

## МЕТАМОДЕЛИРОВАНИЕ И МНОГОУРОВНЕВЫЕ МЕТАДАННЫЕ КАК ОСНОВА ТЕХНОЛОГИИ СОЗДАНИЯ АДАПТИРУЕМЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Людмила Лядова

**Аннотация:** Рассматриваются методы создания распределенных информационных систем, динамически настраиваемых на меняющиеся потребности пользователей и условия эксплуатации. Описываемые средства основаны на использовании многоуровневых моделей и метаданных, представляющих различные стороны функционирования систем на разных уровнях абстракции и с различных точек зрения. Основные уровни метаданных, описывающих систему: логический (описание объектов системы в терминах предметной области), физический (описание представления данных в базе данных) и презентационный (описание интерфейса пользователя системы). Модели и набор метаданных могут изменяться в процессе функционирования системы. На основе базовых моделей могут разрабатываться новые модели (в частности, созданы Web-модель, модели репортинга и бизнес-процессов). Представленный подход реализуется в CASE-технологии METAS, предназначенной для поддержания всего жизненного цикла адаптируемых систем. Функционирование системы строится на интерпретации построенных моделей. Возможности адаптации основаны на средствах реструктуризации данных, генерации и настройки пользовательского интерфейса, управления документами, подключения новых программных компонентов. В CASE-систему включены средства экспорта-импорта, реплицирования данных и моделей, интеграции с внешними системами, а также средства защиты. Разрабатываемые с использованием технологии информационные системы имеют клиент-серверную архитектуру. Технология METAS базируется на использовании языка UML и предметно-ориентированных языков для разработки моделей системы, описания бизнес-правил, специфических для конкретных предметных областей. Предусмотрены средства, позволяющие настраиваться на использование различных реляционных СУБД. Программная платформа – .NET.

**Ключевые слова:** адаптируемые информационные системы, CASE-технология, модель предметной области, метамоделирование, метаданные, предметно ориентированные языки, DSL, DSM.

**ACM Classification Keywords:** D.2 Software Engineering: D.2.2 Design Tools and Techniques – Computer-aided software engineering (CASE); D.2.11 Software Architectures – Domain-specific architectures; D.2.13 Reusable Software – Domain engineering, Reuse models.

**Conference:** The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008

---

### Введение

Адаптируемость (способность систем приспосабливаться к изменениям среды, окружения) является одним из наиболее важных требований, предъявляемых к информационным системам (ИС) различного назначения. Адаптируемость рассматривают достаточно широко, включая в это понятие такие взаимосвязанные нефункциональные требования как способность к развитию, гибкость, расширяемость, интероперабельность и т.п. [1, 2]. Такая широкая природа этого понятия делает его не только интересным для исследования, но и критичным свойством в практике создания ИС, определяющим эффективность вложений в их разработку и внедрение, эксплуатацию и сопровождение, гарантирующим «живучесть» ИС.

Адаптация информационных систем – это процесс их настройки на меняющиеся условия эксплуатации и потребности пользователей и бизнес-процессов как при создании новых систем, так и при сопровождении существующих. Этот итеративный процесс можно считать важнейшей частью жизненного цикла ИС.

Адаптируемость – это характеристика, определяющая способность системы к развитию в соответствии с

нуждами пользователей и бизнеса. Различают понятия адаптивных и адаптируемых систем: *адаптируемые* системы – это «легко изменяемые» системы, включающие средства, которые обеспечивали бы их настройку на новые требования и условия динамически (в ходе эксплуатации), облегчали бы их сопровождение; *адаптивные* системы – это системы, которые меняют свое поведение автоматически в соответствии с изменениями, происходящими в их окружении («контексте»), настраиваются на изменения среды без применения каких-либо средств «ручной» настройки.

Существуют общие подходы к созданию адаптируемых ИС. Одним из таких подходов является ориентация на разработку систем, основанных на *MDA (Model Driven Architecture)* и технологии *DSM (Domain Specific Modeling)*. Средства адаптации могут базироваться на использовании *динамических языков программирования*, допускающих возможность переопределения структуры программ и данных, модернизации программ за счет изменения их компонентов, языков *DSL (Domain-Specific Languages)*, которые разрабатываются для разнообразных предметных областей [3, 4].

