УДК 004.43+004.4'236

Е.Б. Замятина, Л.Н. Лядова, А.О. Сухов

О ПОДХОДЕ К ИНТЕГРАЦИИ СИСТЕМ МОДЕЛИРОВАНИЯ И ИНФОРМАЦИОННЫХ СИСТЕМ НА ОСНОВЕ DSM-ПЛАТФОРМЫ METALANGUAGE

Аннотация: Предлагается подход к интеграции систем моделирования и информационных систем на основе DSM-платформы MetaLanguage, позволяющей создавать языки моделирования и модели предметных областей и определять их трансформации.

Abstract: An approach to integration of information systems and modeling systems is suggested. It is based on the DSM-platform MetaLanguage allowing to create modeling languages and domain models and to define model transformations.

Ключевые слова: моделирование, предметно-ориентированные языки, языковой инструментарий, языковые трансформации.
Keywords: modeling, domain-specific languages, language workbench, language transformations.

Введение

Любая бизнес-система представляет собой сложную хозяйственную, организационную, социальную систему, для выживания которой в современных условиях необходимо не только создать эффективную систему управления, но и обеспечить возможность ее реорганизации, оперативной адаптации к изменениям внешней среды. Решение этих задач невозможно без применения современных концепций управления, в полной мере использующих возможности современных информационных технологий (ИТ). Управление реализуется на основе создаваемых аналитиками моделей бизнес-систем, которые служат основой для разработки информационных систем (ИС) различного назначения, средств автоматизации учета и планирования, систем управления документооборотом, бизнес-процессами и т.п., а также для решения задач анализа и прогнозирования. На крупных предприятиях создаются мощные корпоративные информационные системы (КИС), интегрирующие различные средства автоматизации бизнес-процессов, поддержки управления, которые должны обеспечить процесс превращения данных, накопленных в системе, в информацию, в знания о бизнес-системе, позволяющие реализовать эффективную поддержку принятия решений. Небольшие предприятия часто используют готовые решения, «коробочные» продукты, ориентированные на решение различных задач.

Рассматриваемые системы используют различные типы моделей. В основном для их создания применяются графические нотации, диаграммы различных типов, позволяющие описать объекты моделируемой бизнес-системы и их свойства, связи между ними, выполняемые над ними операции и бизнес-процессы и пр. Существенным является требование обеспечения возможности работы с моделями различных категорий пользователей (бизнес-аналитиков, специалистов в конкретных предметных областях), создание средств, которые позволяли бы им без привлечения программистов создавать и изменять модели в зависимости от своих потребностей. Важным условием снижения трудоёмкости работы пользователей является также возможность интеграции различных ИС, «переиспользования» созданных моделей, передачи их из одной системы в другую для решения различных задач.

Для разработки моделей используются различные языки моделирования. При этом выбор инструментальных средств зачастую определяет и выбор языка описания бизнес-процессов. Таким образом, используемый инструментарий фактически «навязывает» разработчикам и пользователям определённый язык моделирования, который чаще всего не позволяет экспертам, специалистам в конкретных предметных областях участвовать в

разработке и модификации моделей, что является необходимым условием создания эффективных систем управления, повышения оперативности их адаптации, снижения трудоёмкости сопровождения.

Задачи интеграции ИС, а также моделей, разработанных на разных языках и, возможно, с помощью различных инструментальных средств, решаются при реализации многих проектов [9, 10, 13]. Один из наиболее часто используемых подходов основан на разработке и применении метамоделей, онтологий предметных областей [9, 10, 12]. Максимальная гибкость разрабатываемых средств достигается при реализации многоуровневых моделей, описывающих системы с разной степенью детализации и с разных точек зрения [2].

Перечисленные требования могут быть реализованы на основе создания средств предметно-ориентированного моделирования (Domain Specific Modeling, DSM), DSM-платформы, основное назначение которой – разработка высокоуровневых предметно-ориентированных языков (Domain Specific Languages, DSLs), предназначенных для создания моделей бизнес-систем, ориентированных на решение задач в различных предметных областях. Языковой инструментарий может стать основой для интеграции различных инструментальных средств, предназначенных как для разработки ИС на основе созданных моделей (CASE-средств), так и для их анализа (в частности, систем имитационного моделирования) [1, 6].

