Л.Н. Лядова, А.П. Серый, А.О. Сухов[1]

Национальный исследовательский университет «Высшая школа экономики» (Пермский филиал)

Пермский государственный национальный исследовательский университет

LyadovaLN@hse.perm.ru, SerAlexandr@bk.ru, Sukhov_PSU@mail.ru

ПОДХОДЫ К ОПИСАНИЮ ВЕРТИКАЛЬНЫХ И ГОРИЗОНТАЛЬНЫХ ТРАНСФОРМАЦИЙ МЕТАМОДЕЛЕЙ

В современных CASE-средствах для решения различных задач (моделирования структур данных, бизнес-процессов и пр.) специалистами-разработчиками (системными аналитиками, проектировщиками баз данных) используются различные языки моделирования.

Кроме того, при создании крупных информационных систем (ИС) возникает необходимость моделирования нескольких предметных областей, для создания моделей которых могут применяться различные языки. В настоящее время для многих областей деятельности ставится задача создания ИС, которые допускали бы участие экспертов (не программистов) как в разработке системы, так и в настройке ее на меняющиеся условия эксплуатации, потребности бизнес-процессов и конкретных пользователей (разработка электронных административных регламентов и пр.). При этом каждому специалисту необходимо обеспечить возможность работы в привычных для него терминах знакомой ему предметной области − ИС становится «мультиязыковой». Предметно-ориентированные языки (Domain Specific Languages, DSL) и языковые инструментарии, предназначенные для их создания, позволяют решить эту задачу, однако остро встает проблема трансформаций построенных моделей при объединении их в одной системе.

Таким образом, проблема трансформации моделей, т.е. переход от описания модели на одном языке моделирования к описанию на другом, стоит достаточно остро при решении задач, связанных с создани-

[1] Работа выполнена при поддержке РФФИ (проект № 12-07-00763-а)

ем ИС [7]. Остается также нерешенным вопрос открытости большинства CASE-средств: бизнес-процессы, описанные в системе, созданной с помощью одного CASE-средства, не могут быть исполнены системой, созданной с помощью другого, из-за того, что они используют различные нотации описания бизнес-процессов. Именно поэтому создание сквозных, т.е. исполняемых в различных системах бизнес-процессов, невозможно [2].

При разработке информационных систем широко используются технологии, основанные на применении *метамоделирования* и *предметно-ориентированных языков* [1]. Система MetaLanguage представляет собой инструментарий для создания визуальных динамически настраиваемых предметно-ориентированных языков моделирования, используемых при разработке информационных систем [8]. Для задания *формальных правил* построения моделей в системе используются *графовые грамматики*. Анализ различных формализмов описания синтаксиса визуальных языков моделирования [4] показал, что наиболее подходящим средством формального задания языков, учитывающим назначение системы MetaLanguage, являются формальные грамматики, представленные в системе ориентированными псевдо-метаграфами. Формальное описание метаязыка системы MetaLanguage представлено в работе [9].

В работе [3] приведён анализ существующих подходов к реализации трансформаций моделей. Наиболее интересные результаты представлены в перечисленных в библиографическом списке публикациях [10-21].

В данной статье рассматриваются реализованные в системе MetaLanguage подходы к реализации вертикальных и горизонтальных трансформаций (мета)моделей.

Трансформации моделей: требования к реализации

Предметно-ориентированные языки (DSL) – это метамодели, применяемые для разработки моделей при создании ИС. При использовании DSL требуется обеспечить как вертикальные, так и горизонтальные трансформации моделей и языков на различных уровнях.

Существуют различные подходы к реализации трансформаций моделей, однако все они имеют определенные ограничения:

− тесная интеграция с конкретными платформами и, как следствие, «наследование» ее ограничений;

− невозможность выбора языка спецификации метамоделей, изменения его описания, поэтому разработчику приходится довольствоваться возможностями, предоставляемыми конкретными языками

(UML и OCL), которые не обеспечивают достаточных выразительных средств для пользователей системы – специалистов в предметных областях;

− невозможность модификации встроенного языка описания трансформаций, хотя некоторые подходы предоставляют в распоряжение пользователя как декларативные, так и императивные языки, однако изменение их описания в соответствии с потребностями конкретной ИС, ее пользователей невозможно;

− возможность лишь «однонаправленной» трансформации, а для получения обратного преобразования необходимо вручную определить обратное отображение;

− большинство существующих систем позволяют выполнять лишь полные эквивалентные отображения, без учёта необходимости сложных преобразований атрибутов, которые часто встречаются при описании бизнес-процессов;

− в большинстве реализаций существующих подходов не затрагивается проблема трансформации ограничений, налагаемых на концепты и отношения модели, а производится лишь преобразование объектов моделей и связей между ними.

