

ПРОГРАММНЫЕ И ЯЗЫКОВЫЕ СРЕДСТВА ДЛЯ СОЗДАНИЯ АДАПТИРУЕМОЙ К КОНКРЕТНОЙ ПРЕДМЕТНОЙ ОБЛАСТИ СИСТЕМЫ ИМИТАЦИИ

Е.Б. Замятина, Л.Н.Лядова, А.О.Сухов

Пермский государственный национальный исследовательский университет, 614990, Пермь, ул. Букирева, 15

Пермский филиал Национального Исследовательского университета «Высшая школа экономики», 614070, Пермь, ул. Студенческая, 38

Рассматриваются программные средства для создания адаптируемой к конкретной предметной области системы имитации. Рассматривается подход, основанный на генерации имитационной модели с помощью языкового инструментария, предназначенного для разработки предметно-ориентированных языков и их трансформаций. Представленный в статье языковой инструментарий позволяет выполнить перевод имитационной модели с одного языка на другой. Трансформации моделей позволяют выполнять полимодельное исследование сложного динамического объекта или системы.

1. Введение

Известно, что имитационное моделирование зачастую является единственным методом исследования сложных динамических систем и объектов и широко применяется в различных сферах науки и производства. Развитие науки и технологий, и, следовательно, рост сложности исследуемых систем и объектов, ставит перед имитационным моделированием все более сложные задачи. Для получения достоверной информации при проведении имитационных экспериментов требуется привлечение специалистов из разных областей знаний, а, следовательно, программные средства имитации должны позволить исследователям работать в различных визуальных средах, пользуясь различными языками и системами понятий. Так, например, при проектировании компьютерных сетей исследователи могут привлечь аппарат теории графов, сетей Петри или систем массового обслуживания. В этом случае необходима настройка на конкретную предметную область, которую можно выполнить, используя онтологии [1] или, как предлагается в настоящей статье – языковой инструментарий, позволяющий разрабатывать предметно-ориентированные языки и осуществить перевод имитационной модели с одного языка на другой[2,3,5,6].

2. Языковой инструментарий MetaLanguage

Языковой инструментарий MetaLanguage ориентирован на решение перечисленных задач [4, 5]. Порядок разработки моделей в системе MetaLanguage показан на рис. 1. На первом этапе разработки DSL в языковом инструментарии MetaLanguage необходимо создать новую метамодель, указав её имя и описание (если это необходимо). Метамодель в данном случае – это предметно-ориентированный язык моделирования, который используется для создания моделей, ориентированных на решение конкретных задач. Затем можно приступить к построению метамодели с помощью графического редактора моделей.

При создании метамодели в первую очередь определяются базовые конструкции языка. Базовыми элементами, которые используются в MetaLanguage для создания метамodelей (DSL), являются сущность, отношение, ограничение. В процессе создания DSL определяются сущности метамодели, отношения между ними, задаются ограничения, налагаемые на сущности и отношения. После построения метамодели

разработчик получает в распоряжение расширяемый, динамически настраиваемый визуальный язык моделирования.

Используя полученный DSL, пользователь может создавать модели, содержащие объекты, описывающие конкретные сущности предметной области и связи между ними. После построения модели необходимо проверить, удовлетворяет ли она ограничениям, которые были на неё наложены, – выполнить валидацию созданной модели. Если какие-либо ограничения не выполняются, пользователь будет проинформирован об этом.



Рис. 1. Порядок разработки моделей в системе MetaLanguage

Разработанный язык может использоваться в качестве метаязыка для разработки новых языков. При внесении изменений в метамодель система автоматически внесёт все необходимые изменения в модели, созданные с помощью этой метамодели. Используя генератор, пользователь может сохранить построенные метамодели и модели в виде XML-файлов, либо сгенерировать на их основе документацию к системе. Трансформатор позволяет в соответствии с заданными правилами трансформаций (вертикальных и горизонтальных), созданными в той же среде, преобразовать модели [3]. Таким образом, разработанная модель может быть переведена на нужный язык и передана во внешние системы для решения соответствующих задач.

Система MetaLanguage является удобным средством для построения как метамodelей, так и созданных на их основе моделей предметных областей. Созданные с помощью языкового инструментария языки моделирования, метамодели и сами модели могут экспортироваться во внешние системы, обеспечивая их интеграцию.