Максимальная гибкость при создании ИС достигается, если работа системы строится на использовании моделей, которые могут изменяться в процессе функционирования системы, управляющих ее поведением. Модели в MDA могут быть организованы на различных уровнях абстракции и платформенной независимости, что обеспечивает максимальную эффективность процесса разработки и возможность трансформации моделей [5, 6, 7].

Создание адаптируемых систем предполагает использование соответствующих инструментальных средств, поддерживающих предъявляемые к системам требования. Таким образом, можно считать, что свойство адаптируемости является необходимым не только для разрабатываемых систем, но и для применяемых при их разработке инструментальных средств.

В данной работе представлен подход к созданию адаптируемых информационных систем, основанный на построении многоуровневых моделей и использовании метаданных, представляющих информационные системы и их окружение с различных точек зрения и на различных уровнях детализации [8].

---

### Метамоделирование и технологии создания информационных систем

---

Модель – это объект-«заменитель» объекта-«оригинала», который находится в определенном соответствии с оригиналом и обеспечивает представление о некоторых его свойствах. *Модель* системы представляет собой *абстрактное описание* на некотором формальном языке характеристик системы, важных с точки зрения цели моделирования, ее поведения. При создании ИС нельзя ограничиваться созданием только одной модели. Если система сложная, то учет всех ее характеристик в одной модели приведет к чрезвычайной ее сложности. Наилучший подход при разработке любой нетривиальной системы – использовать совокупность нескольких моделей, которые могут быть практически независимыми друг от друга и позволят сделать акценты на разных сторонах системы при решении различных задач поддержания ее жизненного цикла.

В общем случае модели можно разделить на следующие виды: *статические*, описывающие структурные свойства систем; *динамические*, представляющие поведенческие свойства систем; *функциональные*, описывающие функциональные свойства систем. Статическая модель описывает составные части системы, их структуру, атрибуты, взаимосвязи между ними и операции, которые они могут выполнять. Операции статической модели являются событиями динамической и функциями функциональной моделей. Динамическая модель описывает последовательность выполнения операций в процессе функционирования системы. Функциональная модель описывает преобразования, осуществляемые системой. Она раскрывает содержание операций статической модели и событий динамической.

По *степени абстракции* модели можно разделить на *концептуальные модели*, представляющие высокоуровневый взгляд на задачу в терминах предметной области; *модели спецификации*, определяющие «внешний вид» и внешнее поведение системы; *модели реализации*, которые отражают внутреннее устройство системы, конкретный способ реализации наблюдаемого поведения системы.

Существуют различные определения метамодели. Исходя из того, что модели, создаваемые при разработке ИС, должны быть описаны на каком-либо формальном языке – языке моделирования, мы будем считать, что *метамодель* – это модель языка моделирования, применяемого для формализации описания системы. *Лингвистическая метамодель* – это метамодель, которая описывает предметно-независимый язык моделирования. *Онтологическая метамодель* – это метамодель, которая описывает предметно-зависимый язык моделирования.

Четырехуровневая иерархия моделей представляет классический вариант метамоделирования при создании ИС (рис. 1). В данной иерархии каждый вышестоящий уровень определяет язык для описания нижестоящего уровня. Число уровней при реализации конкретных систем может изменяться.

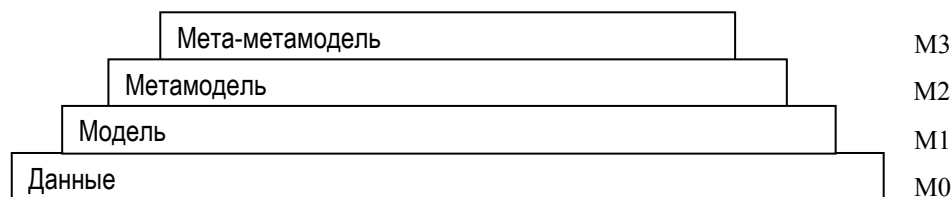


Рис. 1. Классическая четырехуровневая иерархия моделей ИС

При решении задачи моделирования ИС на уровне M0 находятся данные, описывающие состояние предметной области, т.е. модель состояния. Уровень M1 является онтологической метамоделью для уровня M0 и содержит модель предметной области. Уровень M2 определяет лингвистическую метамодель для уровней M1 и M0: на уровне M2 находится модель языка моделирования, с которым работают аналитики, разработчики, CASE-средства и пр. Самый верхний уровень (M3) определяет язык, на котором описываются метамодели уровня M2.

