *M*3. Для этого достаточно рассмотреть ту часть диаграммы *MOF* [3], которая накладывает ограничения на структуру метамоделей (рис. 5).

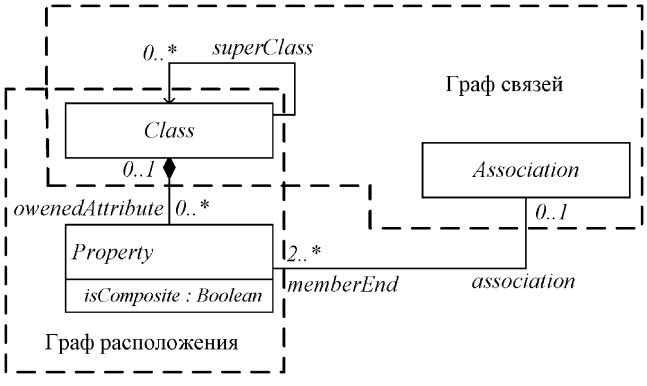


Рис. 5. Связь биграфов с уровнем *M*3 архитектуры *MOF*

Метамодели представляют собой множество элементов *Class*, содержащих набор полей *Property* и связываемых с другими объектами посредством элементов *Property* и *Association*. *Class* и *Property* связаны отношением композиции, в то время как элемент *Association* не может существовать самостоятельно и служит только для представления связей между элементами *Class*. Согласно семантике композиции, связанные объекты находятся в отношении "целое—часть". Такая структура может быть представлена в виде графа расположения, вершинами которого являются элементы *Class* и *Property*.

Возможны три вида отношений между объектами *Class*: наследование с помощью связи *superClass*; композиция, моделируемая объектами *Property* со свойством *is Composite* в значении *true* и объектом *Association*, а также аналогичным образом построенная ассоциация, но свойство *is Composite* объектов *Property* в этом случае имеет значение *false*. Ассоциация может быть *N*-арной. Очевидно, что все перечисленные виды связей между объектами могут быть представлены в виде гиперграфа, вершинами которого являются объекты метамодели, а ребрами — связи между ними.

Таким образом, изображенная на рис. 5 диаграмма может быть полностью описана с помощью компонентов биграфа. Это означает, что биграфы могут выступать в качестве универсального средства метамоделирования. В этом случае, как следует из рис. 1 и 5, существует однозначное соответствие между компонентами биграфа и элементами уровня *M*2 стандарта *MOF*, при котором сигнатура биграфа и множество классов метамодели *MOF* совпадают, отношение композиции, определенное на элементах метамодели, соответствует графу расположений, а ассоциации между элементами — графу связей биграфа метамодели. Благодаря этим соотношениям можно задать как правила перехода от метамодели *MOF* к биграфовому представлению, так и правила обратного перехода от биграфов к *MOF*. Рассмотрим преобразование метамодели *MOF* в ее биграфовое представление на примере метамодели реляционной базы данных на рис. 6 (поля классов не показаны).

Для получения биграфового представления моделей сформулируем заданные метамоделью огра-

ничения в терминах биграфов путем выполнения следующей последовательности действий:

1. Составить сигнатуру биграфов моделей. Множество K типов элементов моделей будет совпадать с множеством классов на диаграмме:

$$K = \{Schema, Table, Column, PrimaryKey, ForeignKey\}.$$

Число портов определяется количеством отношений ассоциации, в которых задействован класс. Так как элементы класса *Column* не могут одновременно быть связаны ассоциацией с *PrimaryKey* и *ForeignKey* (хотя на диаграмме этого ограничения не показано), типу *Column* соответствует один порт. Таким образом, имеем

$$\Sigma_{MM} = \left(\{K\}, \begin{Bmatrix}(Schema, 0), (Table, 0), (Column, 1), (PrimaryKey, 2),\\ (ForeignKey, 2)\end{Bmatrix}\right).$$

