OSTIS-2015
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822

ТЕХНОЛОГИЯ АВТОМАТИЗИРОВАННОГО СИНТЕЗА ИНФОРМАЦИОННЫХ СИСТЕМ С ПОМОЩЬЮ СЕМАНТИЧЕСКИХ МОДЕЛЕЙ ПРЕДМЕТНОЙ ОБЛАСТИ

Бикмуллина И.И.

Казанский национальный исследовательский технический университет им. А.Н. Туполева, г.Казань, Россия

elsiyar-b@yandex.ru

В работе рассмотрена проблема отсутствия автоматизированного структурного синтеза информационных систем на основе семантических отношений предметной области. Предложен искусственно интеллектуальный подход к автоматизированной разработке информационных систем, усовершенствование современной технологии за счет автоматизированного синтеза диаграмм классов UML на основе семантических моделей.
Ключевые слова: UML; синтез диаграммы классов; автоматизированный синтез структурных моделей; информационная система; разработка.

Введение

Современное общество уже немыслимо без информационных систем (ИС), которые пронизывают почти все области деятельности и постоянно развиваются и совершенствуются. Улучшение качества ИС неразрывно связано с процессом их разработки. Каждый маленький элемент этого, теперь уже огромного, процесса играет очень большое значение [Якунин, 2008].

В 60-е гг. прошлого столетия научное направление «Искусственный интеллект» в нашей стране еще только формировалось, но уже возникло четкое понимание того, что без концептуальной структуризации знаний, без их формального представления создать искусственный интеллект невозможно. Добиться этого интуитивными эвристическими приемами не удалось. Опираясь на взаимосвязи лингвистики с семиологией, праксиологией, теорией информации, белорусский ученый В.В. Мартынов предложил способ «исчисления языковых смыслов» [Массель и др., 2014].

В последние годы наиболее распространенной методологией создания программных систем является объектно-ориентированная (ОО) методология, появившаяся в 80-е годы, по которой окружающий нас мир состоит из объектов и отношений между ними.

Согласно В. Далю[Даль, 1975] объект (предмет) - это все, что представляется чувствам (объект вещественный) или уму (объект умственный). Таким образом, *объект* воплощает некоторую сущность и имеет некоторое состояние, которое может изменяться со временем как следствие влияния других объектов, находящихся с первым в каких-либо отношениях. Он может иметь внутреннюю структуру: состоять из других объектов, также находящихся между собой в некоторых отношениях. Исходя из этого, можно построить иерархическое строение мира из объектов. В процессе познания или изменения окружающего нас мира мы всегда принимаем в рассмотрение ту или иную упрощенную модель мира (*модельный мир*), в которую включаем объекты и отношения некоторых интересующих нас классов из окружающего нас мира. Таким образом, окружающий нас мир, можно рассматривать (в некотором приближении) как иерархическую структуру модельных миров [Кольцов и др., 2005].

Так в 90-е годы бурно развивается CASE-технология разработки программных систем и связанные с ней формальные методы спецификации программ. CASE(Computer Aided Software Engineering) или программная инженерия с компьютерной поддержкой – это система средств автоматизированного конструирования программного обеспечения (ПО).

Составными частями процесса разработки ИС с использованием CASE - средств являются:

- разработка словаря предметной области;

-разработка диаграмм с помощью языка представления данных UML[Рамбо и др., 2002]

(унифицированный язык моделирования);

- синтез программ.

Используя CASE-мышление, программист уже сделает свои программы лучше, поскольку представление программных объектов в диаграммах UML[Рамбо и др., 2002] позволяет наглядно увидеть ошибки и недоработки в полученной иерархии, легко этой иерархией манипулировать, что при ручном кодировании программист вряд ли может осуществить. Однако, установившийся, в настоящее время, стереотип «кодирование, а затем лишь моделирование» не позволяет в полной мере использовать мощные средства UML. А наличие ПО с последовательным, верным, автоматизированным процессом разработки ИС помогло бы облегчить труд разработчика.