Для решения поставленной проблемы должны быть решены следующие задачи:

− Разработка формальной модели и алгоритмов горизонтальных и вертикальных трансформаций метамоделей (языков описания предметных областей), снимающих перечисленные выше ограничения.

− Реализация средств трансформации (мета)моделей в исследовательском прототипе системы трансформации, интегрированном в разработанный ранее DSL-инструментарий, предназначенный для созданий динамически настраиваемых предметно-ориентированных языков.

− Апробация полученных результатов при разработке моделей информационных систем для различных предметных областей.

Для решения поставленных задач предлагается использовать следующие методы и подходы:

− Для описания исходной и целевой моделей планируется использовать *ориентированные типизированные атрибутные метаграфы* [3], что позволяет применять данный инструментарий практически в любой предметной области.

− В качестве формальной основы системы трансформации предлагается использовать *правила перезаписи графов*.

Использование в качестве формальной основы *алгебраического подхода* к трансформации графов позволяет производить парсинг графа, проверять графовые модели на противоречивость.

В языковом инструментарии для обеспечения возможности визу-

ального создания и модификации предметно-ориентированных языков применяются *графовые грамматики*.

Большинство средств моделирования основываются на использовании UML или же являются специализированными для определенных предметных областей. В данном случае предлагается универсальный подход: базовый уровень метаданных рассматривается как графовая модель системы, на которой реализуются основные алгоритмы ее функционирования.

Для создания прототипа CASE-системы, в которую должен быть интегрирован DSL-инструментарий (DSM-платформы), использована *технология DSM* (*Domain Specific Modeling*) *с интерпретацией моделей*. При интерпретации модели применяются методы и алгоритмы, которые в системах с генерацией кода используются для генерации кода по созданной модели.

Над базовым уровнем моделей с помощью DSL-инструментария, базирующегося на графовых грамматиках, создаются метамодели, предметно-ориентированные языки моделирования. Для перехода от одного уровня метамоделей к другому, от одной предметной области системы к другой реализуются трансформации моделей на основе графовых грамматик созданных разработчиками DSL.

Этот подход, обеспечивающий максимальную гибкость системы, не реализован пока в полном объеме ни в одной из существующих промышленных систем, а научные проекты ограничиваются определенными предметными областями. Предлагаемый подход обеспечивает поддержку всего жизненного цикла информационных систем и уникальные возможности их динамической адаптации к различным предметным областям, меняющимся потребностям бизнес-процессов и пользователей.

Вертикальные трансформации графа метамодели

Вертикальная трансформация – это преобразование, при котором исходная и целевая модель принадлежат различным уровням абстракций, например, при отображении объектов метамодели на объекты модели предметной области.

Построение графа модели

Определим отображение графа метамодели на граф модели, которому соответствует *операция создания графа модели*. Такое отображение позволит поддерживать модели в актуальном состоянии, поскольку при модификации метамодели во все модели, созданные на ее основе, будут внесены необходимые изменения.

Введём следующие обозначения:

- $Ent = \{ent_i\}, i \in \mathrm{N}$ – множество вершин графа метамодели, соответствующих сущностям;
- $Rel = \{rel_i\}, i \in \mathrm{N}$ – множество вершин графа метамодели, соответствующих отношениям;
- $EAttr = \bigcup_{i=1}^{|Ent|} EAttr_i$ – множество вершин графа метамодели, соответствующих атрибутам всех сущностей;
- $RAttr = \bigcup_{i=1}^{|Rel|} RAttr_i$ – множество вершин графа метамодели, соответствующих атрибутам всех отношений;
- $EntI = \bigcup_{i=1}^{|Ent|} EntI_i$ – множество вершин графа модели, соответствующих экземплярам всех сущностей метамодели;
- $RelI = \bigcup_{i=1}^{|Rel|} RelI_i$ – множество вершин графа модели, соответствующих экземплярам всех отношений метамодели;
- $EAttrI = \bigcup_{i=1}^{|Ent|} \bigcup_{j_i=1}^{|EntI|} EAttrI_{j_i}$ – множество вершин графа модели, соответствующих значениям атрибутов всех экземпляров сущностей модели;
- $RAttrI = \bigcup_{k=1}^{|Rel|} \bigcup_{l_k=1}^{|RelI|} RAttrI_{l_k}$ – множество вершин графа модели, соответствующих значениям атрибутов всех экземпляров отношений модели.

Множества Ent, Rel, $EAttr$, $RAttr$, $EntI$, $RelI$, $EAttrI$, $RAttrI$ являются конечными в каждый фиксированный момент времени, но при создании/удалении экземпляра сущности, экземпляра отношения или их атрибута множества расширяются/сокращаются.