В зависимости от количества уровней создаваемых моделей и способа их использования при разработке информационных систем и технологии их создания можно разделить на несколько классов.

В традиционной информационной системе «внутри» системы находятся данные, описывающие состояние ее предметной области. Эти данные соответствуют некоторой модели предметной области, которая может быть описана на любом языке (в том числе и на естественном). Модель разрабатывается аналитиками, после чего разработчики реализуют ее при помощи выбранных инструментальных средств создания ИС, систем программирования. В случае изменения модели приходится переписывать и перекомпилировать исходные коды приложений системы. Чаще всего при реализации ИС и в процессе ее сопровождения разработчики «уходят» от начальной модели, которая не обновляется при внесении изменений в структуры данных и алгоритмы их обработки, что приводит к различным проблемам.

В традиционных CASE-технологиях модель предметной области определяется формально и находится «внутри» CASE-системы. Модель описывается в терминах метамодели, которая может быть определена на любом языке и реализуется с помощью CASE-средств. Изменение модели ведет к необходимости изменения кода, сгенерированного CASE-системой. Как и в случае с традиционной информационной системой, метамодель разрабатывается аналитиками, после чего реализуется разработчиками. Изменение метамодели влечет за собой переписывание и перекомпиляцию CASE-средства, однако такие изменения происходят крайне редко. Современные CASE-средства предоставляют инструменты для создания и редактирования моделей, а также позволяют сгенерировать большую часть кода информационной системы. Полученная на выходе система обычно реализует все необходимые структуры данных, определяемые моделью, обеспечивает доступ к данным в базах данных (БД) и предоставляет стандартный интерфейс пользователя для работы с ними. Программные компоненты, реализующие специфические для конкретной системы поведенческие и функциональные аспекты, дописываются чаще всего вручную. В случае изменения модели CASE-система позволяет заново

сгенерировать код приложений ИС, при этом код, добавленный программистами «вручную», сохраняется (при соблюдении определенных правил его написания). После повторной генерации обычно требуется ручная доработка кода. Достоинством подхода является то, что существенно экономится время на начальных этапах разработки. Кроме того, поддерживается соответствие между системой и моделью.

*Информационные системы, управляемые метаданными*, обеспечивают более мощные возможности для динамической адаптации. В данном случае также используются три уровня моделей, однако построенная модель предметной области находится «внутри» информационной системы в процессе ее эксплуатации. Таким образом, программное обеспечение информационной системы выступает в роли *интерпретатора*, а модель – в роли «управляющей системы», задающей правила функционирования ИС. Недостатком такого подхода является то, что несколько снижается производительность системы в ходе ее эксплуатации. Кроме того, если отсутствует возможность подключения внешних программных компонентов, расширяющих функциональность системы, то страдает универсальность вследствие невозможности реализации специфических для конкретной системы функций, отражающих бизнес-логику предметной области. Соответственно, метамодель должна быть максимально мощной. К достоинствам следует отнести тот факт, что при изменении модели не требуется повторное кодирование или перекомпиляция – информационная система просто начинает работать в соответствии с новой моделью.

*Технология DSM (Domain Specific Modeling) с генерацией кода* обеспечивает моделирование в терминах предметной области. В данном случае для решения каждой задачи применяется свой язык моделирования, в котором используются исключительно понятия и отношения из предметной области ИС.

Здесь используется уже мета-метамодель, которая реализуется Мета-CASE-средством. При помощи этого средства описывается метамодель, которая определяет предметно-зависимый язык моделирования. На основе этой модели генерируется CASE-средство, при помощи которого описывается модель предметной области и генерируется информационная система. Мета-CASE- и CASE-средства могут быть объединены в одну CASE-систему.

Использование предметно-зависимого языка (Domain Specific Language, DSL) позволяет существенно упростить процесс создания моделей предметной области, в котором могут принимать активное участие эксперты – специалисты в данной предметной области. Прочие преимущества и недостатки, связанные с генерацией кода, совпадают с соответствующими характеристиками традиционной CASE-технологии.