В России широко используемыми CASE-средствами, поддерживающими ОО методы, являются Rational Rose Enterprise Edition 2000/2002 (фирмы Rational Software Corporation), Oracle Developer Suite 2000 (фирмы Oracle) и др.

Но для того чтобы перейти к создании и сопровождению кода при помощи CASE-средства, поддерживающего язык UML (например, Rational Rose), программист должен перестроить свое представление о создании программ. Так как необходимо мыслить уже в терминах языка UML, мыслить диаграммами, а переход к такому типу мышления требует примерно такого же усилия, как переход от процедурного программирования к ОО. Потому что в процессе разработки ПО главной сложностью является формализация предметной области, к которой относится решаемая задача. Для этого необходима хорошая предметно-ориентированная модель, проникающая значительно дальше поверхностного взгляда на проблему. Если в такой модели удастся правильно отразить внутреннюю структуру предметной области, то разработчики ПО получат именно тот инструмент, в котором они нуждаются.

Проблема исследования заключается в необходимости автоматизации процесса разработки (синтеза) диаграммы классов UML.

Поэтому актуальной можно признать цель данного исследования – повышение качества проектной проработки ИС за счет автоматизации процесса синтеза структурных моделей. Из множества структурных моделей информационной технологии выбрана модель диаграммы классов[Якунин, 2008] UML.

1. Анализ современной технологии разработки программы

Разработка интеллектуальных систем и систем обработки сложно-структурированной информации, является трудоемким процессом, но еще более сложным и трудоемким является сопровождение таких систем. Поэтому исследование методов, средств, технологии разработки и сопровождения остается одной из актуальных задач [Грибова и др., 2013].

В настоящее время наблюдается тенденция все большего вовлечения экспертов в разработку ИС с целью ее ускорения. Представляется очевидным, что чем меньше будет разрыв между знаниями и представлениями эксперта и средствами представления знаний инструментальной системы, тем легче ими будет пользоваться эксперту [Загорулько, 2013].

К числу основных условий, необходимых для создания в ближайшем будущем эффективных технологий проектирования ИС, можно отнести [Голенков и др., 2013]:

(1) Осознание того, что основным практическим результатом ИИ являются не сами ИС, а мощные и эффективные технологии их разработки.

(2) Осознание того, что без активного и массового привлечения молодых участников создания нового поколения проектирования ИС, указанные технологии не только не будут созданы, но и не смогут в дальнейшем динамично развиваться.

1.1. Определение этапов современной технологии разработки программы

Традиционно автоматизация сопровождается разработкой программного обеспечения практически под конкретную задачу, под класс задач.

Создание прикладного программного обеспечения (ПО) — это сложный процесс. Наиболее критичной его частью является непосредственно программирование приложения, так как на этом этапе осуществляется мысленное отображение понятий прикладной предметной области в используемый язык программирования, что требует знания как языка предметной области, так и языка программирования [Новиков и др., 2009]. На основе анализа технологий создания программных систем получена схема современной технологии создания программной системы (рисунок 1).

Рисунок 1 – Этапы современной технологии разработки ИС

Современная технология разработки программы (рисунок 1) в идеале предполагает:

1 этап. Считается, что эксперт самостоятельно, вручную или в текстовом редакторе разрабатывает онтологию, глоссарий предметной области. Однако, на данном этапе всегда участвую: эксперт рассматриваемой предметной области (например, машиностроение) и сам разработчик. При этом разрабатываемая система может опираться на несколько предметных областей, что влечет привлечение к разработке нескольких экспертов. В

этом случае разработчик помимо своего дела должен освоить еще ряд специальностей и добиться понимания с экспертами. А эксперту, в процессе разработки онтологии предметной области, трудно выражать свои мысли в процедурном виде [Джарратано и др., 2006], поэтому эти описания в значительном виде должны иметь декларативный характер [Зюзьков, 2003]. Декларативный характер описания близок к языку такому, на котором привык эксперт выражать свои мысли, описывать предметную область. При этом обработке декларативных описаний наиболее соответствует декларативный язык программирования, например, Пролог[Зюзьков, 2003], язык позитивно образованных формул[Васильев и др., 2000]. И все же автоматизация первого этапа в настоящее время изучена явно недостаточно.