Построим отображение $fe: Ent \to EntI$, которое для каждой вершины-сущности графа метамодели, определяет множество вершин графа модели, соответствующих экземплярам этой сущности, т.е.

- $(\exists\, ent_i \in Ent)(\exists\, entI_{j_i} \in EntI): fe(ent_i) = entI_{j_i}$, если сущность не является абстрактной и имеет экземпляры;
- $(\exists\, ent_i \in Ent): fe(ent_i) = \varnothing$, если сущность является абстракт-

ной или не имеет экземпляров.

Отображение fe задает *операцию создания вершины, соответствующей экземпляру сущности*.

Построим отображение множества вершин $EAttr$ графа метамодели, соответствующих атрибутам сущностей, на множество вершин графа модели $EAttrI$:

$$fea : EAttr \to EAttrI .$$

Тогда

$$(\forall eattr_{j_i} \in EAttr)(\exists\, eattrI_{k_{j_i}} \in EAttrI) : fea(eattr_{j_i}) = eattrI_{k_{j_i}},$$

$$i = \overline{1, |Ent|},\ j_i = \overline{1, |EntI_i|},\ k_{j_i} = \overline{1, |EAttrI_{j_i}|}.$$

Отображению fea соответствует *операция задания значений атрибутам экземпляра сущности*.

Рассмотрим множество вершин графа метамодели, которые соответствуют отношениям. Каждой такой вершине поставим в соответствие множество вершин графа модели, которое соответствует экземплярам определенного отношения, в результате получим отображение $fr : Rel \to RelI$, такое, что выполняется

- $(\exists\, rel_i \in Rel)(\exists\, reli_{j_i} \in RelI) : fr(rel_i) = reli_{j_i}$, если отношение имеет экземпляры;
- $(\exists\, rel_i \in Rel) : fr(rel_i) = \varnothing$, если отношение экземпляров не имеет.

Данное отображение определяет операцию создания вершины, соответствующей экземпляру отношения.

Определим операцию задания значений атрибутов экземпляра отношения. Для этого построим отображение множества вершин графа метамодели $RAttr$, соответствующих набору атрибутов отношения, на множество вершин графа модели, соответствующих значениям атрибутов $RAttrI$:

$$fra : RAttr \to RAttrI ,$$

причём

$$(\forall rattr_{j_i} \in RAttr)(\exists\, rattrI_{k_{j_i}} \in RAttrI) :$$

$$fra(rattr_{j_i}) = rattrI_{k_{j_i}},\ i = \overline{1, |Rel|},\ j_i = \overline{1, |RelI_i|},\ k_{j_i} = \overline{1, |RAttrI_{j_i}|}.$$

Таким образом, отображения fe, fea, fr, fra задают соответствие между множеством вершин графа метамодели и множеством вершин графа модели (см. рис. 1).

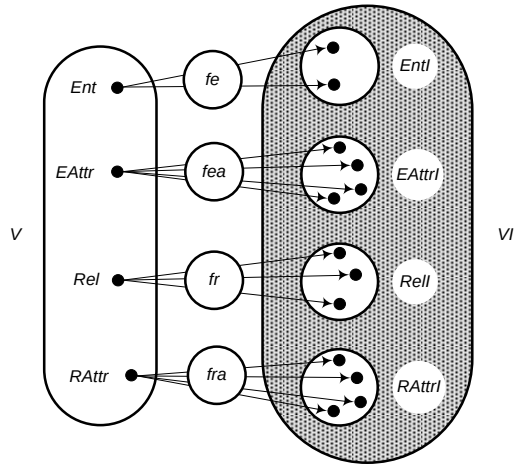

Рис. 1. Отображение множества вершин графа метамодели на множество вершин графа модели

Теперь определим правила, согласно которым дугам графа GMM ставятся в соответствие дуги графа GM .

Построим отображение $gea : EEA \to EEAI$, ставящее каждой дуге множества EEA графа метамодели в соответствие определенные дуги из множества $EEAI$ графа модели, т.е.

$$(\forall eea_{j_i} \in EEA)(\exists\, eeaI_{k_{j_i}} \in EEAI) : gea(eea_{j_i}) = eeaI_{k_{j_i}},$$

$$i = \overline{1, |Ent|},\ j_i = \overline{1, |EntI_i|},\ k_{j_i} = \overline{1, |EAttrI_{j_i}|}.$$

Аналогично можно определить отображение $gra : ERA \to ERAI$, для которого выполняется:

$$(\forall era_{j_i} \in ERA)(\exists\, eraI_{k_{j_i}} \in ERAI) : gra(era_{j_i}) = eraI_{k_{j_i}},$$

$$i = \overline{1, |Rel|},\ j_i = \overline{1, |RelI_i|},\ k_{j_i} = \overline{1, |RAttrI_{j_i}|}.$$