3. Пример разработки и трансформации языка моделирования в системе MetaLanguage

Системы массового обслуживания (СМО) широко используются для анализа характеристик бизнес-систем в различных областях. При этом для исследования используются различные методы и средства (статистический анализ, имитационное моделирование). Рассмотрим пример разработки языка моделирования систем

массового обслуживания в системе MetaLanguage и описание правил трансформации моделей, описанных на этом языке, в язык моделирования GPSS (из-за ограничений объёма публикации приведён упрощённый вариант описания языка и правил трансформации).

Метамодель языка описания СМО содержит следующие сущности (рис. 2.):

- *Генератор* – сущность, отвечающая за генерацию потока заявок на обслуживание в системе (транзактов). Интервалы между поступлениями заявок – случайные величины, имеющие определённое распределение. Данная сущность имеет следующие атрибуты «Название», «Начальная задержка», «Количество транзактов», «Приоритет».
- *Очередь* – сущность, представляющая множество заявок (транзактов), ожидающих обслуживания (освобождения обслуживаемого устройства в случае, если оно занято). Сущность «Очередь» имеет следующие атрибуты: «Название», «Дисциплина», «Максимальная длина», «Текущая длина».
- *Обслуживаемое устройство* – сущность, отвечающая за обслуживание заявок. Устройство обладает ограниченными возможностями обслуживания заявок. Обслуживание требует времени; время обслуживания – случайное число с заданной функцией распределения. Атрибутами данной сущности являются: «Название», «Количество каналов», «Время обслуживания».
- *Размножитель* – сущность, позволяющая создать несколько копий заявки, каждая из которых будет претендовать на обслуживание. Атрибутом данной сущности является «Название», «Количество копий», «Блок» (название блока в модели СМО, на который следует передать копии заявок на обслуживание).
- *Коллектор* – сущность, позволяющая объединить несколько потоков заявок в один поток. Сущность «Коллектор» имеет атрибуты «Название» и «Количество».
- *Терминатор* – сущность, удаляющая заявки из модели.
- *Распределение* – абстрактная сущность, являющаяся родительской для сущностей «Нормальное распределение», «Равномерное распределение», «Распределение Стьюдента» и др.
- *Нормальное распределение* – распределение, в соответствии с которым выполняется генерация новых заявок и/или их обслуживание. Данная сущность имеет два атрибута «Математическое ожидание», «Дисперсия».
- *Равномерное распределение* – распределение, в соответствии с которым выполняется генерация новых заявок и/или их обслуживание. Данная сущность имеет два атрибута «Левая граница интервала», «Правая граница интервала».
- *Распределение Стьюдента* – распределение, в соответствии с которым выполняется генерация новых заявок и/или их обслуживание. Данная сущность имеет атрибут «Количество степеней свободы».

Далее опишем отношения между сущностями метамодели. Как видно из рис. 2., метамодель содержит следующие отношения ассоциации:

- Однонаправленное отношение ассоциации «Создает транзакты», соединяющее сущности «Генератор» и «Очередь», показывает, что после создания заявок они помещаются в очередь на обслуживание.
- Двухнаправленное отношение ассоциации «Обслуживает транзакты» позволяет указать, каким обслуживаемым устройством обрабатываются заявки, находящиеся в очереди, и куда они направляются после обслуживания.
- Однонаправленное отношение ассоциации «Передаёт транзакты», соединяющее сущности «Обслуживаемое устройство» и «Размножитель», позволяет разделить заявки на несколько потоков.

- Однонаправленное отношение ассоциации «Объединяет потоки» соединяет сущности «Обслуживающее устройство» и «Коллектор» и указывает, какой коллектор объединяет потоки заявок на обслуживание после их обработки несколькими обслуживающими устройствами.
- Однонаправленное отношение ассоциации «Разделяет транзакты», соединяющее сущности «Размножитель» и «Очередь», позволяет указать в какие очереди должны быть помещены заявки после их разделения на несколько потоков.
- Однонаправленное отношение ассоциации «Удаляет транзакты», соединяющее сущности «Обслуживающее устройство» и «Коллектор» с сущностью «Терминатор», позволяет указать, что после обслуживания или объединения заявки должны быть удалены.

