

# Программная инженерия

Пр 5  
2011  
ИН

Учредитель: Издательство "НОВЫЕ ТЕХНОЛОГИИ"

Издается с сентября 2010 г.

Главный редактор  
ГУРИЕВ М.А.

**Редакционная коллегия:**

АВДОШИН С.М.  
АНТОНОВ Б.И.  
БОСОВ А.В.  
ВАСЕНИН В.А.  
ГАВРИЛОВ А.В.  
ДЗЕГЕЛЁНОК И.И.  
ЖУКОВ И.Ю.  
КОРНЕЕВ В.В.  
КОСТЮХИН К.А.  
ЛИПАЕВ В.В.  
ЛОКАЕВ А.С.  
МАХОРТОВ С.Д.  
НАЗИРОВ Р.Р.  
НЕЧАЕВ В.В.  
НОВИКОВ Е.С.  
НОРЕНКОВ И.П.  
НУРМИНСКИЙ Е.А.  
ПАВЛОВ В.Л.  
ПАЛЬЧУНОВ Д.Е.  
ПОЗИН Б.А.  
РУСАКОВ С.Г.  
РЯБОВ Г.Г.  
СОРОКИН А.В.  
ТЕРЕХОВ А.Н.  
ТРУСОВ Б.Г.  
ФИЛИМОНОВ Н.Б.  
ШУНДЕЕВ А.С.  
ЯЗОВ Ю.К.

**Редакция:**

ЛЫСЕНКО А.В.  
ЧУГУНОВА А.В.

## СОДЕРЖАНИЕ

<b>Кухаренко Б. Г.</b> Принцип открытости-закрытости в программной инженерии и паттерны проектирования. Часть 2 .....	2
<b>Колоденкова А. Е.</b> Оценка жизнеспособности программных проектов в условиях нечеткости исходных данных .....	10
<b>Шокуров А. В.</b> К созданию программных средств кодирования растровых изображений изменяемой детализации для их адаптивного интерактивного анализа .....	17
<b>Язов Ю. К., Соловьев С. В., Ступников А. В.</b> Некоторые аспекты обеспечения совместного функционирования прикладных систем поддержки принятия решений с системами электронного документооборота .....	29
<b>Инюхин А. В.</b> Механизмы аутентификации Grid в web-приложениях и сервисах .....	35
<b>Долинина О. Н.</b> Метод генерации тестов для отладки баз знаний экспертных систем .....	40
<b>Contents</b> .....	48

Журнал зарегистрирован  
в Федеральной службе  
по надзору в сфере связи,  
информационных технологий  
и массовых коммуникаций.

Свидетельство о регистрации

ПИ № ФС77-38590 от 24 декабря 2009 г.

Журнал распространяется по подписке, которую можно оформить в любом почтовом отделении (индексы: по каталогу агентства "Роспечать" — 22765, по Объединенному каталогу "Пресса России" — 39795) или непосредственно в редакции.  
Тел.: (499) 269-53-97. Факс: (499) 269-55-10.  
[Http://novtex.ru](http://novtex.ru) E-mail: [prin@novtex.ru](mailto:prin@novtex.ru)

Журнал включен в систему Российского индекса научного цитирования. Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

© Издательство "Новые технологии", "Программная инженерия", 2011

## Принцип открытости-закрытости в программной инженерии и паттерны проектирования. Часть 2\*

*Показано как принцип открытости-закрытости проявляется в объектно-ориентированном программировании на уровне микроархитектуры программных систем. Паттерны проектирования представляют иерархии классов, которые формируют общее решение задачи проектирования программных систем. Рассматриваются методы обнаружения в программных системах модифицированных версий паттернов, которые отличаются от их стандартного представления дополнительными уровнями наследования. Эффективность используемого метода подсчета сходства графов компонентов программных систем и графов паттернов демонстрируется при обнаружении канонических и демонстративных примеров паттернов на языке Java.*