2 этап. В идеале, на данном этапе эксперт и разработчик совместными усилиями проектируют диаграмму UML. Однако, чаще всего моделирование осуществляется разработчиком самостоятельно, вручную или составляется им в приложении какого-либо ПО.

Эта модель отражает физическую реализации системы и описывает создаваемый продукт на уровне классов и компонентов. В отличие от модели анализа, модель проектирования имеет явно выраженную зависимость от условий реализации, применяемых языков программирования и компонентов. Для максимально точного понимания архитектуры системы, эта модель должна быть максимально формализована, и поддерживаться в актуальном состоянии на протяжении всего жизненного цикла разработки системы [Трофимов, 2003]. Однако стремление разработчиков как можно быстрее получить программный код приводит к тому, что моделирование диаграмм UML разработчики используют лишь, в крайнем случае, когда проектировании достаточно сложного программного изделия зашло в тупик, а это бывает достаточно запоздалое использование данной технологии. Что ведет к снижению качества программного продукта, а также в необходимости дополнительного ведения документации с целью сопровождения ПО.

3 этап. Разработчик при автоматизированном проектировании ПО на основе разработанных моделей UML с помощью Rational Rose или Visual Studio автоматизированно генерирует программный исходный код.

Таким образом, так как вмешательство разработчика необходимо на каждом этапе проектирования, то увеличивается как загрузка разработчика, так и время создания программного изделия. Поэтому разработка диаграмм UML целиком зависит от квалификации программиста, знании данного программиста в области UML технологии, а также времени отведенного как на разработку диаграмм UML, так и на вникание в область рассматриваемой предметной области. Если автоматизировать процесс проектирования диаграмм классов UML, то интерес программистов к использованию инструмента моделирования UML может существенно возрасти.

1.2. Язык представления данных UML

Наиболее распространённым и развитым средством визуального моделирования является UML[Object Management Group, 2014] – язык графического описания для объектного моделирования в области разработки программного обеспечения. Методология UML позволяет производить дальнейшую разработку проекта с определением объекта автоматизации и построением для него информационной модели на физическом уровне вплоть до создания схемы базы данных и генерации кода заголовочных файлов на различных ОО языках [Ивашенко и др., 2004]. Составной частью ОО методологии является задача синтеза структурной модели информационной технологии. Частным случаем задачи синтеза структурной модели является получение диаграммы классов UML.

Класс (тип class языков программирования) описывает множество однотипных объектов с одним и тем же набором атрибутов и функций [Медведев, 2005].

2. Предлагаемое решение проблемы

Решение поставленной в данном исследовании проблемы заключается в модификации современной технологии разработки ИС, создании ПО, позволяющего автоматизировано синтезировать UML диаграмму классов. Разрабатываемый программный продукт как бы будет являться надстройкой над программным продуктом, Rational Rose.

Решение проблемы, недостаточной проработки программного продукта в результате нарушения технологии разработки ПО, заключается в расширении интеллектуальных возможностей автоматизированного проектирования ИС с использованием семантических структурных отношений предметной области.

2.1. Усовершенствованная технология разработки ИС

В предложенной технологии разработки ИС (рисунок 2) наряду с автоматизированной генерацией кода автоматизируется и этап синтеза диаграммы UML. В данной работе из-за невозможности «объять необъятное» синтез диаграммы UML рассматривается на примере синтеза диаграммы классов UML. Также в предлагаемой технологии во время описания онтологии предметной области экспертом, вмешательство программиста становится минимальным, а эксперт получает более высокий статус – статус разработчика.