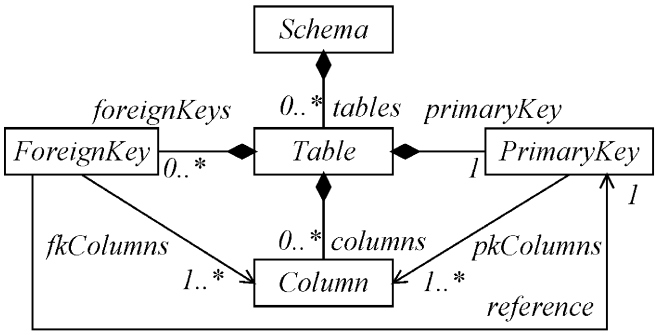


Рис. 6. Метамодель реляционной базы данных

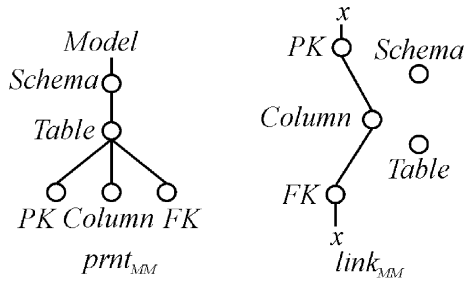


Рис. 7. Представление метамодели реляционной базы данных в виде биграфа

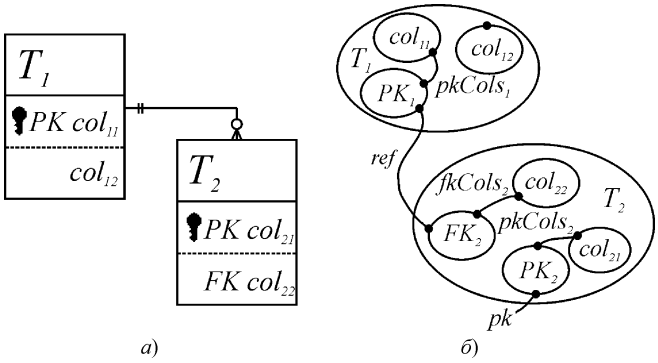


Рис. 8. Представление модели из двух таблиц со связью по внешнему ключу в виде биграфа

2. С учетом отношения композиции составить граф расположения метамодели $prnt_{MM}$, задать значение единственного корня, которым будет элемент, обозначающий модель целиком. Если элементы метамодели связаны отношением композиции, то соответствующие им вершины графа расположения будут соединены ребром.

3. Рассмотреть отношения ассоциации между классами и составить граф связей $link_{MM}$. Так как компоненты биграфов представляют собой неориентированные графы, направление ассоциации в расчет не принимается, но, если ассоциация используется для связи объектов с различным расположением (*ForeignKey* может ссылаться на *PrimaryKey* другой таблицы), на графе связей она разбивается на две связи, причем началу ассоциации (*ForeignKey*) соответствует внутреннее имя, а концу (*PrimaryKey*) — внешнее имя.

Для метамодели на рис. 6 получим графы расположения и связей, изображенные на рис. 7 (сокращения: *FK* — *ForeignKey*, *PK* — *PrimaryKey*).

Биграф модели

$$G_M = (V_M, E_M, P_M, ctrl_M, prnt_M, link_M): \langle m_M, X_M\rangle \to \to \langle n_M, Y_M\rangle$$

должен удовлетворять следующим условиям, накладываемым полученным представлением метамодели.