Интеграция систем на основе DSM-платформы

В разных системах, применяемых на предприятиях, используются различные модели, методы представления и обработки данных, ориентированные на решение соответствующих задач. При обмене данными, при передаче их из одной системы в другую выполняется преобразование данных. Методы и средства решения этих задач проработаны и реализуются во всех промышленных системах управления данными. Однако более полная интеграция систем требует не только обмена данными, но и передачи созданных моделей из одной системы в другую. Модель, созданная с использованием какого-либо языка моделирования в одной системе, ориентированной на решение некоторых задач, может быть передана в другую систему для решения других задач. При разработке информационных систем различного назначения применяются разные типы моделей и языки моделирования. Для интеграции систем необходимо учитывать особенности используемых в них языков и моделей. При этом может потребоваться перевод модели с одного языка моделирования на другой. Оперативность решения задач пользователей зависит от того, смогут ли они самостоятельно настраивать используемые средства, создавать и модифицировать модели, задавать правила их преобразования.

Таким образом, DSM-платформа, обеспечивающая возможность создания новых языков моделирования и правил трансформаций созданных языков и моделей, становится центральным звеном, обеспечивающим интеграцию различных систем (рис. 1).

В рамках поставленной проблемы, должен быть решён комплекс задач по созданию DSM-платформы, удовлетворяющей следующим требованиям:

- универсальность – возможность построения языков моделирования для широкого спектра предметных областей и решаемых задач;
- возможность многоуровневого и мультиязыкового моделирования, наличие средств вертикальных и горизонтальных трансформаций моделей, возможность экспорта и импорта построенных моделей;
- возможность изменения конструкций метаязыка и динамического изменения описаний языков моделирования без повторной генерации исходного кода редактора DSL;
- автоматическое поддержание в согласованном состоянии описания метамоделей и моделей предметной области при внесении изменений в метаязык и/или метамодель;

- единообразие средств представления, описания и использования как моделей, так и метамоделей: построение моделей различных уровней иерархии и работа с ними должна производиться с помощью одних и тех же программных средств;
- доступность работы с системой для различных категорий пользователей: профессиональных разработчиков (программистов, системных аналитиков, проектировщиков баз данных и др.), экспертов в предметных областях, конечных пользователей (бизнес-аналитиков и пр.).

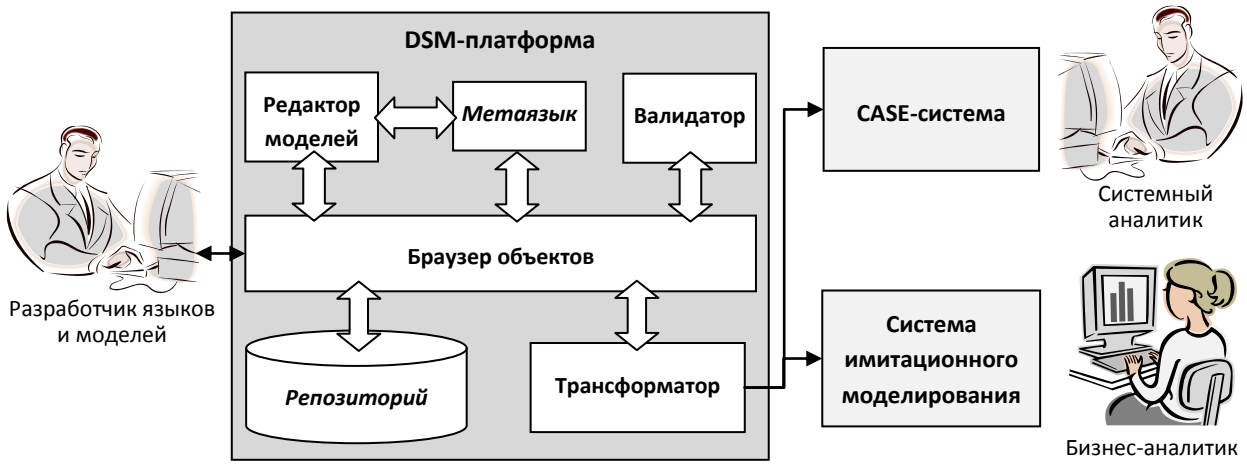


Рис. 1. Схема интеграции систем на основе DSM-платформы

Перечисленные требования в полном объёме в настоящее время не реализуются ни в одной из существующих DSM-платформ [7, 8].