Усовершенствованная технология разработки ИС (рисунок 2) состоит из следующих этапов:

1 этап: эксперт в приложении автоматизированного синтеза диаграмм UML (разрабатываемого ПО) на основе требований программы синтеза описывает (самостоятельно) онтологию, глоссарий предметной области.

2 этап: программа автоматически синтезирует UML диаграмму, в данной работе автоматизированный синтез рассматриваться будет на примере диаграммы классов.

3 этап: получение на основе полученной на вход диаграммы классов UML «исходника» программы (сгенерированного программного кода, например, в Rational Rose).

Рисунок 2 – Предлагаемая технология создания программной системы

Итак, разница между усовершенствованной и современной технологиями разработки ИС заключается в автоматизации второго этапа - синтеза диаграммы классов UML.

Так если в современной технологии разработки ИС (рисунок 1) чаще всего онтологию предметной области создает разработчик, используя эксперта лишь в виде консультанта, после чего разработчик (или привлекая эксперта) разрабатывает UML диаграмму. То в усовершенствованной технологии создания ИС (рисунок 2) эксперт разрабатывает онтологию предметной области с помощью дружественного интерфейса программы, которая автоматически синтезирует диаграмму классов UML.

Таким образом, эксперты в предлагаемой технологии создания ИС должны сами стать полноправными разработчиками программной системы наравне с разработчиком-программистом.

2.2. Предлагаемая методика автоматизированного процесса проектирования

Предлагается ИИ подход к разработке UML модели на примере построения диаграмм классов.

Синтез диаграмм классов включает:

1)построение всевозможных моделей диаграммы классов;

2)построение оптимальной модели.

Содержанием задачи автоматического синтеза диаграммы классов является выделение классов и назначение структурных отношений между классами. Как правило, при проектировании информационной технологии могут быть построены несколько структурных моделей, поэтому в нашем случае решение задачи может быть вариантным. Выбор подходящего варианта из представленных диаграмм классов UML остаётся за проектировщиком.

Общая схема процесса автоматизированного синтеза структурной модели, диаграммы классов UML приведена на рисунке 3, где представлен и процесс разработки экспертом онтологии предметной области (1 блок рисунка 3), и формализация предметной области(1этап и 2 блок рисунка 3) в приложении усовершенствованной технологии проектирования ИС, и синтез, который приведет к структурному решению как описание процесса проектирования, и вывод геометрического представления диаграммы (последний блок рисунка 3).

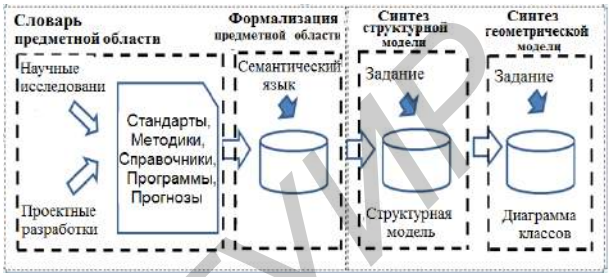

Рисунок 3 – Укрупненная схема процесса проектирования

Отсюда, предлагаемая методика включает в себя (рисунок 3):

1. Формирование описания модели предметной области экспертом. На основе онтологии, словаря предметной области (1 блок) эксперт заполняет приложение ПО автоматизированного синтеза UML диаграммы.

Формализация задачи моделирования семантики структурной модели программы, заключается в определении структурной модели программы с помощью добротной системы понятий.

Ниже приведены поля в приложении автоматизированного синтеза диаграммы классов UML, которые необходимо заполнить эксперту. Где применяюся следующие обозначения:

a – наименование (с маленькой буквы) понятия (consept);

B – наименование (с большой буквы) атрибута (свойства объекта);

b – значение атрибута (свойства объекта);

C – наименование (с большой буквы) функции;

c – значения функции;

D – ограничение области применения (воздействия данного объекта).