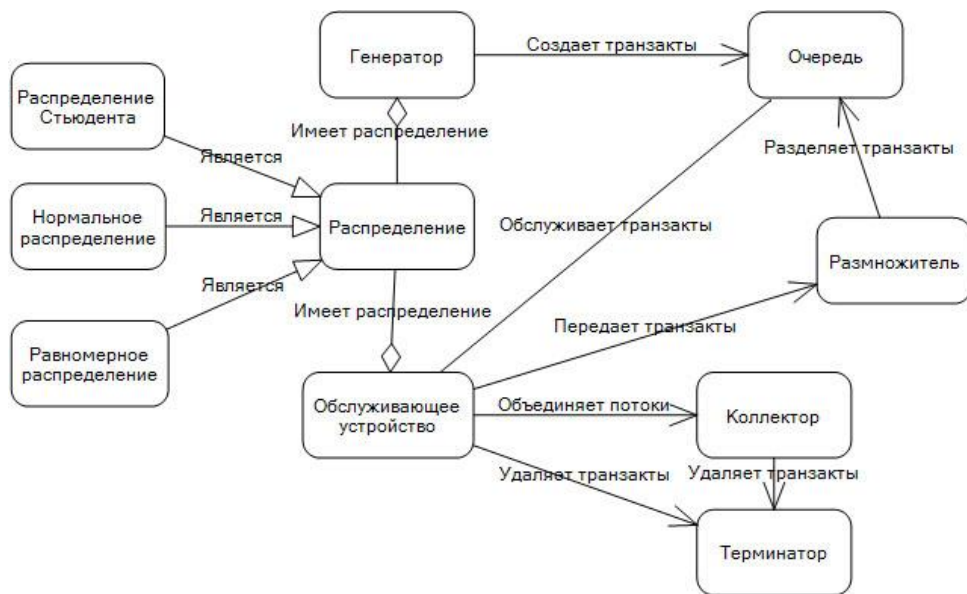


Рис. 2. Метамоделю языка описания имитационных моделей

Метамоделю языка построения моделей СМО также содержит три отношения наследования «Является», которые соединяют абстрактную сущность «Распределение» с дочерними сущностями «Нормальное распределение», «Равномерное распределение», «Распределение Стьюдента». Дочерние сущности наследуют все отношения сущности-родителя. Агрегация «Имеет распределение» позволяет задать распределение, в соответствии с которым выполняется генерация новых заявок (транзактов) и/или их обслуживание.

На рис. 3 изображена модель, построенная с использованием разработанного языка описания СМО. Как видно из рисунка, модель содержит генератор, четыре обслуживающих устройства (PD1, PD2, PD3, PD4), четыре очереди (QQ1, QQ2, QQ3, QQ4), размножитель, коллектор и терминатор; используется два распределения. Опишем правила преобразования построенной метамоделю языка описания СМО в нотации языка GPSS.

Применение этих правил к построенной модели позволит сгенерировать программу на языке GPSS и выполнить анализ модели с использованием данной системы имитационного моделирования.