1. Соблюдение числа портов вершины каждого типа. Для этого необходимо совпадение сигнатур:

$$\Sigma_M = \Sigma_{MM}.$$

2. Корректность расположения вершин в графе расположения модели $prnt_M$: две вершины графа расположения модели соединены ребром, если соединены ребром вершины, соответствующие их типам в графе расположения метамодели:

$$\forall (v, u) \in V_M : (ctrl_M(v), ctrl_M(u)) \in prnt_{MM} \Rightarrow \Rightarrow (v, u) \in prnt_M.$$

3. Корректность связей между элементами: если существует связь между вершинами модели с определенными портами, то в метамодели между типами этих вершин также должна существовать связь с теми же портами:

$$\forall (p \in P, q \in Q, e \in E_M) : (p, e) \in link_M \wedge (q, e) \in \in link_M \Rightarrow \exists (e' \in E_{MM}) : ((ctrl_M(v), i), e') \in \in link_{MM} \wedge ((ctrl_M(u), j), e') \in link_{MM},$$

где

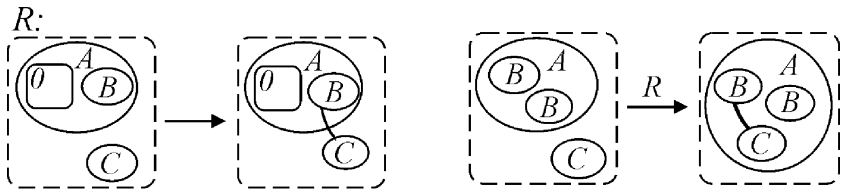
$$P = \{(v, i) | v \in V_M \wedge i \in ar(ctrl_M(v))\};$$

$$Q = \{(u, j) | u \in V_M \wedge j \in ar(ctrl_M(u)) \wedge (u \neq v \vee j \neq i)\}.$$

На рис. 8, *а* изображен пример реляционной модели, состоящей из двух таблиц, имеющих связь по внешнему ключу, а также на рис. 8, *б* показано соответствующее представление этой структуры в виде биграфа.

Моделирование поведения системы

Биграфы в том виде, в котором они рассмотрены выше, служат для описания состояния системы в некоторый момент времени. Для моделирования поведения системы (т. е. ее переходов из одного состояния в другое) используются правила реакции (*reaction rules*), схожие с правилами вывода в графовых грамматиках, но содержащие в левой (*redex*) и правой (*reactum*) частях биграфы. Процесс применения правила осуществляется путем поиска фрагмента, входящего в левую часть правила и замены его фрагментом в правой части.



а) *б*)

Рис. 9. Правило реакции (*а*) и результат его однократного применения (*б*)

На рис. 9, *а* представлено правило реакции R, устанавливающее связь между элементом C и элементом B, вложенным в элемент A, при этом в A возможно наличие других элементов. На рис. 9, *б* показан результат применения этого правила к биграфу, состоящему из элемента C, элемента A и двух вложенных в него элементов B. При этом выбор элемента B, который после применения правила будет соединен с элементом C, является недетерминированным.

В случае с трансформациями моделей набор правил реакции будет в себя включать собственно правила трансформации и возможные изменения, вносимые пользователями.

Биграф с сигнатурой Σ и набор правил реакции образуют биграфовую реагирующую систему (*bigraphical reactive system*) $Bg(\Sigma, \Re)$, где $\Re$ — набор правил реакции вида $R = (r, r')$.

Анализ поведения моделируемой системы осуществляется путем проверки желаемых свойств, выраженных с помощью биграфов. Например, сохраняет ли набор правил реакции определенное взаиморасположение элементов и характер связей между ними, или не приводит ли набор правил к каким-либо нежелательным состояниям системы. В целях более выразительного описания свойств системы можно использовать, например, возможности временной модальной логики. В этом случае каждое свойство системы будет представлено в виде некоторой формулы f. Для проверки свойств необходимо, анализируя набор правил реакции, выделить конечное множество возможных состояний системы и отношение переходов между ними, т. е. преобразовать биграфовую реактивную систему в модель Крипке:

$$Bg(\Sigma, \Re) \to (S, R, L),$$

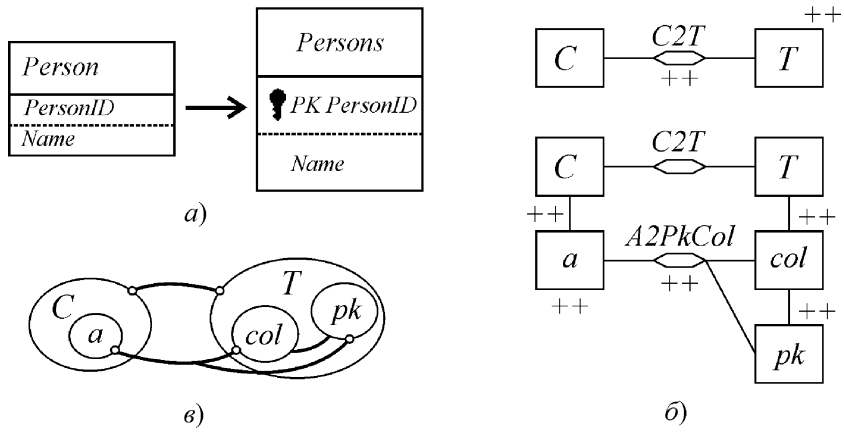
где S — множество состояний системы; $R \subseteq S \times S$ — отношение переходов между состояниями; $L : S \to 2^{AP}$ — функция, ставящая в соответствие каждому состоянию системы подмножество истинных в нем атомарных высказываний множества AP, выраженных в терминах биграфов. Для решения задачи верификации требуется обнаружить множество состояний системы, в которых выполнится f. В качестве метода решения этой задачи возможно использование алгоритмов проверки на моделях [14].

Кроме того, правила реакции позволяют описывать взаимодействие между элементами системы, работающими параллельно. В этом случае благодаря моделированию можно выяснить, корректно ли осуществляется доступ к общему ресурсу, не возникают ли конфликты при работе нескольких пользователей и т. д.

Спецификация правил трансформации

Рассмотрим процесс создания правил трансформации моделей на примере трансформации объектов диаграммы классов *UML* в объекты схемы реляционной базы данных. В этом случае каждый класс трансформируется в таблицу, а его атрибуты — в поля этой таблицы (рис. 10, *а*).

На рис. 10, *б* изображены два правила трансформации в нотации *TGG* (*Triple Graph Grammars*) [6], необходимые для преобразования объектов классов в таблицы. Правила состоят из трех частей: слева и справа расположены элементы моделей (C — класс, T — таблица, a — атрибут класса, col — столбец таблицы, pk — первичный ключ), а посередине —



а) *в*) *б*)

Рис. 10. Трансформация между объектами *UML*-диаграммы классов и реляционной схемы (*а*), правила трансформации в нотации *TGG* (*б*), правило трансформации, составленной с помощью биграфа (*в*)

связующие элементы (*C2T* — связь между классом и таблицей; *A2PkCol* — связь между атрибутом класса и первичным ключом таблицы). Знаком "++" обозначены элементы, которые созданы в результате применения правила. Элементы, не отмеченные этим знаком, должны существовать на момент применения правил, т. е. составляют условие его применения.

Механизм применения правила следующий: в графе модели выполняется поиск элементов из левой части правила. Если они найдены и условие применения правила выполняется, создаются элементы в правой части правила, помеченные знаком "++". Верхнее правило для каждого класса создает таблицу, а нижнее правило предназначено для создания у таблицы, полученной в результате применения первого правила, первичного ключа. Результатом применения каждого из правил является, помимо создания новых элементов, установление связей между элементами исходной модели классов и новыми элементами реляционной модели. Благодаря этой связи обеспечивается корректная последовательность обхода элементов, обработка изменений в моделях (например, при удалении какого-либо элемента удаляются также и все элементы, имеющие с ним связь).