Построим отображение $ger : EERR \to EERRI$, ставящее каждой дуге множества $EERR$ графа метамодели в соответствие определенные дуги из множества $EERRI$ графа модели, т.е.

$$(\forall eerr_i \in EERR)(\exists\, eerrI_{k_{j_i}} \in EERRI) : ger(eerrI_{k_{j_i}}) = eerr_i,$$

$$i = \overline{1, |Ent|},\ j_i = \overline{1, |EntI_i|},\ k_{j_i} = \overline{1, |EAttrI_{j_i}|}.$$

Таким образом, отображения gea, gra, ger задают соответствие

между множеством дуг графа метамодели и множеством дуг графа модели.

Создание графа модели – это отображение графа метамодели на граф модели, при котором выполняются преобразования $fe,\ fea,\ fr,\ fra,\ gea,\ gra,\ ger$.

Интерпретация метамодели

Построим отображение графа модели на граф метамодели. Оно определяет *операцию интерпретации модели*, что позволяет выполнять операции над экземплярами сущностей и проверять ограничения, наложенные на сущности и отношения.

Поскольку вершины графа модели являются экземплярами вершин графа метамодели, то можно задать отображение множества вершин графа GM на множество вершин графа GMM.

Построим сюръекцию $fe^{-1} : EntI \to Ent$, которая каждому экземпляру сущности модели ставит в соответствие сущность метамодели

$$(\forall entI_{j_i} \in EntI)(\exists !ent_i \in Ent):$$

$$fe^{-1}(entI_{j_i}) = ent_i,\ i = \overline{1, |Ent|},\ j_i = \overline{1, |EntI_i|},$$

причем несколько элементов множества $EntI$ могут соответствовать одной сущности, т.е. выполняется

$$(\forall ent_i \in Ent)(\exists\, entI_{j_i}, entI_{k_i} \in EntI, entI_{j_i} \neq entI_{k_i}):$$

$$fe^{-1}(entI_{j_i}) = fe^{-1}(entI_{k_i}) = ent_i.$$

Построим отображение обратное отображению fea:

$$fea^{-1} : EAttrI \to EAttr.$$

Такая сюръекция каждой вершине множества $EAttrI$ ставит в соответствие единственную вершину множества $EAttr$, т.е.

$$(\forall eattrI_{k_{j_i}} \in EAttrI)(\exists !eattr_{j_i} \in EAttr):\ fea^{-1}(eattrI_{k_{j_i}}) = eattr_{j_i},$$

$$i = \overline{1, |Ent|},\ j_i = \overline{1, |EntI_i|}, k_{j_i} = \overline{1, |EAttrI_{j_i}|},$$

причем несколько элементов множества $EAttrI$ могут соответствовать одному элементу множества $EAttr$, т.е. выполняется

$$(\forall eattr_{j_i} \in EAttr)(\exists\, eattrI_{k_{j_i}}, eattrI_{l_{j_i}} \in EAttrI, eattrI_{k_{j_i}} \neq eattrI_{l_{j_i}}):$$

$$fea^{-1}(eattrI_{k_{j_i}}) = fea^{-1}(eattrI_{l_{j_i}}) = eattr_{j_i}.$$

Рассмотрим множество вершин графа модели, которые соответ-

ствуют экземплярам отношений. Каждой такой вершине поставим в соответствие единственную вершину графа метамодели, которая соответствует заданному отношению, в результате получим сюръективное отображение $fr^{-1} : RelI \to Rel$, такое, что выполняется

$$(\forall relI_{j_i} \in RelI)(\exists ! rel_i \in Rel):$$

$$fr^{-1}(relI_{j_i}) = rel_i,\ i = \overline{1, |Rel|},\ j_i = \overline{1, |RelI_i|},$$

причем несколько экземпляров отношения могут быть созданы на основе одного отношения, т.е. выполняется

$$(\forall rel_i \in Rel)(\exists\, relI_{j_i}, relI_{k_i} \in RelI,\ relI_{j_i} \neq relI_{k_i}):$$

$$fr^{-1}(relI_{j_i}) = fr^{-1}(relI_{k_i}) = rel_i.$$

Сюръективное отображение $fra^{-1} : RAttrI \to RAttr$, обратное отображению fra, каждой вершине модели, соответствующей значению атрибута отношения, ставит в соответствие единственную вершину метамодели из множества $RAttr$, причем

$$(\forall rattrI_{k_{ji}} \in RAttrI)(\exists ! rattr_{j_i} \in RAttr):\ fra^{-1}(rattrI_{k_{ji}}) = rattr_{j_i},$$

$$i = \overline{1, |Rel|},\ j_i = \overline{1, |RelI_i|},\ k_{j_i} = \overline{1, |RAttrI_{j_i}|},$$