*Технология DSM с интерпретацией метаданных* (рис. 2) обеспечивает максимальные возможности адаптации. Данный вариант является комбинацией двух предыдущих. Метамодель, модель и данные ИС находятся «внутри» информационной системы. В этом случае CASE-средства позволяют создать модели и интерпретировать их в ходе эксплуатации системы (для этого разрабатываются специальные run-time компоненты).

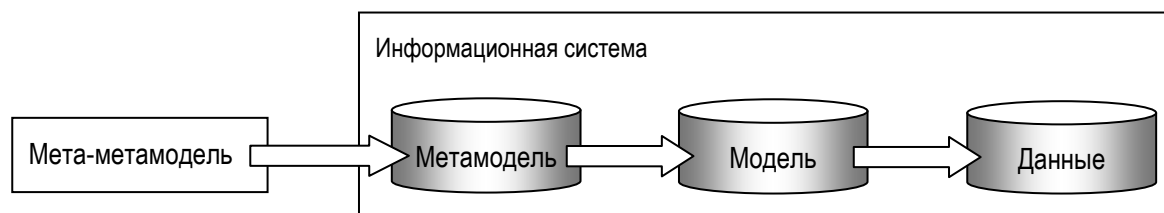


Рис. 2. Технология DSM с интерпретацией метаданных

Для того чтобы данный подход оказался применимым на практике, необходимо чтобы мета-метамодель была максимально выразительной. Интерпретация сразу двух уровней метамodelей приводит к ощутимой потере производительности, однако при достаточной выразительности мета-метамodelи получается чрезвычайно гибкая система. Данный подход реализуется в представленной в данной работе CASE-технологии METAS, разрабатываемой в АНО науки и образования «Институт компьютеринга».

---

## Технология создания динамически адаптируемых информационных систем, управляемых метаданными

---

Максимальная гибкость ИС может быть достигнута, если как при разработке системы, так и в ходе ее эксплуатации применяются *метаданные*, описывающие особенности предметной области, для которой создается система, условия ее работы и характеристики бизнес-процессов и пользователей.

CASE-технология METAS (METAdata System) – это основа для создания динамически настраиваемых информационных систем, управляемых метаданными, повышения их адаптируемости в процессе эксплуатации за счет использования многоуровневых моделей. Ключевым моментом технологии является использование *взаимосвязанных метаданных*, описывающих информационную систему и ее окружение с различных точек зрения и на разных уровнях детализации.

Основное отличие данной CASE-технологии от других, генерирующих код приложений ИС на каком-либо языке программирования по заданным спецификациям, описывающим предметную область ИС (ее модель) и окружение, состоит в том, что в данном случае это описание используется во время работы программного ядра ИС, которое выполняет функции представления данных и их обработки по определенным этими метаданными правилам, *интерпретируя* их. Это создает хорошие предпосылки для создания «интеллектуальной» системы, которая может настраиваться на потребности пользователя и меняющиеся условия эксплуатации, *в ходе работы с ней пользователей*. Кроме того, при таком подходе проект обладает высокой степенью обратной связи, так как разработчик, меняя модели (метаданные), сразу видит соответствующие изменения в реализуемой на основе данной технологии ИС (в ее информационных объектах и связях между ними, в интерфейсе пользователя и функциональности и т.п.), потому что он фактически работает с той же системой, что и пользователь, но используя специальный CASE-инструментарий. При обычном же подходе, реализованном в большинстве CASE-систем, разработчик описывает модель системы, после чего выполняется генерация кода приложений, и только после этого он может оценить результат внесенных в модель изменений.

Технология METAS базируется на использовании языка UML и предметно-ориентированных языков для разработки моделей ИС, описания бизнес-правил, специфических для конкретных предметных областей; на технологии RUP (Rational Unified Process) и технологии разработки XP (eXtreme Programming); на платформе .NET Framework и инструментальных средствах MDK Suite (Meta Data Kernel Suite).