Языковой инструментарий MetaLanguage

Языковой инструментарий MetaLanguage ориентирован на решение поставленных задач [4, 5, 15]. Порядок разработки моделей в системе MetaLanguage показан на рис. 2. 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.

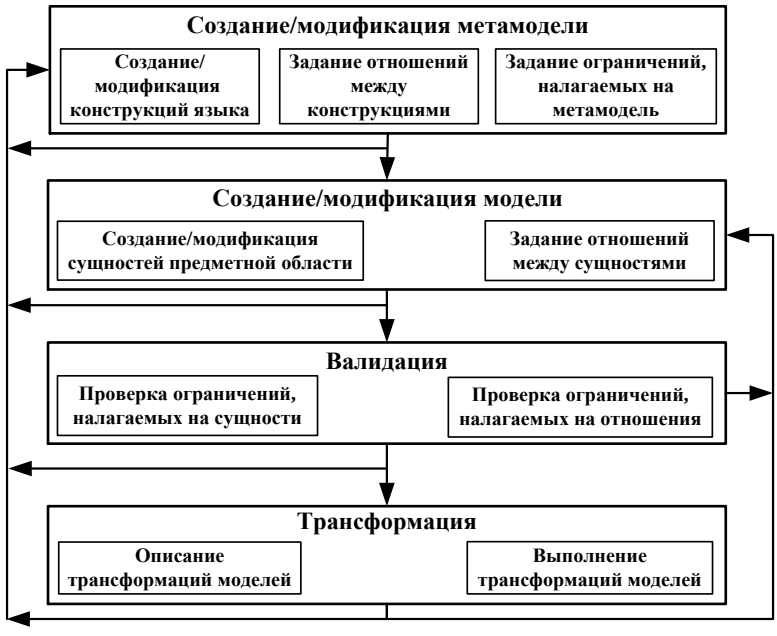


Рис. 2. Порядок разработки моделей в системе MetaLanguage

При создании метамодели в первую очередь определяются *базовые конструкции языка*. Базовыми элементами, которые используются в MetaLanguage для создания метамоделей (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.

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, 14]. 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 метамоделей, 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. Специальные средства позволяют при необходимости выполнить поиск наиболее подходящих моделей в репозитории на основе созданных онтологий [11].

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. 3):

- *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 обслуживающего устройства в случае, если оно занято). Сущность «Очередь» имеет следующие атрибуты: «Название», «Дисциплина», «Максимальная длина», «Текущая длина».
- *Обслуживающее 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».

Рис. 3. Метамодель языка описания имитационных моделей

Далее опишем отношения между сущностями метамодели. Как видно из рис. 3, 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.
- Двунаправленное отношение ассоциации «Обслуживает транзакты» позволяет указать, каким обслуживающим устройством обрабатываются заявки, находящиеся в очереди, и куда они направляются после обслуживания.
- Однонаправленное отношение ассоциации «Передает транзакты», соединяющее сущности «Обслуживающее 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 обсуживающими 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.

Метамодель 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. 4 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 (Q1, Q2, Q3, Q4), размножитель, коллектор и терминатор; используется два распределения.