**Ключевые слова:** объектно-ориентированное программирование, принцип открытости-закрытости, микроархитектура, паттерны проектирования, обнаружение паттернов, алгоритмы графов

### Обнаружение паттернов проектирования

#### Автоматическое обнаружение паттернов проектирования

Первая попытка автоматического обнаружения паттернов описана в работе [1]. В этой работе код Smalltalk подвергается обратной инженерии для обнаружения четырех хорошо известных паттернов из каталога [2]. Алгоритм основан на информации из иерархии классов, отношений ассоциации и агрегирования, также из сообщений, передаваемых между классами. В работе [3] описана программная система, способная идентифицировать несколько паттернов проектирования, представленных в исходном коде на C++. ОМТ-диаграммы классов, представляющие паттерны, инспектируются, чтобы построить правила языка Prolog, способствующие распознаванию паттернов. Такой подход требует определения новых правил языка Prolog для нового паттерна проектирования, который должен обнаруживаться. Согласно работе [4], чтобы эффективно обнаруживать паттерны проектирования, представленные в программных

системах, желательна интеллектуальная комбинация статического и динамического анализа. В терминах UML-нотации это требует анализа диаграмм класса, чтобы вскрывать статическую информацию, и исследования последовательности диаграмм взаимосвязи (*collaboration diagrams*) для динамической информации. В работе [5] впервые применен статический анализ для получения набора кандидатов образцов паттернов и последующего выполнения динамического анализа этого набора. Однако этот подход сильно зависит от характеристик каждого паттерна: для каждого нового паттерна требуется специфический алгоритм вычисления статических кандидатов и установления правил, позволяющих динамический анализ. Это препятствует развитию расширяемой методологии автоматического обнаружения паттернов. В работе [6] описан метод идентификации структурных паттернов в программной системе для исследования того, насколько полезным может быть средство обнаружения паттернов для понимания и модификации программ. На первом шаге используется метрика для идентификации возможных кандидатов в паттерны, а на втором шаге ограничения кратчайших путей (*shortest path constraints*) генерируются из кратчайших путей между ролями в этом паттерне. Наконец, для некоторых паттернов, которым важны вызовы мето-

\* Часть 1 опубликована в журнале "Программная инженерия" № 4, 2011.

дов, генерируются ограничения делегирования (*delegation constraints*). Эти три шага метода обнаружения паттернов позволяют сократить пространство поиска. Окончательные образцы паттернов извлекаются на основе структурной информации. Метод тестировался на программных системах общего назначения от небольшого до среднего размера. Главным недостатком этого подхода (это отмечали сами авторы), является низкая точность (много ложных обнаружений). В работе [7] используется схема обратной инженерии (*reverse engineering framework*) Columbus, описанная в работе [8], для извлечения абстрактного семантического графа (*abstract semantic graph*) и язык разметки паттернов проектирования (*Design Pattern Markup Language* — DPML) для описания характеристик ролей паттерна. Алгоритм разведывания паттернов (*pattern mining algorithm*) пытается совместить роли, представленные в DPML-файлах с классами в абстрактных семантических графах. Пространство поиска сокращается за счет фильтрации, основанной на структурной информации. Этот метод тестировался на четырех программных системах общего назначения от среднего до большого размера. Исследование показывает, что чем больше упрощается описание паттерна, тем больше появляется ложных обнаружений. Поскольку алгоритм выполняет точное совмещение, остается открытым вопрос, может ли этот подход идентифицировать модифицированные версии паттернов. В другом подходе, предложенном в работе [9], используется графический формат как промежуточное представление. Паттерны проектирования выражаются в терминах визуальных грамматик (*visual grammars*), затем строится библиотека паттернов проектирования. Паттерны проектирования можно обнаружить в исследуемой программной системе, используя метод разбора визуального языка (*visual language parsing technique*) и одновременно сравнивая результаты разбора с существующей библиотекой. Главное преимущество этого подхода состоит в том, что процесс обнаружения может непосредственно визуализироваться, однако этот подход не может быть оценен на реальных программных системах, поскольку это средство не интегрируется с существующим исходным кодом к средствам извлечения диаграмм классов (*class-diagram extractors*). Все упомянутые выше методы не способны обнаруживать модифицированные версии паттернов, т. е. отклоняющиеся от их стандартного представления. Это накладывает серьезное ограничение на применимость этих методов к реальным программным системам.