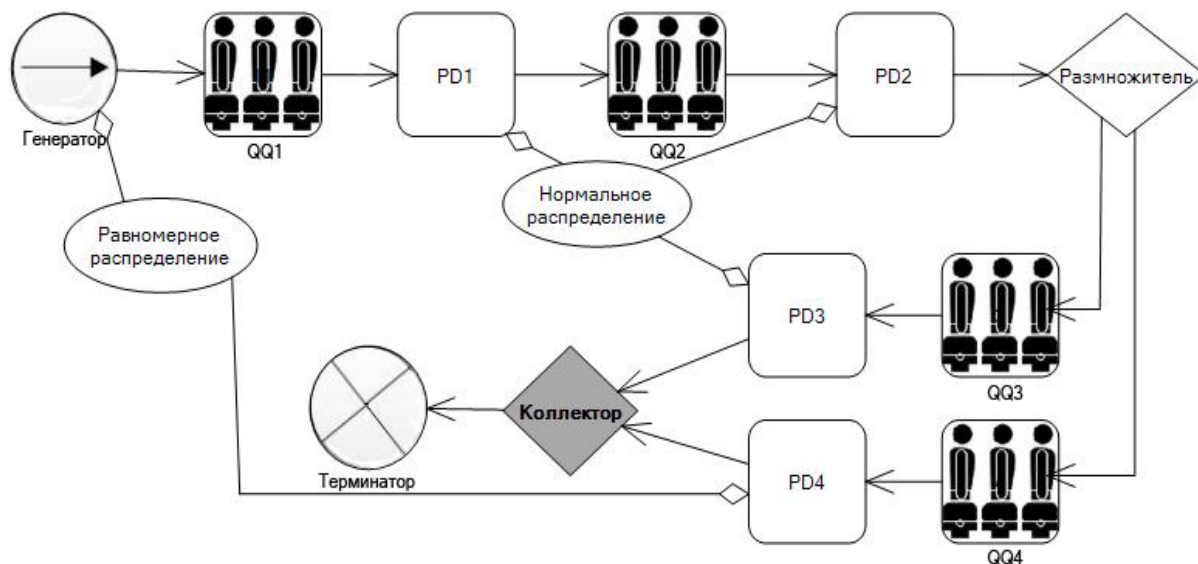
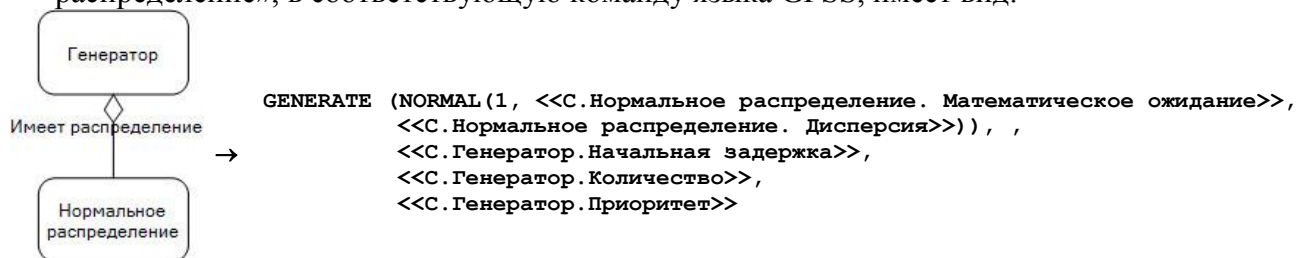


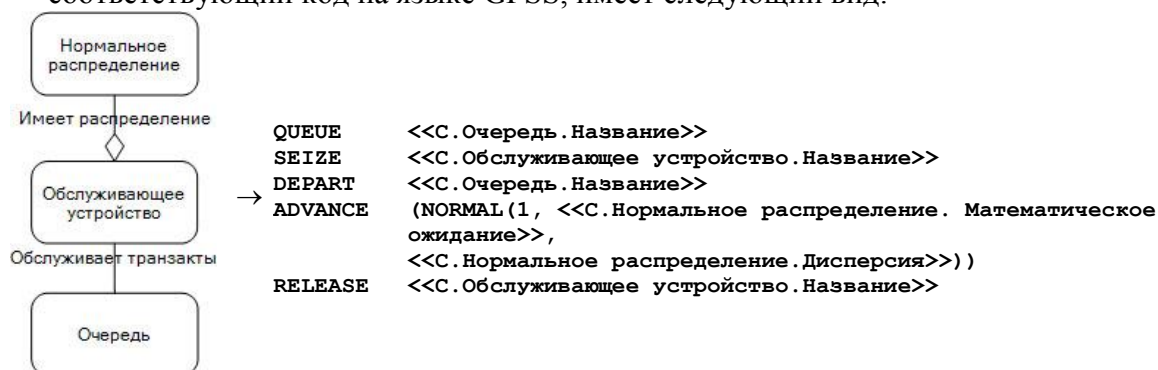
Рис. 3. Пример модели СМО

Правило «Генератор_Норм», преобразующее экземпляр сущности «Генератор», связанный экземпляром отношения агрегации с экземпляром сущности «Нормальное распределение», в соответствующую команду языка GPSS, имеет вид:

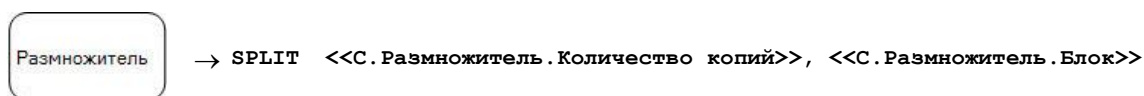


Аналогичным образом описываются правила для других видов распределений.