Понятие *a*
Атрибуты
B: {*b*}.
Функция *C* (*B*: {*b*}): вещественное.
Значение *c*
Свойство *D*
Конец *a*

Рассмотрим на примере: одной из задач в банковских информационных технологиях является получение прогноза функционирования банков. Прогнозирование осуществляется с помощью статистических характеристик банковских процессов. Для простоты выберем следующие

статистические характеристики: среднее арифметическое значение (оценка математического ожидания) и дисперсию. Необходимо решить задачу синтеза диаграммы классов. Семантическое описание задачи выглядит следующим образом:

Раздел *исходные_данные*
Понятие *массив_экспериментальных_данных*
Атрибуты
Размер : {целое}.
Данные : {вектор [Размер] из вещественное}.
Функции *Заполнить_вектор(Данные : {вектор [Размер] из вещественное}).*
Значение *_Ввод_вещественного_вектора(Данные).*
Свойство *Размер > 0.*
Конец *массив_экспериментальных_данных*
Конец_раздела *исходные_данные*
Раздел *основные_статистические_характеристики*
Понятие *среднее_значение*
Атрибуты
Данные : {массив_экспериментальных_данных}.
Значение : {вещественное}.
Функции *Получить_значение(Данные : {вектор из вещественное}): вещественное.*
Значение *_Сумма_вектора(Данные) / Размер из Данные.*
Свойство *нет*
Конец *среднее_значение*
Понятие *дисперсия*
Атрибуты
Данные : {массив_экспериментальных_данных}.
Среднее_значение : {среднее_значение}.
Значение : { вещественное }.
Функции *Получить_значение(Данные : {вектор из вещественное}, Среднее_значение : {среднее_значение}): вещественное.*
Значение *_Сумма_Квадрата_вектора(Данные) / Размер из Данные - _Квадрат(Среднее_значение).*
Свойство *нет*
Конец *дисперсия*
Конец_раздела
основные_статистические_характеристики

На данном этапе закладываются в семантический язык статистические характеристики:

(1) Спецификация свойств и особенностей класса, в нашем случае, прогноз банков состоит из разделов. Так спецификация «*массив экспериментальных данных*» содержит раздел: «*исходные данные*». А раздел *«основные статистические характеристики»* содержит такие спецификации, как «*дисперсия*» и «*значение среднего*».

(2) Спецификация профессионального раздела представляет собой систему описаний понятий исследования. Определение отдельного понятия имеет структуру: ***Понятие*** <название> <тело понятия> ***Конец*** <название>.

(3) Описание тела понятия является основой для автоматизированной обработки и может содержать следующие разделы: «*Параметры*», «*Состав*», «*Атрибуты*», «*Внешние*», «*Свойства*».

(4) Раздел описания «*Атрибуты*» определяет атрибутные свойства понятия и представляет собой бинарное отношение «денотат характеризуется …» между определяемым понятием и элементами раздела. Описание и использование раздела «*Атрибуты*» строится по тем же правилам, что и описание раздела «*Состав*».

2. Синтез структурной модели. Содержанием задачи автоматического синтеза диаграммы классов является выделение классов и назначение структурных отношений между классами. Таким образом, разрабатывается система аксиом, которая может быть использована в качестве правил принятия решений в автоматизированной системе синтеза диаграмм классов UML.

Ввиду простоты нашего пример а выделение классов является элементарным: структура классов соответствует структуре понятий. Поэтому основное внимание сосредоточим на синтезе структурных отношений. Проиллюстрируем, какие диаграммы классов можно синтезировать в нашем случае (перечень неполный). На рисунке 4 изображено шесть вариантов диаграммы классов, в которых используются отношение обобщения (рис. 4а), отношение композиции (рис. 4б), отношение агрегации (рис. 4в), отношение композиции и обобщения (рис. 4г, рис. 4д), отношение клиент-сервер (рис. 4е).