причем несколько элементов множества $RAttrI$ могут соответствовать одному элементу множества $RAttr$, т.е. выполняется

$$(\forall rattr_{j_i} \in RAttr)(\exists\, rattrI_{k_{ji}}, rattrI_{l_{ji}} \in RAttrI,\ rattrI_{k_{ji}} \neq rattrI_{l_{ji}}):$$

$$fra^{-1}(rattrI_{k_{ji}}) = fra^{-1}(rattrI_{l_{ji}}) = rattr_{j_i}.$$

Таким образом, четыре отображения fe^{-1}, fea^{-1}, fr^{-1}, fra^{-1} задают соответствие между множеством вершин графа модели и множеством вершин графа метамодели (см. рис. 2).

Для перемещения между сущностями, отношениями и их экземплярами расширим множество дуг графа модели дугами-ссылками, соединяющими экземпляры сущностей и отношений с теми сущностями и отношениями метамодели, на основе которых они созданы. Обозначим множество таких дуг через

$$T = \bigcup_{i=1}^{|Ent|+|Rel|} T_i,\ T_i = \{t_{j_i}\},\ j = \overline{1, |EntI_i| + |RelI_i|}.$$

Теперь определим правила, согласно которым дугам графа модели GM ставятся в соответствие дуги графа метамодели GMM.

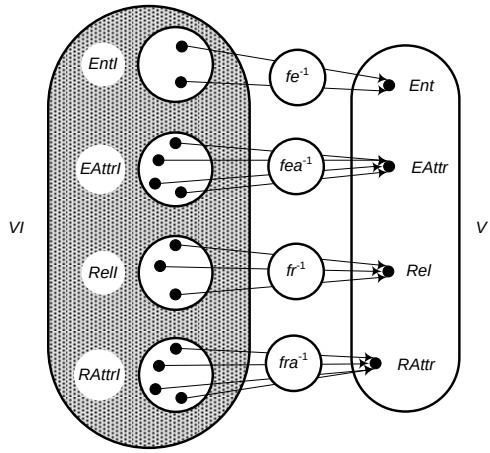

Рис. 2. Отображение множества вершин графа модели на множество вершин графа метамодели

Построим отображение $gea^{-1} : EEAI \to EEA$, ставящее каждой дуге множества $EEAI$ графа модели в соответствие единственную дугу из множества EEA графа метамодели, т.е.

$$(\forall eeaI_{k_{j_i}} \in EEAI)(\exists !eea_{j_i} \in EEA) : gea^{-1}(eeaI_{k_{j_i}}) = eea_{j_i},$$

$$i = \overline{1, |Ent|},\ j_i = \overline{1, |EntI_i|},\ k_{j_i} = \overline{1, |AttrI_{j_i}|}.$$

Аналогично определим отображение $gra^{-1} : ERAI \to ERA$, для которого выполняется:

$$(\forall eraI_{k_{j_i}} \in ERAI)(\exists !era_{j_i} \in ERA) : gra^{-1}(eraI_{k_{j_i}}) = era_{j_i},$$

$$i = \overline{1, |Rel|},\ j_i = \overline{1, |RelI_i|},\ k_{j_i} = \overline{1, |AttrI_{j_i}|}.$$

Как видно из определения, отображения gea^{-1} и gra^{-1} сюръективны.

Построим сюръективное отображение $ger^{-1} : EERRI \to EERR$, ставящее каждой дуге множества $EERRI$ графа модели в соответствие единственную дугу из множества $EERR$ графа метамодели, т.е.

$$(\forall eerrI_{j_i} \in EERRI)(\exists !eerr_i \in EERR) : ger^{-1}(eerrI_{j_i}) = eerr_i,$$

$$i = \overline{1, |EERR|},\ j_i = \overline{1, |EERRI_i|}.$$

Таким образом, отображения $gea^{-1}, gra^{-1}, ger^{-1}$ определяют однозначное соответствие между множеством ребер графа модели и множеством ребер графа метамодели.

Интерпретация модели – это отображение графа модели на граф метамодели, при котором выполняются преобразования $fe^{-1}, fea^{-1}, fr^{-1}, fra^{-1}, gea^{-1}, gra^{-1}, ger^{-1}$.

Горизонтальные трансформации метамоделей

Горизонтальная трансформация – это преобразование, при котором исходная и целевая модели принадлежат одному уровню абстракций. Примером горизонтальной трансформации является рефакторинг [5].

Существуют различные подходы к реализации горизонтальных трансформаций, определяемые связями, создаваемыми между (мета)моделями в системе, реализующей трансформации (рис. 3): явные типизированные связи (explicit typed references), неявные мягкие связи (implicit soft references) и сложные семантические связи (complex semantic connections).