Метаданные представляют *формализованное описание* ИС, размещенное в базе метаданных (БМД), используемое для настройки приложения на условия эксплуатации в процессе его разработки, а затем – загрузки и выполнения. Они описывают следующие аспекты ИС: объекты ИС и поведение объектов ИС, бизнес-операции и бизнес-процессы, первичные документы и отчеты, визуальный интерфейс пользователя ИС, модель защиты. Метаданные представляют *модели*, каждая из которых описывает определенную часть, аспект ИС (некоторые модели могут описывать одни и те же части ИС, но с различных точек зрения). Таким образом, метаданные в METAS – это взаимосвязанные модели (рис. 3), одна модель может основываться на другой, и представлять собой более высокоуровневое описание ИС. Программные компоненты системы работают с метаданными соответствующего уровня (или нескольких взаимосвязанных уровней).

Метаданные ИС разделены на слои, представляющие следующие основные модели:

- *Физическая модель* (Physical Model) – метаданные, описывающие представление объектов ИС в БД (например, таблиц БД, в которой хранятся данные об объектах, и связей между ними). В процессе функционирования они служат основой логической модели. Модель автоматически генерируется по созданному на логическом уровне описанию системы.
- *Логическая модель* (Logical Model) – метаданные, описывающие сущности предметной области, для которой создается ИС, их поведение (через операции), а также общие операции ИС. Данная модель основывается на нотациях языка UML и позволяет работать пользователям системы в терминах предметной области.

- *Презентационная модель* (Presentation Model) – метаданные, описывающие визуальный интерфейс пользователя при работе с объектами ИС.