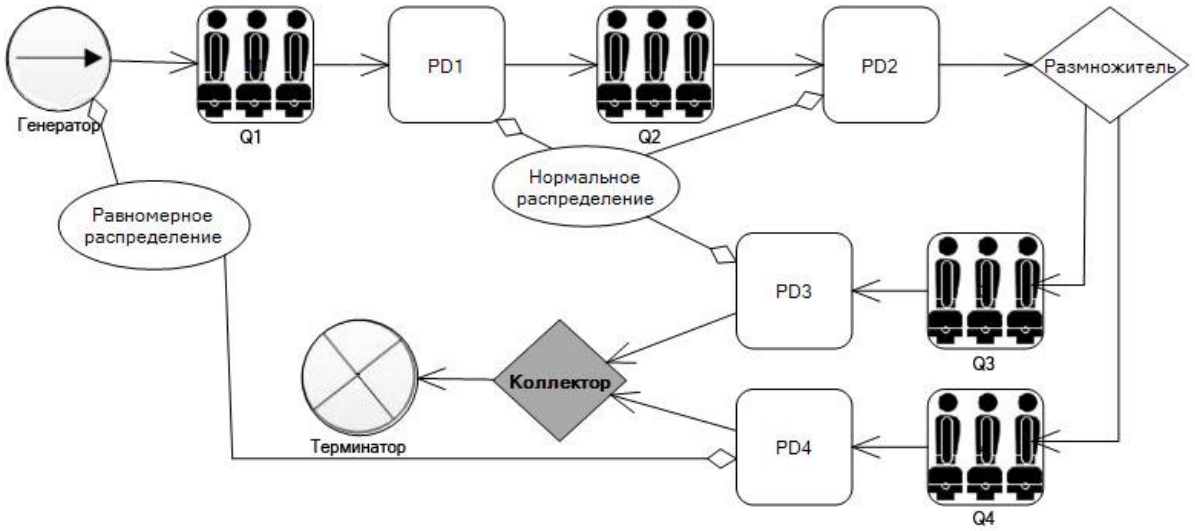


Рис. 4. Пример модели СМО

Опишем правила преобразования построенной метамодели языка описания СМО в нотации языка GPSS. Применение этих правил к построенной модели позволит сгенерировать программу на языки GPSS и выполнить анализ модели с использованием данной системы имитационного моделирования.

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

→

```
GENERATE (NORMAL(1, <<C.Нормальное распределение.
         Математическое ожидание>>,
         <<C.Нормальное распределение. Дисперсия>>)), ,
         <<C.Генератор.Начальная задержка>>,
         <<C.Генератор.Количество>>,
         <<C.Генератор.Приоритет>>
```

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:

→

```
QUEUE    <<C.Очередь.Название>>
SEIZE    <<C.Обслуживающее устройство.Название>>
DEPART   <<C.Очередь.Название>>
ADVANCE  (NORMAL(1, <<C.Нормальное распределение.
         Математическое ожидание>>,
         <<C.Нормальное распределение.Дисперсия>>))
RELEASE  <<C.Обслуживающее устройство.Название>>
```

Аналогичным образом описываются правила для других видов распределений.

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

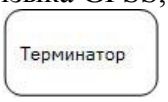
→

```
SPLIT  <<C.Размножитель.Количество копий>>,
       <<C.Размножитель.Блок>>
```

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

→ `ASSEMBLE  <<C.Коллектор.Количество>>`

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



→ `TERMINATE  1`

После применения описанной трансформации к модели, изображённой на рис. 4 системой MetaLanguage был сгенерирован следующий код на языке GPSS:

```
GENERATE   (UNIFORM(1, 2, 8), , 20, 100, 1)
QUEUE      Q1
SEIZE      PD1
DEPART     Q1
ADVANCE    (NORMAL(1, 3, 1))
RELEASE    PD1
QUEUE      Q2
SEIZE      PD2
DEPART     Q2
ADVANCE    (NORMAL(1, 3, 1))
RELEASE    PD2
SPLIT      1, Q4
QUEUE      Q3
SEIZE      PD3
DEPART     Q3
ADVANCE    (NORMAL(1, 3, 1))
RELEASE    PD3
QUEUE      Q4
SEIZE      PD4
DEPART     Q4
ADVANCE    (UNIFORM(1, 2, 8))
RELEASE    PD4
ASSEMBLE   2
TERMINATE  1
```

Аналогично можно определить правила трансформации конструкция языка описания СМО в графические нотации языка GPSS (каждому оператору языка соответствует блок). Модель системы в виде СМО может быть автоматически сгенерирована на основе описаний бизнес-процессов, разработанных аналитиками, если определить правила трансформации конструкций используемых языков моделирования бизнес-процессов в соответствующие конструкции построенного языка описания СМО. Построенная модель может уточняться и расширяться.