Аналогичный способ описания правил трансформации можно предложить, используя биграфовое представление моделей (рис. 10, *в*). Правило состоит из двух частей, элементы которых связаны друг с другом с помощью гиперграфа. Наличие иерархической связи между элементами позволяет составить всего одно правило вместо двух в нотации *TGG*. В этом случае правило сначала создаст элементы для родительского элемента, а после этого — для дочерних элементов. Для описания связей между элементами каждому типу элементов добавляется дополнительный порт, который будет использован для установления связи с элементами в другой модели (изображены в виде белых кружков на рис. 10, *в*).

Как было отмечено ранее, стандарт *MOF* позволяет единообразно задавать отображения между метамоделями, что нашло применение в описании трансформаций моделей и отражено в стандарте *QVT* (*Query/View/Transformation*) [15]. Анализ связи между положениями данного стандарта и представлением правил трансформации в виде биграфов является довольно обширным и выходит за рамки статьи, но так как предлагаемый подход во многом схож с трансформациями моделей с помощью *TGG*, можно ознакомиться с работой [16].

Используя биграфы, можно создавать шаблоны правил. При составлении правил для каждой структуры в модели, подлежащей трансформации, нужно задать аналогичную структуру в терминах той метамодели, в экземпляр которой она преобразуется. Так как данный процесс осуществляется разработчиком вручную, возможны ошибки, которые могут привести к некорректному результату трансформации. Чтобы избежать такого рода ошибок, предлагается использовать биграфы и определенную на них операцию композиции в целях создания структур, имеющих желаемые свойства и связи с другими объектами, которые можно считать корректными на основании проведенного анализа метамоделей. Поясним данное предложение на конкретном примере.

На рис. 11 изображен шаблон для правила трансформации двух классов, связанных между собой отношением ассоциации, причем до этого момента отработали правила трансформации (аналогичные изображенным на рис. 10), согласно которым каждому из этих классов соответствует таблица, а их атрибутам — столбцы в этой таблице (порты на рисунке не показаны; точки внутри классов — анонимные элементы [13], использующиеся для связей между элементами, которые, в отличие от обычных связей, не приводят к удалению элементов, связанных с удаляемым).

Чтобы обработать элемент *Assoc*, связанный с обоими классами, нужно подобрать структуру в терминах реляционной модели, которая бы корректно отразила функциональное и структурное назначение элемента *Assoc*. Таких структур, причем корректных, может быть несколько в зависимости от кратности отношения. Чтобы гарантировать, что в правиле будет одна из них, создадим в нижней части правила шаблон, повторяющий элемент *Assoc*. Так как каждому классу соответствует таблица, в каждой из них разместим положения (*place*) (*1* и *3* на рис. 11), в которые могут войти структуры, обеспечивающие связь между таблицами. Так как элемент *Assoc* структурно отделен от классов, дополнительное положение (*place*) (*2* на рис. 11) разместим вне таблиц, связав его с положениями

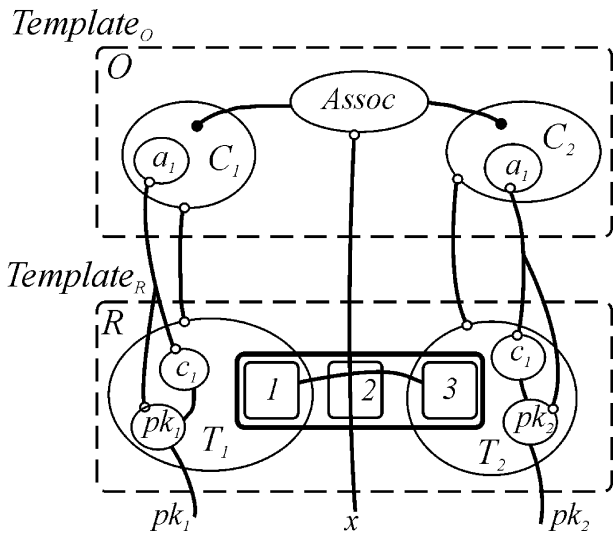


Рис. 11. Шаблон правила трансформации двух классов, связанных отношением ассоциации