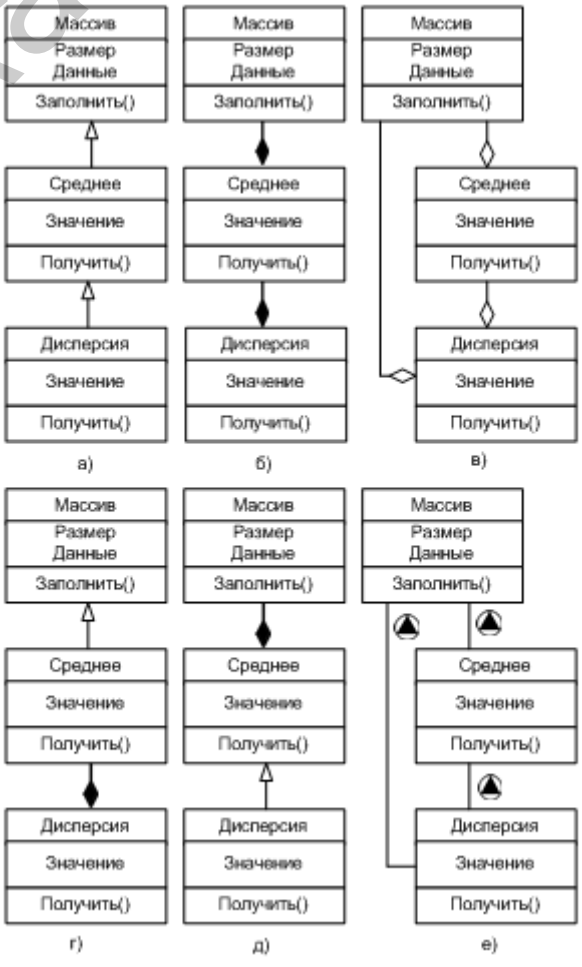

Рисунок 4 – Варианты диаграммы классов

Следовательно, создается формальная система, которая в качестве механизма структуризации данных включает в себя правила их построения, отражающие логику свойств структурной модели предметной области. В результате получим именно ту систему, которая будет использоваться для автоматизированного проектирования диаграммы классов, т.е. систему обработки описаний с четким теоретическим фундаментом.

Заключение

В работе представлено усовершенствование современной технологии разработки ИС за счет автоматизированного проектирования диаграмм классов UML на основе семантических моделей.

Автоматизация синтеза диаграммы классов дает возможность уменьшить трудоемкость проектирования структурных моделей ИС, повысить качество программного проекта за счет возможности многовариантного анализа структурных моделей, стимулировать разработчика на предварительное продумывание структуры прорабатываемой системы, изменить характер труда программиста в направлении от ручной проработки диаграмм UML к выбору оптимального варианта из множеств автоматически полученных вариантов.

Библиографический список

[Якунин, 2008] Якунин Ю.Ю. Технологии разработки программного обеспечения. Версия 1.0: электрон. учеб. пособие / Ю. Ю. Якунин. – Красноярск : ИПК СФУ, 2008.

[Массель и др., 2014] Массель Л.В., Массель А.Г. Ситуационное управление и семантическое моделирование в энергетике/ Л.В. Массель, А.Г. Массель// OSTIS, 2014, С. 111-116

[Даль, 1975] Толковый словарь русского языка/Даль. В. – М.: Советская энциклопедия, 1975

[Кольцов и др., 2005] Кольцов А.С., Федорков Е.Д. Технология программирования: Учеб. пособие. Воронеж: Воронеж. гос. техн. ун-т, 2005. -272 с.

[Рамбо и др., 2002] Рамбо Дж., Якобсон А., Буч Г. UML: специальный справочник– СПб.: Питер, 2002. - 656 с.