В работе [10] представлен метод, который исследует UML-диаграммы, предлагает модификации микроархитектуры программной системы и приводит к паттернам проектирования. Частью этого процесса является автоматическое обнаружение паттернов проектирования в программной системе. Входом процесса является UML-проект (диаграммы классов и их взаимодействия — *class and collaboration diagrams*) программной системы в XML-формате (*XML Metadata*

*Interchange*). Статический и динамический анализ выполняется с использованием базы знаний, состоящей из правил языка Prolog, которые описывают главные характеристики паттернов проектирования для получения окончательного набора образцов паттернов. Для введения новых паттернов проектирования должны быть составлены новые правила языка Prolog. Более того, авторы не публикуют результаты оценки для реальных программных систем. Метод обнаружения паттернов проектирования через так называемый "отпечаток" — "*fingerprinting*" был предложен в работе [11]. Этот подход сокращает пространство поиска посредством идентификации классов, играющих определенную роль в мотивах проектирования (*design motifs*), используя метрики, основанные на их внешних атрибутах. На следующем шаге фактические реализации паттернов находятся с использованием структурного совмещения. Эффективность такого алгоритма существенно зависит от обучающих образцов (*learning samples*), которые составляют репозиторий ролей мотивов проектирования (*design motif roles*). В работе [12] предложен метод, относительно которого утверждается, что он идентифицирует модифицированные версии паттернов проектирования. Эта программная система "PTIDEJ" исследует задачу обнаружения паттернов как задачу удовлетворения ограничениям. Она формулируется посредством исследования абстрактной модели паттерна и исследуемого исходного кода. Набор переменных также как набор ограничений для переменных выводятся из абстрактной модели паттерна, в то время как исследуемой областью являются сущности, представленные в исходном коде исследуемой программной системы. Средство PALM используется, чтобы идентифицировать на уровне микроархитектуры (*microarchitecture*) исходного кода те стандартные компоненты, архитектура которых идентична или похожа на микроархитектуру паттерна проектирования. Основным недостатком этого подхода в том, что для обнаружения нового паттерна, в это средство должна быть вложена новая абстрактная модель (для задачи удовлетворения ограничениям (*constraint satisfaction problem*)). В работе [13] использован анализ понятий (*concept analysis*), основанный на отношениях между классами (*class relationships*). Их приложение не использует какую-либо базу знаний представлений паттернов проектирования. Паттерны проектирования, представленные в системе, выводятся непосредственно из исследуемой программной системы через поиск рекуррентных групп классов (*recurrent groups of classes*). Преимущество этого подхода в том, что он легко расширяется, поскольку легко обнаруживаются новые паттерны проектирования. Одним из недостатков этого подхода является его вычислительная сложность (*computational complexity*), которая сокращается посредством рассматривания контекста порядка выше трех классов. Это означает, что последовательность классов длиной, больше чем три, рассматривается как строящая (создающая) некоторое понятие. Другой подход к автоматическому обнаружению паттернов