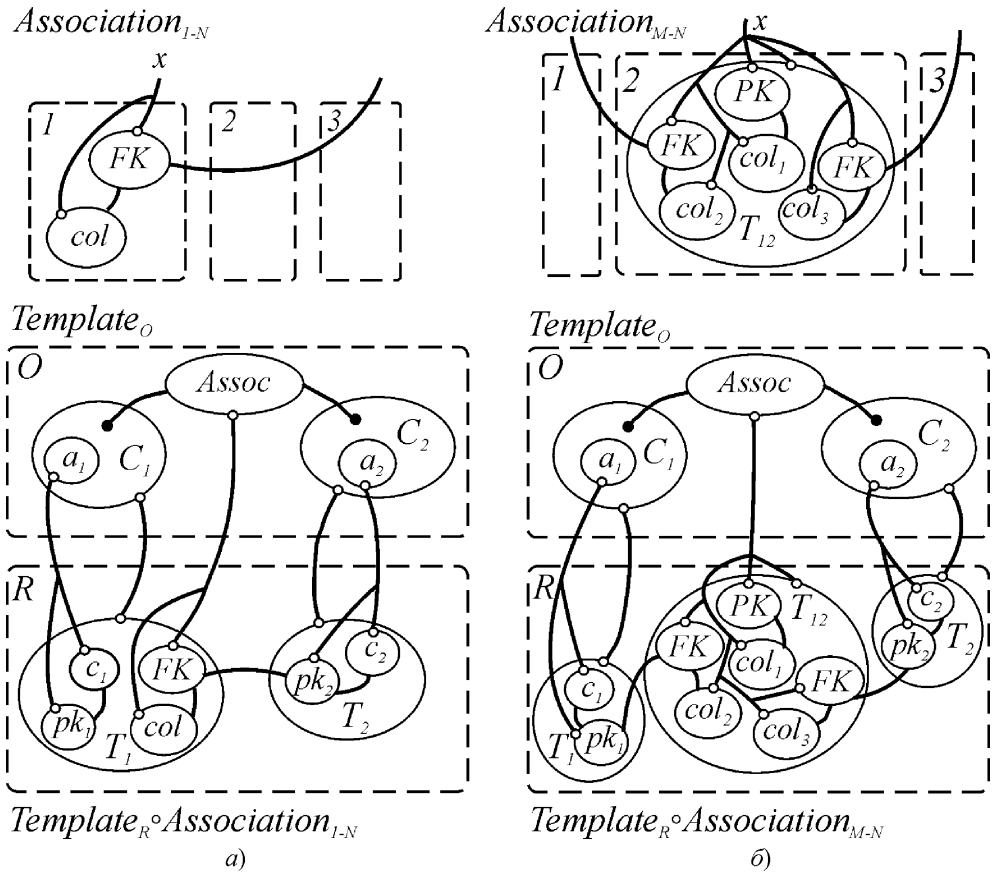


Рис. 12. Структуры, удовлетворяющие шаблону и полученные в результате правила трансформации

(*place*) внутри таблиц, а также с самим элементом *Assoc*. Таким образом, задан формат той структуры, которая в реляционном представлении будет соответствовать элементу *Assoc*. Для обеспечения связи между структурами создано внутреннее имя *x*.

На рис. 12 в верхней части представлены варианты структур, в которые будет трансформирован элемент *Assoc* (рис. 12, *а* — для отношения "один ко многим", 12, *б* — "многие ко многим"). Эти биграфы содержат элементы, которые предоставляют внутренние интерфейсы для соединения с другими элементами извне после выполнения операции композиции. В данном случае образуется связь между внешними и первичными ключами в таблицах (например, *FK* и pk_2 на рис. 12, *а*). Результатом композиции с биграфом-шаблоном, являются правила трансформации моделей, изображенные в нижней части рис. 12.