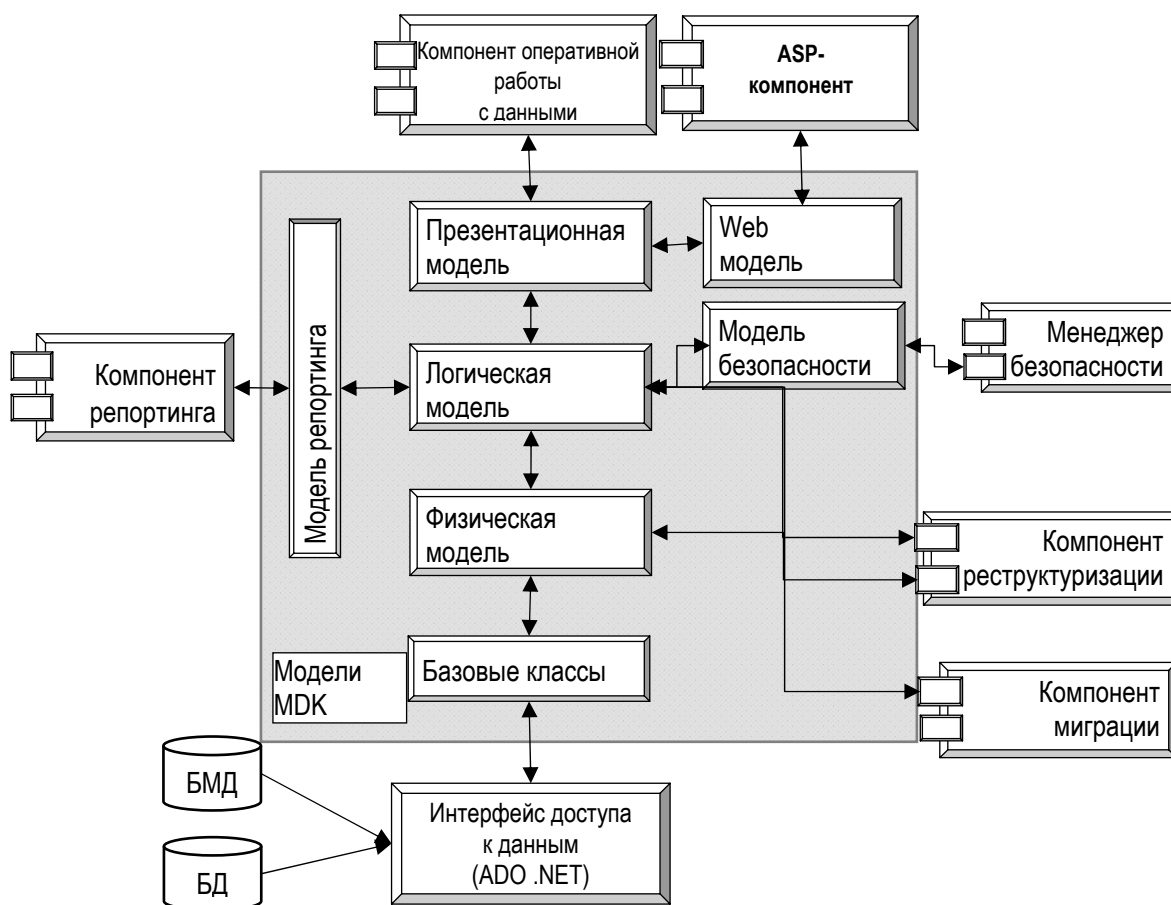


Рис. 3. Модели метаданных и компоненты METAS

Набор характеристик, отражаемых в модели, может быть динамически расширен. Набор метаданных может также расширяться путем добавления новых моделей, описывающих новые стороны и свойства ИС или существующие, но с новых точек зрения. В частности, в систему включены следующие модели, опирающиеся на перечисленные выше основные:

- *модель репортинга* (Reporting Model) – метаданные, описывающие запросы, первичные документы и отчеты, формируемые в ходе выполнения бизнес-операций и бизнес-процессов, используемые для анализа данных;
- *модель бизнес-процессов* (Business-process Model) – метаданные, описывающие бизнес-операции и бизнес-процессы, поддерживаемые ИС;
- *Web-модель*, которая обеспечивает доступ к ресурсам ИС для удаленных пользователей через Web-интерфейс.

*Модель безопасности* (Security Model) позволяет контролировать полномочия пользователей, их права на выполнение операций над объектами ИС или на доступ к моделям метаданных. Подсистема защиты работает с собственной БД.

CASE-инструментарий позволяет описывать объекты и бизнес-процессы ИС, строить запросы и отчеты в терминах предметной области, настраивать стандартно сгенерированные формы ввода и отображения данных, размещенных в БД системы, а также экспортировать и импортировать модели и данные динамически. Стандартная бизнес-логика может быть расширена путем определения новых типов и

операций, специфичных для конкретной ИС. Обеспечивается адаптация ИС без перепрограммирования ее компонентов и без участия разработчиков. Средства интеграции ИС на основе технологии BizTalk Server реализованы как отдельное приложение.

### Архитектура информационной системы, созданной на основе технологии METAS

Использование CASE-технологии METAS позволяет создать ИС, архитектура которой представляет собой клиент-серверное приложение (рис. 4), разбитое на *домены*.

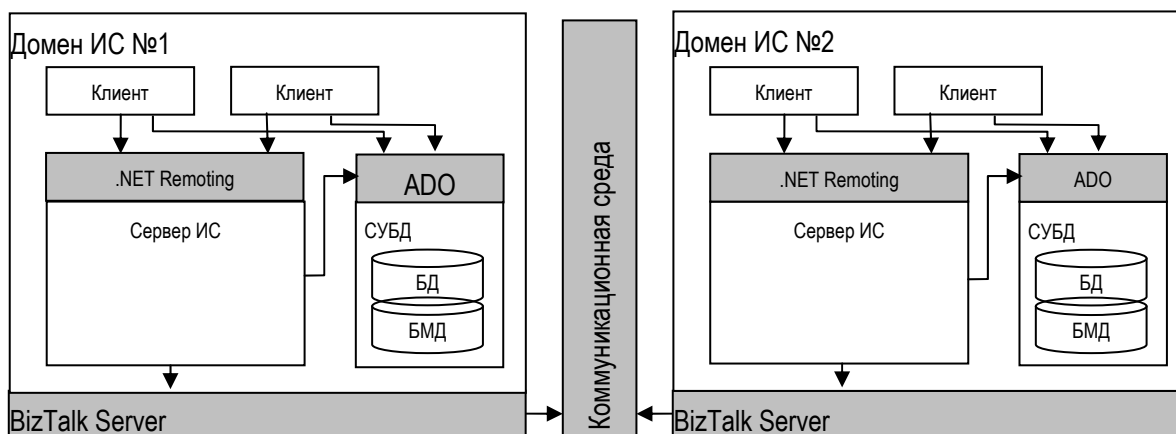


Рис. 4. Доменная архитектура ИС, созданной на основе METAS

Разбивка на домены предназначена для реализации распределенных ИС, включающих автономно функционирующие, не имеющие оперативной связи подсистемы. *Домен ИС* – законченное распределенное приложение, представляющее собой подсистему корпоративной ИС, установленную в отдельном учреждении, состоящее из одного сервера и нескольких клиентов.