проектирования представлен в работе [14]. Он основан на понятии элементарных паттернов проектирования (*elemental design patterns*). Элементарные паттерны проектирования (*elemental design patterns*) из работы [15] — это базовые понятия, на основе которых строятся более сложные паттерны проектирования. Главной силой подхода, основанного на понятии элементарных паттернов проектирования, является способность обнаруживать паттерн после примененного к нему рефакторинга (*refactoring*) [16]. На первом уровне идентифицируются элементарные паттерны проектирования, а на втором уровне найденные элементарные паттерны проектирования комбинируются, чтобы идентифицировать фактические паттерны проектирования. Для того чтобы непосредственно представлять отношения между объектами, методами и полями используется формальный язык, называемый Rho-Calculus [17–18]. Тот же самый язык используется для формализации как паттернов проектирования, так и исследуемой программной системы. Далее используется автоматическое средство доказательства теорем (*automated theorem prover*), чтобы обнаружить образцы паттернов в программной системе. Однако из работы [14] не ясно, какая эвристика используется для комбинирования существующих (имеющихся) предикатов, чтобы достичь этот результат. Очевидно, что когда никаких эвристик не применяется, вычислительная сложность исследования всех возможных комбинаций является чрезмерной. Применимость этого метода представляется с иллюстрацией шагов, требуемых для обнаружения паттерна Decorator в небольшой созданной автором программной системе. В работе [19] делается попытка найти связь между присутствием конкретных паттернов проектирования в программном обеспечении и числом дефектов. Средство обратной инженерии (*reverse engineering tool*) "Understand for C++" выполняет разбор исходного кода и производит структурные метаданные, которые хранятся в базе данных. Затем через запросы в базу данных обнаруживаются паттерны, которые соответствуют заданной структурной сигнатуре для каждого паттерна [20]. Отмена (несколько ложных обнаружений) и точность (несколько ложных пропусков) являются приемлемыми. Этот метод проверялся на большой коммерческой системе. Отмена (аннулирование) оценивалась на случайном образце классов с использованием статистического анализа. Наконец, в работе [21] как исследуемая программная система, так и паттерн, который предстоит обнаружить, описываются в терминах графа. Для обнаружения паттерна используется алгоритм подсчета, представленный в работе [22], который в качестве входа использует как граф программной системы, так и граф паттерна, и вычисляет сходство между их вершинами. Главное преимущество этого подхода состоит в способности обнаруживать не только паттерны в их базовой форме, но также их модифицированные версии (при условии, что модификация ограничена одной характеристикой

паттерна). Этот подход к обнаружению паттернов проектирования рассматривается ниже.

### Алгоритм вычисления сходства

В работе [22] сформулирован итеративный алгоритм для вычисления сходства между вершинами двух графов. Пусть  $G_A$  и  $G_B$  — два направленных графа, соответственно, с  $n_A$  и  $n_B$  вершинами. Матрица сходства  $S$  определяется как матрица размерности  $n_B \times n_A$ , действительные элементы  $s[i; j]$  которой выражают, насколько  $j$  в  $G_A$  сходна с вершиной  $i$  в  $G_B$ , и называются подсчетом подобия (*similarity score*) между двумя вершинами. Пусть  $A, B$ , — матрицы смежности графов  $G_A$  и  $G_B$ , соответственно:  $Z_0$  — матрица размерности  $n_B \times n_A$ , заполненная единицами:  $\|\cdot\|_1$  — одномерная норма матрицы и сходимость алгоритма относится к последовательности четных итераций. Алгоритм, используемый для вычисления матрицы сходства  $S$ , показан на рис. 1.

В контексте обнаружения паттернов, алгоритм на рис. 1 может использоваться для вычисления сходства между вершинами графа, описывающего паттерн ( $G_A$ ) и соответствующего графа, описывающего программную систему ( $G_B$ ). Это приводит к нескольким матрицам сходства размерности  $n_B \times n_A$  (одной для каждого типа представленной информации). Чтобы получить полную картину сходства между паттерном и программной системой, следует использовать информацию, предоставляемую всеми матрицами сходства. Для обеспечения обоснованности результатов все подсчеты сходства должны быть в пределах интервала  $[0, 1]$ . Следовательно, индивидуальные матрицы первоначально суммируются, и результирующая матрица нормируется делением элементов столбца  $i$  (соответствующего подсчетам сходства между всеми классами программной системы и ролью паттерна (*pattern role*) на число матриц ( $k_i$ ), в которые вовлечена данная роль паттерна (*given role*)). Это эквивалентно применению аффинного преобразования, при котором результирующая матрица умножается на квадратную диагональную матрицу размерности  $n_A \times n_A$ , где элемент  $[i; i]$  равен  $1/k_i$ .