Таким образом, построенный шаблон задает ограничения для правила и служит дополнительной мерой защиты от ошибок при создании правил трансформации моделей. Заметим, что после операции композиции должна быть выполнена проверка на предмет того, является ли составленная пользователем часть правила (в данном случае реляционное представление) корректной с точки зрения метамодели, которой эта часть определяется.

Заключение

Отличительной особенностью биграфов как средства моделирования является объединение информации о взаимном расположении и связях между элементами рассматриваемой системы. Представление иерархических *UML*-подобных моделей в виде биграфов позволяет задавать ограничения на моделях, которые можно использовать в моделировании процессов трансформации.

Применение биграфов позволяет реализовать такие процедуры, как разработка правил трансформации, верификация процесса трансформации путем задания спецификации процессов в виде набора свойств и их доказательстве методом проверки на моделях. С помощью рассмотренных статических и динамических компонентов биграфов, а также перехода от стандарта *MOF* к биграфовому представлению, можно создать набор программных инструментов для работы с процессами трансформации моделей.

Таким образом, подход к проектированию процессов трансформации моделей, основанный на биграфах, является более формализованным и контролируемым в отличие от применяющихся в настоящее время методов.

Список литературы

1. **Unified** Modeling Language. URL: http://www.uml.org/ (дата обращения: 20.05.2015).
2. **Meta-Object** Facility. URL: http://www.omg.org/mof/ (дата обращения: 20.05.2015).
3. **Biehl M.** Literature Study on Model Transformations: tech. report, July 2010 / Royal Institute of Technology. Stockholm, 2010. 28 p.
4. **Schurr A.** Specification of graph translators with triple graph grammars // Proc. Of the 20th International Workshop on Graph-Theoretic Concepts in Computer Science. Herrsching, Germany, 1994. Vol. 903. P. 151—163.
5. **Hildebrandt S., Lambers L., Giese H., Reike J.** A survey of triple graph grammar tools // Bidirectional transformations. 2013. Vol. 57. P. 1—18.
6. **Kindler E., Wagner R.** Triple Graph Grammars: Concepts, Extensions, Implementations, and Application Scenarios: tech. report, June 2007. University of Padeborn. Padeborn, 2007. 79 p.
7. **Czarnecki K., Helsen S.** Feature-based survey of model transformation approaches // IBM Systems Journal — Model-driven software development. — 2006. Vol. 45. Is. 3. P. 621—645.
8. **Миронов А. М.** Теория процессов. URL: http://intsys.msu.ru/staff/mironov/processes.pdf (дата обращения: 20.05.2015).
9. **Milner R.** Bigraphical reactive systems // Proc. of 12th Conference on Concurrency Theory (CONCUR). Aalborg, 2001. Vol. 2154. P. 16—35.
10. **Milner R.** Bigraphs: a space for interaction. URL: http://www.cl.cam.ac.uk (дата обращения: 20.05.2015).
11. **Milner R.** Bigraphs: a model for mobile agents. URL: http://www.cl.cam.ac.uk (дата обращения: 20.05.2015).
12. **Grohmann D., Miculan M.** Reactive Systems over Directed Bigraphs // Proc. of 18th Conference on Concurrency Theory (CONCUR). Lisbon, Portugal, 2008. Vol. 4703. P. 380—390.
13. **Debois S., Damgaard T.** Bigraphs by Example: tech. report, October 2005. The IT University of Copenhagen. Copenhagen, 2005. 28 p.
14. **Кларк Э. М., Грамберг О., Пелед Д.** Верификация моделей программы: Model Checking / Пер. с англ. Под ред. Р. Смелянского. М.: МЦНМО, 2002. 416 с.
15. **Query/View/Transformation.** URL: http://www.omg.org/spec/QVT/1.1/ (дата обращения 20.05.2015).
16. **Greenyer J., Kindler E.** Reconcilling TGGs with QVT // Proc. of 10th International Conference on Model Driven Engineering Languages and Systems. Nashville, 2007. Vol. 4735. P. 16—30.