Заключение

Представленный языковой инструментарий является достаточно удобным и гибким инструментом разработки языков моделирования и правил преобразования моделей, созданных с использованием этих языков. Средства трансформации моделей – это основа для реализации средств интеграции различных систем, использующих модели для решения прикладных задач. Возможность экспорта и импорта моделей обеспечивает «переиспользуемость» разработанных моделей при решении задач создания ИС, анализа процессов и систем. Это позволяет снизить трудоёмкость работы аналитиков, повысить эффективность функционирования информационных систем, использования имеющихся ресурсов.

Исследовательский прототип системы MetaLanguage был использован для разработки нескольких предметно-ориентированных языков, в частности, моделирования административных регламентов, производственных процессов, приложений для мобильных устройств и пр. 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).

Библиографический список

1. *Замятина Е.Б.* Интеграция информационных систем и систем имитационного моделирования на основе многоуровневых моделей / *Е.Б. Замятина, Л.Н. Лядова, А.И. Миков, А.И. Якимов* // Математика программных систем: межвуз. сб. науч. ст. – Пермь: Изд-во Перм. гос. ун-т. – Пермь, 2008. С. 12-23.
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.

7. *Сухов А.О.* Классификация предметно-ориентированных языков и языковых инструментариев / *А.О. Сухов* // Математика программных систем: межвуз. сб. науч. ст. – Пермь: Изд-во Перм. гос. нац. исслед. ун-та, 2012. – Вып. 9. – С. 74-83.
8. *Сухов А.О.* Сравнение систем разработки визуальных предметно-ориентированных языков / *А.О. Сухов* // Математика программных систем: межвуз. сб. науч. ст. – Пермь: Изд-во Перм. гос. нац. исслед. ун-та, 2012. – Вып. 9. – С. 84-111.
9. *Balasubramanian K.* Component-Based System Integration via (Meta)Model Composition / *K. Balasubramanian, D.C. Schmidt, Z. Molnar, A. Ledeczi* // Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07). Tucson, Arizona. March 26-29, 2007. P. 93-102.
10. *Bräuer M.* Towards Semantic Integration of Multiple Domain-Specific Languages Using Ontological Foundations / *M. Bräuer, H. Lochmann* // Proceedings of the 4th International Workshop on (Software) Language Engineering (ATEM'07), Nashville, 2007. 15 pp.
11. *Elokhov E.* An Approach to the Selection of DSL Based on Corpus of Domain-Specific Documents / *E. Elokhov, E. Uzunova, M. Valeev, A. Yugov, V. Lanin* // Proc. of the 7th Spring/Summer Young Researchers' Colloquium on Software Engineering. M.: Изд-во Инст. сист. прогр. РАН, 2013. – P. 139-143.
12. *Heitkötter H.* A Framework for Creating Domain-specific Process Modeling Languages // Proceedings of the 7th International Conference on Software Paradigm Trends (ICSOFT 2012). Rome, Italy. July 24-27, 2012. P. 127-136.
13. *Pedro L.* Composing Visual Syntax for Domain Specific Languages / *L. Pedro , M. Risoldi, D. Buchs, B. Barroca, V. Amaral* // Human-Computer Interaction, Vol. 5611 (2009). Heidelberg, Springer. P. 889-898.
14. *Sukhov A.O.* Horizontal Transformations of Visual Models in MetaLanguage System / *A.O. Sukhov, L.N. Lyadova* // Proc. of the 7th Spring/Summer Young Researchers' Colloquium on Software Engineering. M.: Изд-во Инст. сист. прогр. РАН, 2013. – P. 31-40.
15. *Sukhov A.O.* MetaLanguage: a Tool for Creating Visual Domain-Specific Modeling Languages / *A.O. Sukhov, L.N. Lyadova* // Proc. of the 6th Spring/Summer Young Researchers' Colloquium on Software Engineering. M.: Изд-во Инст. сист. прогр. РАН, 2012. – P. 42-53.