При использовании явных типизированных ссылок связь между моделями становится «монолитной», задаёт жёсткую зависимость между языками, что затрудняет повторное использование построенных предметно-ориентированных языков, созданных с их использованием моделей. Реализация мягких неявных связей основывается на установлении «строкового» соответствия между значениями атрибутов в моделях. Использование моделей и выполнение трансформаций в этом случае возможно только при наличии DSM-платформы, которая «знает» языки и может обрабатывать установленные связи, «разрешать ссылки». Реализация сложных семантических связей обеспечиваем максимум возможностей при создании мультиязыковых систем, в которых создаются и могут изменяться различные DSLs. Однако создание DSM-платформ, обеспечивающих такие возможности, – сложная задача, которую можно решить только на основе соответствующих формальных средств.

Представленные графовые модели, описанные в [6, 8, 9], дают возможность реализации системы горизонтальных трансформаций, удовлетворяющей всем предъявляемым к такой системе требованиям.

Одним из наиболее часто используемых формализмов описания синтаксиса визуальных языков моделирования являются графовые грамматики. Правила вывода системы графовых грамматик могут быть представлены в виде набора пар псевдо-метаграфов, описывающих левую и правую части продукций.

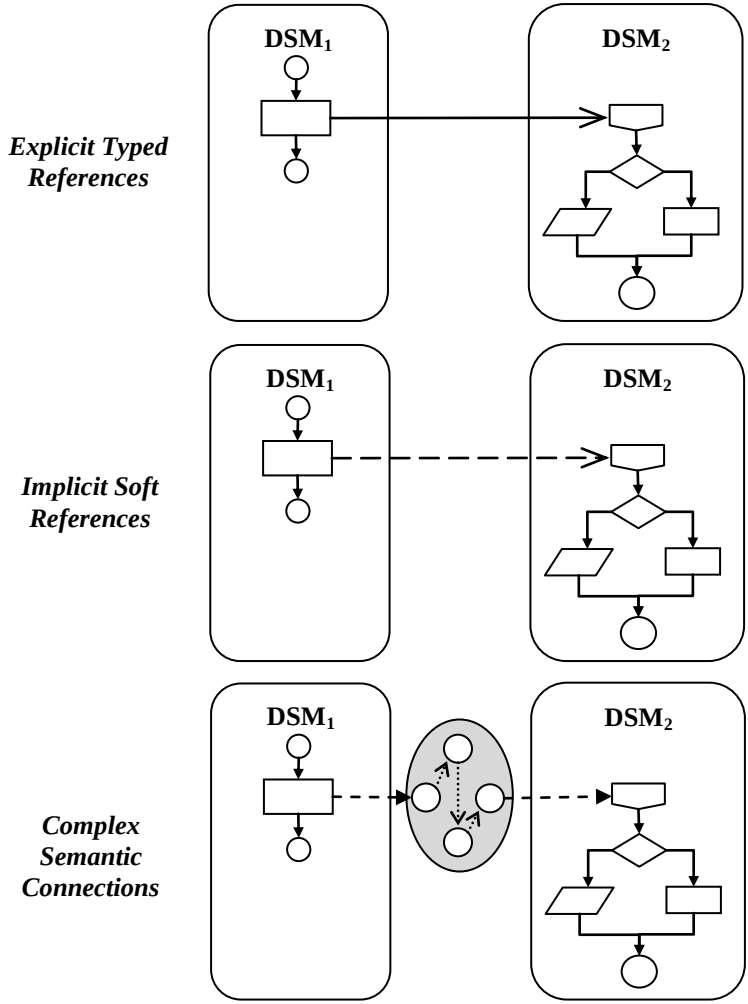

Рис. 3. Типы связей между (мета)моделями

Для внутреннего представления правил грамматики используется один граф, элементы которого содержат информацию о том, к какой части правила они относятся. Для пользователя же строится и визуали-

зируется стандартное представление графовых грамматик, где правила имеют левую и правую части.

Система последовательно проверяет каждое правило в грамматике на возможность его применения. Для этого в хост-графе ищется подграф, изоморфный левой части правила. На рис. 4 показана блок-схема алгоритма исполнения графовой грамматики.