D. V. Oshkalo, Postgraduate Student, dmitry.oshkalo@gmail.com
Bauman Moscow State Technical University, Moscow

Bigraph-Based Development of Model Transformation Processes

Model transformation is one of the key activities of model-driven software development, which enables model and code generation, model refactoring, model synchronization, change propagation, etc. It is performed by applying a set of transformation rules to the model elements. The most practically used rule-based approaches (like relational QVT or graphical TGG) provide a formal background for specifying model transformations, but the lack of technique of developing a transformation rules' set for every concrete scenario makes the development of model transformation processes ineffective and error-prone, since there is no possibilities which allows to formulate and verify the properties of the rule set.

A using of bigraphs is proposed as a solution to build a framework for modeling model transformation processes as event-driven systems, which provides an extensible specification, that allows to define concrete model transformation rules. This approach considers models as bigraphs, which are graph-based theoretical tools that emphasize interplay between physical locality and connectivity. Bigraphs are also used as a constraint technique for building transformation rules' templates. This method allows checking the properties of the model transformations and modeling the transformation processes in multiuser environment when simultaneous model changes performed.

Keywords: *bigraphs, model-driven development, data models, model transformation, transformation rules, UML*

References

1. **Umfied** Modeling Language, available at: http://www.uml.org/ (accessed 20.05.2015).
2. **Eclipse** *Modeling Framework*, available at: http://eclipse.org/ modeling/emf/ (accessed 20.05.2015).
3. **Biehl M.** *Literature Study on Model Transformations*, Tech. report, July 2010, Royal Institute of Technology, Stockholm, 2010, 28 p.
4. **Schurr A.** Specification of graph translators with triple graph grammars, *Proc. of the 20th International Workshop on Graph-Theoretic Concepts in Computer Science, Herrsching, Germany*, 1994, vol. 903, pp. 151—163.
5. **Hildebrandt S., Lambers L., Giese H., Reike J.** A survey of triple graph grammar tools, *Bidirectional transformations*, 2013, vol. 57, pp. 1—18.
6. **Kindler E., Wagner R.** *Triple Graph Grammars: Concepts, Extensions, Implementations, and Application Scenarios.* Tech. report, June 2007, University of Padeborn, Padeborn, 2007, 79 p.
7. **Czarnecki K., Helsen S.** Feature-based survey of model transformation approaches, *IBM Systems Journal — Model-driven software development*, 2006, vol. 45, is. 3, pp. 621—645.
8. **Mironov A. M.** *Teorija processov*, available at: http://intsys.msu.ru/staff/mironov/processes.pdf (accessed 20.05.2015).
9. **Milner R.** Bigraphical reactive systems. *Proc. of 12th Conference on Concurrency Theory (CONCUR), Aalborg, Denmark*, 2001, vol. 2154, pp. 16—35.
10. **Milner R.** Bigraphs: a space for interaction, available at: http://www.cl.cam.ac.uk (accessed 20.05.2015).
11. **Milner R.** *Bigraphs: a model for mobile agents*, available at: http://www.cl.cam.ac.uk (accessed 20.05.2015).
12. **Grohmann D., Miculan M.** Reactive Systems over Directed Bigraphs. *Proc. of 18th Conference on Concurrency Theory (CONCUR), Lisbon, Portugal,* 2008, vol. 4703, pp. 380—390.
13. **Debois S., Damgaard T.** *Bigraphs by Example.* Tech. report, October 2005, The IT University of Copenhagen, Copenhagen, 2005, 28 p.
14. **Clarke E. M., Grumberg O., Peled D.** Model Checking, MIT Press Cambridge, MA., 1999, 330 p.
15. **Meta-Object** Facility, available at: http://www.omg.org/mof/ (accessed 20.05.2015).
16. **Greenyer J., Kindler E.** Reconcilling TGGs with QVT // Proc. of 10th International Conference on Model Driven Engineering Languages and Systems. Nashville, 2007. Vol. 4735. P. 16—30.