Другим подходом к идентификации образцов графа паттерна в графе программной системы является применение алгоритмов совмещения графов [23], которые разделяются на две главные категории [24].

$S = \text{Similarity}(A, B)$

1. Установить  $Z_0 = 1$ .

2. Итерировать четное число раз

$$Z_{k+1} = \frac{BZ_k A^T + B^T Z_k A}{\|BZ_k A^T + B^T Z_k A\|}$$

и остановиться при достижении сходимости.

3. Выход  $S$  — это последнее значение  $Z_k$ .

Рис. 1. Алгоритм вычисления матрицы сходства

1. Точные алгоритмы совмещения графов, для которых эта задача состоит в поиске взаимно однозначного отображения (изоморфизм — *isomorphism*) между вершинами двух графов, имеющих одно и то же число вершин, так что существует также взаимно однозначное соответствие между дугами графов. В контексте обнаружения паттернов проектирования применение такого алгоритма потребовало бы удаления всех возможных подграфов графа программной системы, которые имеют одинаковое число вершин с паттерном. Наиболее существенным недостатком, однако, является то, что конкретный паттерн проектирования может быть реализован в различных формах, которые отличаются от базовой структуры, представленной в литературе, и как результат точное совмещение оказывается неудовлетворительным для обнаружения паттернов проектирования.

2. Алгоритмы неточного совмещения графов, которые применяются, когда изоморфизм между двумя графами не может быть найден, и имеют целью нахождение наилучшего совмещения между обоими графами. Как пример, имеются алгоритмы, которые вычисляют в качестве меры расстояния меру редактирования (*edit distance*) между двумя графами, определяемую как число модификаций, которое следует выполнить, чтобы получить из одного графа другой граф [25]. Такие алгоритмы обнаружения паттернов проектирования могут привести к неточным результатам. Ниже это иллюстрируется на примере.

Рассмотрим пример применения алгоритма вычисления сходства (см. рис. 1), приведенный в работе [21]. Пусть исследуемая программная система имеет два сегмента, представленных соответствующими диаграммами классов (рис. 2). Паттерн проектирования, который должен быть обнаружен, также представлен графически на рис. 2. Этот паттерн известен как элементарный паттерн проектирования *RedirectInFamily* из работы [15]. Он формирует паттерны проектирования *Composite* и *Decorator* [2]. Очевидно, диаграмма класса

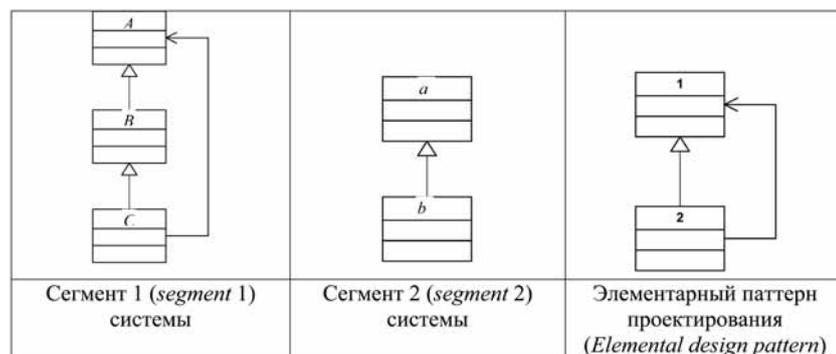


Рис. 2. UML-диаграмма классов для двух сегментов программной системы и паттерна проектирования