[Грибова и др., 2013] Грибова В.В., Клещев А.С Облачные семантические технологии проектирования интеллектуальных сервисов/ В.В. Грибова, А.С. Клещев// OSIS, 2013, С. 25-30

[Загорулько, 2013] Загорулько Ю.А. Технологии разработки интеллектуальных систем, основанные на интегрированной модели представления знаний/ Ю.А. Загорулько// OSIS, 2013, С. 31-42

[Голенков и др., 2013] Голенков В.В., Гулякина Н.А. Открытый проект, направленный на создание технологии компонентного проектирования интеллектуальных систем/ В.В. Голенков, Н.А. Гулякина// OSIS, 2013, С.55-77

[Новиков и др., 2009] Новиков Ф. А., Тихонова У. Н. Автоматный метод определения проблемно-ориентированных языков (часть 1)/ Ф. А. Новиков, У. Н. Тихонова // Информационно-управляющие системы, 2009, №6, С.34-40.

[Джарратано и др., 2006] Джарратано Дж., Райли Г. Экспертные системы: принципы разработки и программирование : Пер. с англ./ Дж. Джарратано [и др.]; — М. : Вильямс, 2006, С.779—851

[Зюзьков, 2003] Зюзьков В.М. Математическое введение в декларативное программирование/ В.М. Зюзьков// 2003. - 83 с.

[Васильев и др., 2000] Васильев С.Н., Жерлов А.К., Федосов Е.А., Федунов Б.Е. Интеллектуальное управление динамическими системами : монография/ С.Н. Васильев [и др.]; — М.: Физико-математическая литература, 2000. - 352 с.

[Трофимов, 2003] Трофимов С. Рабочие процессы RUP и диаграммы UML/ С. Трофимов // http://www.caseclub.ru/, 2003

[Object Management Group, 2014] UML 2.4.1 / Object Management Group//http://www.omg.org/spec/UML/2.4.1/, 2014

[Иващенко и др., 2004] Иващенко А.В., Сталькин А.А., Калышенко У.М. Применение методологии UML при автоматизации управления бизнес-процессами/ А.В. Иващенко, А.А. Сталькин, У.М. Калышенко //Исследовано в России : эл. журнал СамарскогоГАУ http://zhurnal.ape.relarn.ru/articles/2004/, 2004

[Медведев, 2005] Медведев В.И. Программирование на C++,C++.NET/C#:учеб. пособие.- Казань: Мастер Лайн, 2005. - 270 с.

TECHNOLOGY OF AUTOMATED SYNTHESIS THE INFORMATION SYSTEMS USING SEMANTIC MODELS OF SUBJECT AREA

Bikmullina I.I.

Kazan National Research Technical University named after A. N. Tupolev (KAI), Kazan, Russia

elsiyar-b@yandex.ru

The paper considers the problem of the lack of automated structural synthesis of information systems based on the semantic relations of subject area. We propose an artificially intelligent approach to automated information systems development, improvement of modern technology at the expense of automated synthesis the UML class diagrams on based on semantic models.

Introduction

Creating the application software (software) is a complex process. The most critical part of him is the directly programming the application, so as on this stage used the mental image the concepts of the application subject area in the programming language, and this requires knowledge as of the language a subject area, as the programming language. The problem of the study is the need to automate the development process (synthesis) class diagrams UML. Therefore, the significant objective of this study is improving the quality of the design study of information system on account of automating the process of synthesis of structural models. From the set of structural models of information technology chosen model class diagram.

Conclusion

The paper presents the improvement of modern information systems development technology through the automated synthesis of UML class diagrams based on semantic models.

Automation of synthesis of a class diagram makes it possible to reduce the complexity of the development of structural models of information systems, improve the quality of software project due to the possibility of multivariate analysis of structural models, allow us to stimulate the developer on preliminary thinking through study the structure of the system , allow us to change the nature labor of the programmer in the direction from the manual study of UML diagrams in the direction to the choice of the optimal variant from the sets automatically received options.