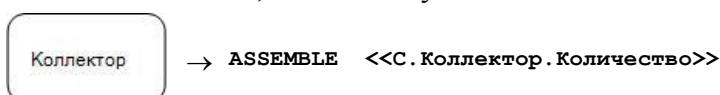
Правило «Очередь», преобразующее связанные экземпляры сущностей «Очередь», «Обслуживающее устройство», «Нормальное распределение» в соответствующий код на языке GPSS, имеет следующий вид:



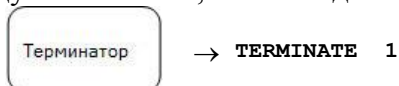
Правило «Размножитель» преобразует экземпляр сущности «Размножитель» в команду SPLIT языка GPSS. Данное правило имеет вид:



Правило «Коллектор», преобразующее экземпляр сущности «Коллектор» в команду ASSEMBLE языка GPSS, имеет следующий вид:



Правило «Терминатор», преобразующее экземпляр сущности «Терминатор» в соответствующую команду языка GPSS, имеет вид:



Использование языкового инструментария позволяет оперативно создавать не только предметно-ориентированные языки, но и определять необходимые трансформации, что обеспечивает интероперабельность созданных моделей и метамоделей, возможность их импорта и экспорта.

4. Выводы

Представленный языковой инструментарий является достаточно удобным и гибким инструментом разработки языков моделирования и правил преобразования моделей, созданных с использованием этих языков. Исследовательский прототип системы MetaLanguage был использован для разработки нескольких предметно-ориентированных языков. Использование системы в учебном процессе показало, что инструментарий легко осваивается пользователями.

В настоящее время актуальной является задача оптимизации алгоритмов трансформации, расширения возможностей учёта семантики при описании трансформаций.

Благодарности

Работа выполнена при поддержке РФФИ (проект № 12-07-00763-а), РФФИ №12-07-00302_а и Министерства образования и науки (проект № 8.5782.2011).

Список литературы

1. Замятина Е.Б. Лингвистические и интеллектуальные инструментальные средства симулятора компьютерных сетей TRIADNS / Замятина Е.Б., Миков А.И., Михеев Р.А.. International Journal “Information theories & Applications (IJ ITA). Vol 19, Number 4, 2012, pp.355-368. ITNEA, Sofia, 1000, P.O.B. 775, Bulgaria

2. Лядова Л.Н. Многоуровневые модели и языки DSL как основа создания интеллектуальных CASE-систем / Л.Н. Лядова // Труды международных научно-технических конференций «Интеллектуальные системы» (AIS'08) и «Интеллектуальные САПР» (CAD-2008). Научное издание в 4-х томах. Т. 2. – М.: Физматлит, 2008. С. 37-41.

3. Лядова Л.Н. Подходы к описанию вертикальных и горизонтальных трансформаций метамоделей / Л.Н. Лядова, А.П. Серый, А.О. Сухов // Математика программных систем: межвуз. сб. науч. ст. – Пермь: Изд-во Перм. гос. нац. исслед. ун-та, 2012. – Вып. 9. – С. 33-49.

4. Лядова Л.Н. Визуальные языки и языковые инструментарии: методы и средства реализации / Л.Н. Лядова, А.О. Сухов // Труды межд. научно-технической конференции «Интеллектуальные системы» (AIS'10). – М.: Физматлит, 2010. – Т. 1. – С. 374-382.

5. Сухов А.О. Инструментальные средства создания визуальных предметно-ориентированных языков моделирования / А.О. Сухов // *Фундаментальные исследования*. – 2013. – № 4 (ч. 4). – С. 848-852.

6. Сухов А.О. Интеграция систем имитационного моделирования и предметно-ориентированных языков описания бизнес-процессов / А.О. Сухов // *Математика программных систем: межвуз. сб. науч. ст. / Перм. гос. ун-т.* – Пермь, 2009. С. 12-23.