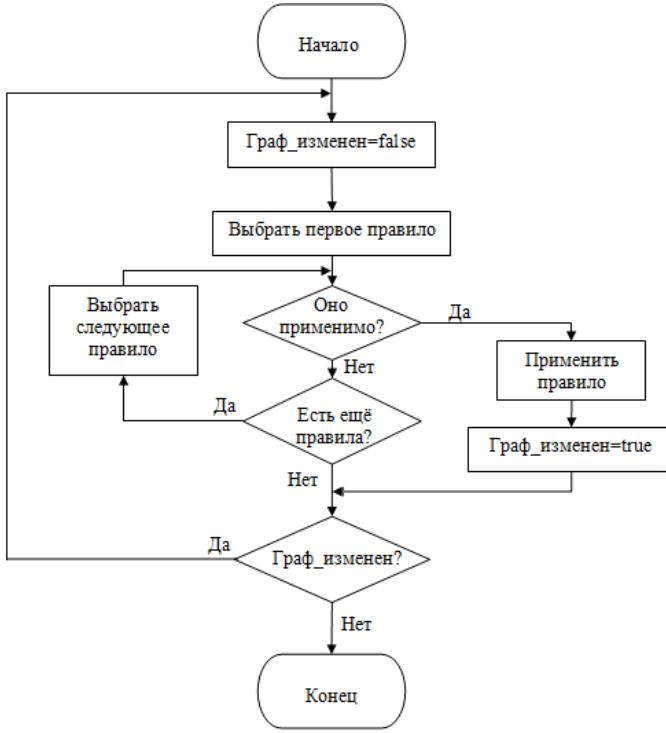

Рис. 4. Исполнение графовой грамматики

Для реализации перебора используется специальный класс *NodeSet*, хранящий набор вершин и их индексы в хост-графе.

В ходе исполнения грамматики система строит набор вершин, содержащий вершины с индексами 0..*M*, где *M* – количество вершин в левой части применяемого правила.

Затем последовательно строятся все перестановки данного набора, и каждая проверяется на соответствие искомому графу.

Если ни одна из перестановок не соответствует искомому графу, то набор вершин *NodeSet* заменяется на следующий по порядку. Для

него выполняется перебор перестановок. Это происходит до тех пор, пока не найдётся подграф, изоморфный левой части правила, либо будет установлено, что подходящей перестановки нет.

Если какое-то правило можно применить к данному хост-графу, оно применяется. Каждая вершина и дуги из левой части правила заменяются соответствующими элементами из правой части согласно правилу. Значения полей новых элементов вычисляются на основе данных начальных вершин и дуг по правилам, указанным в грамматике. Если ни одно из правил применить нельзя, то исполнение грамматики завершено.

При разработке грамматики пользователь может расширить формализмы новыми вершинами и дугами. Такие расширения могут использоваться для снижения трудоёмкости работы по рисованию графов. Так, например, если в табличном формализме, реализуемом в реляционных СУБД, нет связи «*многие-со-многими*», пользователь будет вынужден создавать дополнительную таблицу для реализации каждой такой связи. При использовании созданного инструментария он может создать дугу нового типа на основе имеющихся элементов. Аналогично, если в предметной области часто используется определённая структура, имеет смысл создать новую вершину, которая будет представлять соответствующий подграф. Для этого необходимо описать «расшифровку» новой вершины/дуги с помощью имеющихся элементов формализма, её название и, если создаётся новая вершина, визуальный образ (дуги их не имеют). Этот новый тип элементов будет содержать поля всех элементов соответствующего подграфа (расшифровки). Созданный элемент может быть использован для описания новых типов вершин и дуг. При создании нового элемента пользователь может создать его в специальном окне редактора, а может сначала нарисовать граф в редакторе правил, выделить в нем соответствующий подграф и создать представляющий его элемент.

В перспективе разработанный алгоритм может быть оптимизирован – усовершенствован может быть алгоритм определением применимости правила (сопоставления левой части правила подграфам хост-графа). Возможные подходы к оптимизации рассматриваются в одной из статей настоящего сборника.

Заключение

С каждым годом интерес научного сообщества к проблеме трансформации моделей растет, об этом свидетельствует наличие большого числа публикаций, в которых рассматриваются различные подходы к описанию трансформаций [6].

В статье представлены теоретические основы реализации трансформаций моделей, приведено описание формальных моделей и алгоритмов, реализация которых позволяет снять ограничения, присущие другим подходам. Графовые грамматики позволяют наглядно описать преобразования, которые должны происходить в системе при выполнении заданных в грамматике операций. Реализованные в исследовательском прототипе средства могут быть оптимизированы в ходе дальнейших исследований.