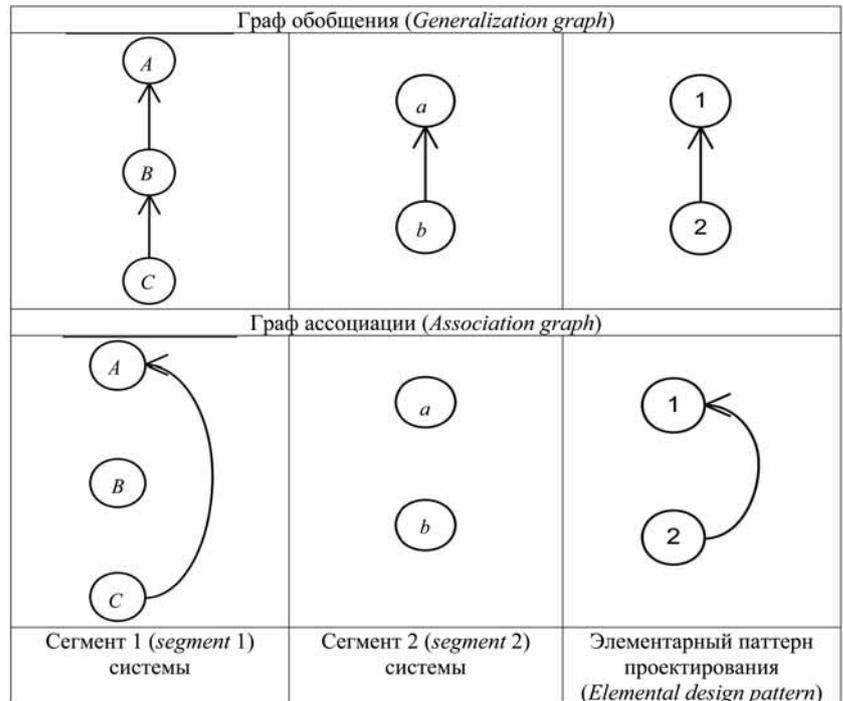


Рис. 3. Графы для UML-диаграмм на рис. 2 (символы в узлах показывают имя соответствующего узла)

сегмента 1 (*segment 1*) является модифицированной версией паттерна проектирования, содержащей дополнительный уровень наследования (*inheritance level*). Вместе с тем, диаграмма классов сегмента, (*segment 2*) формирует паттерн, поскольку она состоит только из простой иерархии классов. На рис. 3 представлены эти диаграммы классов как графы (по одному для ассоциаций (*associations*) и по одному для обобщений (*generalizations*)).

Алгоритм неточного совмещения графов (*inexact matching algorithm*), рассматривающий в качестве расстояния меру редактирования (*edit distance measure*) заключил бы, что диаграмма класса сегмента 2 похожа на этот элементарный паттерн. Дело в том, что чтобы получить граф элементарного паттерна из соответствующего графа сегмента 2, требуется только одна операция редактирования (добавление одной дуги в графе ассоциаций (*association graph*) между дугами *b* и *a*).

Вместе с тем, чтобы получить граф паттерна из соответствующего графа сегмента 1, в общем, требуется пять операций редактирования (граф обобщения (*generalization graph*): удаление дуг (*B, A*) и (*C, B*), удаление узла *B*, добавление дуги (*C, A*), граф ассоциации (*association graph*): удаление узла *B*). Следовательно, всякое отношение обобщения между двумя классами будет рассматриваться как сильный (предпочтительный) кандидат на паттерн, в то время как модифицированная версия сегмента 1

(*modified version of segment 1*) рассматривается как довольно слабый кандидат. Вместе с тем, алгоритм вычисления сходства (рис. 1) дает более точные результаты для этого примера. На рис. 4 показаны соответствующие матрицы смежности графов, представленных на рис. 3.