Связь между доменами ИС устанавливается для реплицирования моделей и данных, обмена документами, отчетами.

### Заключение

Основными преимуществами представленной технологии являются:

- *гибкость и возможность динамической адаптации* системы к изменениям условий функционирования, потребностям пользователей с минимальными затратами;
- *возможность интеграции* с внешними системами;
- *отсутствие необходимости специальной подготовки пользователей*, обеспечение возможности работы в привычной для них среде и в терминах знакомой предметной области;
- *невысокие требования к программно-аппаратной платформе* при достаточно мощных возможностях сбора, хранения и обработки данных.

Разработанные средства служат основой для разработки средств автоматической адаптации, основанных на использовании онтологий и агентных технологиях.

### Благодарности

Работа выполнена при поддержке гранта РФФИ № 08-07-90006-Бел\_а.

**Библиографический список**

---

- [1] Chung L., Subramanian N. Adaptable system/Software architectures // Journal of Systems Architecture: the EUROMICRO. Special issue: Adaptable system/Software architectures. Vol. 50, Issue 7 (July 2004). Pp. 365-366.
- [2] Subramanian N., Chung L. Software Architecture Adaptability: An NFR Approach // Proc. Int. Workshop on Principles of Software Evolution (IWPSE'01) / Vienna, Austria. ACM Press, (September 2001). Pp. 52-61. [PDF] ([www.utdallas.edu/~chung/ftp/IWPSE.pdf](http://www.utdallas.edu/~chung/ftp/IWPSE.pdf)).
- [3] Cook St. Domain-Specific Modeling and Model Driven Architecture // MDA Journal, January 2004. Pp. 2-10. [PDF]. ([www.bptrends.com/publicationfiles/01-04%20COL%20Dom%20Spec%20Modeling%20Frankel-Cook.pdf](http://www.bptrends.com/publicationfiles/01-04%20COL%20Dom%20Spec%20Modeling%20Frankel-Cook.pdf)).
- [4] Savidis A. Dynamic software assembly for automatic deployment-oriented adaptation // Preproceedings of the Workshop on Software Evolution through Transformations: Model-based vs. Implementation-level Solutions "SETra 2004". A satellite event of ICGT 2004. October 2004, Rome, Italy. Pp. 191-198.
- [5] Almeida J.P., Pires L.F., van Sinderen M. Costs and Benefits of Multiple Levels of Models in MDA Development // Proceedings of the Second European Workshop on Model Driven Architecture (MDA) with an emphasis on Methodologies and Transformations "Computer Science at Kent". September 2004. Canterbury, UK. Pp. 12-21.
- [6] Favre J.-M., Nguyen T. Towards a Megamodel to Model Software Evolution Through Transformations // Preproceedings of the Workshop on Software Evolution through Transformations: Model-based vs. Implementation-level Solutions "SETra 2004". A satellite event of ICGT 2004. October 2004, Rome, Italy. Pp. 56-70.
- [7] Atkinson C., Kühne Th. The Essence of Multilevel Metamodeling // Proceedings of UML 2001 – The Unified Modeling Language. Modeling Languages, Concepts, and Tools: 4th International Conference, V. 2185 of LNCS. Toronto, Canada, October 2001. Springer. Pp. 19-33.
- [8] Лядова Л.Н. Технология создания динамически адаптируемых информационных систем // Труды междунар. науч.-тех. конф. «Интеллектуальные системы» (AIS'07). Т. 2. – М.: Физматлит, 2007. С. 350-357.

---

**Сведения об авторе**

---

**Людмила Лядова** – заместитель директора Автономной некоммерческой организации науки и образования «Институт компьютеринга»; Россия, 614097, г. Пермь, ул. Подлесная, 19/2-38;  
e-mail: [LNLyadova@mail.ru](mailto:LNLyadova@mail.ru)