Библиографический список

1. *Лядова Л.Н.* Многоуровневые модели и языки DSL как основа создания интеллектуальных CASE-систем // Сборник трудов третьей международной научно-технической конференции «Инфокоммуникационные технологии в науке, производстве и образовании» (Инфоком-3): Часть 2. – Ставрополь, 2008. – С. 65-71.
2. *Рычков А.Ю.* Управление бизнес-процессами в системах, основанных на метаданных // Вестник Пермского университета. Серия: Математика. Механика. Информатика. – Пермь, 2009. – № 3. – С. 153-156.
3. *Серый А.П.* Методы и средства разработки инструментария для создания языков описания трансформаций предметно-ориентированных языков // Материалы IV Международной студенческой электронной научной конференции «Студенческий научный форум 2012». Мостква, 2012. – [Режим доступа: http://rae.ru/forum2012/pdf/2723.pdf]. – Дата обращения: 20.12.2012.
4. *Сухов А.О.* Анализ формализмов описания визуальных языков моделирования // Современные проблемы науки и образования. – 2012. – № 2. – С. 1-9. URL: www.science-education.ru/102-5655 (дата обращения: 15.10.2012).
5. *Сухов А.О.* Классификация предметно-ориентированных языков и языковых инструментариев // Математика программных систем: межвузовский сборник научных статей / Перм. гос. нац. исслед. ун-т. – Пермь, 2012. – С. 74-83.
6. *Сухов А.О.* Методы трансформации визуальных моделей // Материалы III международной научно-технической конференции "Технологии разработки информационных систем ТРИС-2012". – Геленджик, 2012. – Т. 1. – С. 120-124.
7. *Сухов А.О.* Решение проблем традиционного подхода к разработке информационных систем на основе использования пред-

метно-ориентированных языков // Материалы конференции "Технологии Microsoft в теории и практике программирования". – Томск, 2009. – С. 180-181.
8. *Сухов А.О.* Среда разработки визуальных предметно-ориентированных языков моделирования // Математика программных систем: межвуз. сб. научн. ст. – Пермь: Пермский университет, 2008. – С. 84-94.
9. *Сухов А.О.* Теоретические основы разработки DSL-инструментария с использованием графовых грамматик // Информатизация и связь – 2011. – № 3. – С. 35-37.
10. *Balasubramanian D., Narayanan A., Buskirk C., Karsai G.* The Graph Rewriting and Transformation Language: GReAT: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://www.springerlink.com/content/b1k10p776651765w/ (дата обращения: 27.12.2011).
11. *Balogh A., Varro D.* Advanced Model Transformation Language Constructs in the VIATRA2 Framework: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://static.inf.mit.bme.hu/pub/varro/2006/sac2006_vtcl.pdf (дата обращения: 27.12.2011).
12. *Brambilla M., Fraternali P., Tisi M.* A Transformation Framework to Bridge Domain Specific Languages to MDA: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://home.dei.polimi.it/mbrambil/legacytomda/resources/MDWE_best08.pdf (дата обращения: 27.12.2011).
13. *Csertan G., Huszerl G., Majzik I., Pap Z., Pataricza A., Varro D.* VIATRA - Visual Automated Transformations for Formal Verification and Validation of UML Models (Tool demonstration) : [Электронный ресурс] . Систем. требования: Adobe Acrobat Reader. – URL: http://static.inf.mit.bme.hu/pub/ase2002_varro.pdf (дата обращения: 27.12.2011).
14. *De Lara J., Vangheluwe H., Alfonseca M.* Meta-modelling and graph grammars for multi-paradigm modelling in AToM 3: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://www.springerlink.com/content/atmw5mtfqa33cmh1/ (дата обращения: 27.12.2011).
15. *Grabska E., Strug B.* Applying Cooperating Distributed Graph Grammars in Computer Aided Design: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://www.springerlink.com/content/1710752758t62806/ (дата обращения: 27.12.2011).

16. *Greenyer J., Kindler E.* Reconciling TGGs with QVT: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: https://sosa.ucsd.edu/confluence/download/attachments/8257990/Reconciling%2BTGGs%2Bwith%2BQVT.pdf (дата обращения: 27.12.2011).
17. *Jouault F., Kurtev I.* Transforming Models with ATL: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://wwwhome.cs.utwente.nl/~kurtev/files/Models2005.pdf (дата обращения: 27.12.2011).
18. *Juha-Pekka T., Matti R.* MetaEdit+: defining and using domain-specific modeling languages and code generators: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://portal.acm.org/citation.cfm?id=949365 (дата обращения: 27.12.2011).
19. *Montanari U., Rossi F.* Graph Rewriting, Constraint Solving and Tiles for Coordinating Distributed Systems: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://www.springerlink.com/content/xmm0090271n6r833/ (дата обращения: 27.12.2011).
20. *Stevens P.* Bidirectional Model Transformations in QVT: Semantic Issues and Open Questions: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://homepages.inf.ed.ac.uk/perdita/bidirectional.pdf (дата обращения: 27.12.2011).
21. *Vizhanyo A., Neema S., Shi F., Balasubramanian D., Karsai G.* Improving the Usability of a Graph Transformation Language: [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader. – URL: http://www.isis.vanderbilt.edu/node/3492 (дата обращения: 27.12.2011).