Матрицы сходства между соответствующими графами сегмента 2 и элементарным паттерном имеют следующий вид (где функция сходства *Similarity* (...)) соответствует алгоритму вычисления матрицы сходства *S* графов):

$$\begin{aligned} Gen_{pattern,seg2} &= \\ &= Similarity(Gen_{pattern}, Gen_{seg2}) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \\ & Assoc_{pattern,seg2} = \\ Similarity(Assoc_{pattern}, Assoc_{seg2}) &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}. \end{aligned}$$

Сумма двух этих матриц имеет вид

$$\begin{aligned} Sum_{pattern,seg2} &= Gen_{pattern,seg2} + \\ &+ Assoc_{pattern,seg2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \end{aligned}$$

В то же время нормированные подсчеты (*normalized scores*), которые в итоге будут выдвигать на передний план похожие узлы, вычисляются следующим образом:

$$\begin{aligned} NormScores_{pattern,seg2} &= Sum_{pattern,seg2} \begin{bmatrix} 1/k_1 & 0 \\ 0 & 1/k_2 \end{bmatrix} = \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} = \\ &= \begin{matrix} 1 & 2 \\ a & b \end{matrix} \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix}, \end{aligned}$$

где  $k_1$  и  $k_2$  соответствуют числу матриц, в которых, соответственно, включены роли паттерна 1 и 2 (*pattern roles 1 and 2*) (в этом случае, обе роли включены в матрицу ассоциации и в матрицу обобщения (*association and the generalization matrix*)). Вместе с тем, матрицы сходства между соответствующими графами сегмента 1 и элементарного паттерна имеют вид

$$\begin{aligned} Gen_{pattern,seg1} &= \\ &= Similarity(Gen_{pattern}, Gen_{seg1}) = \begin{bmatrix} 1/2 & 0 \\ 1/2 & 1/2 \\ 0 & 1/2 \end{bmatrix}, \end{aligned}$$

Матрицы обобщения ( <i>Generalization matrices</i> )		
$Gen_{seg1} = \begin{matrix} & A & B & C \\ A & \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \\ B & \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ C & \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \end{matrix}$	$Gen_{seg2} = \begin{matrix} & a & b \\ a & \begin{bmatrix} 0 & 0 \end{bmatrix} \\ b & \begin{bmatrix} 1 & 0 \end{bmatrix} \end{matrix}$	$Gen_{pattern} = \begin{matrix} & 1 & 2 \\ 1 & \begin{bmatrix} 0 & 0 \end{bmatrix} \\ 2 & \begin{bmatrix} 1 & 0 \end{bmatrix} \end{matrix}$
Матрицы ассоциаций ( <i>Association matrices</i> )		
$Assoc_{seg1} = \begin{matrix} & A & B & C \\ A & \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \\ B & \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \\ C & \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \end{matrix}$	$Assoc_{seg2} = \begin{matrix} & a & b \\ a & \begin{bmatrix} 0 & 0 \end{bmatrix} \\ b & \begin{bmatrix} 0 & 0 \end{bmatrix} \end{matrix}$	$Assoc_{pattern} = \begin{matrix} & 1 & 2 \\ 1 & \begin{bmatrix} 0 & 0 \end{bmatrix} \\ 2 & \begin{bmatrix} 1 & 0 \end{bmatrix} \end{matrix}$
Сегмент 1 ( <i>segment 1</i> ) системы	Сегмент 2 ( <i>segment 2</i> ) системы	Элементарный паттерн проектирования ( <i>Elemental design pattern</i> )

Рис. 4. Матрицы смежности для графов на рис. 3

$$Assoc_{pattern,seg1} = Similarity(Assoc_{pattern}, Assoc_1) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix},$$

$$NormScores_{pattern,seg1} =$$

$$(Gen_{pattern,seg1} + Assoc_{pattern,seg1}) \cdot \begin{bmatrix} 1/k_1 & 0 \\ 0 & 1/k_2 \end{bmatrix} =$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix} =$$

$$= \begin{matrix} 1 & 2 \\ A & B \\ 3/4 & 1/4 \\ 0 & 3/4 \end{matrix}.$$

Наибольшие элементы последней матрицы показывают сильное сходство между классами (*A, 1*) и (*C, 2*) соответствующих UML-диаграмм для сегмента 1 и элементарного паттерна (см. рис. 2). В противоположность результатам, полученным посредством алгоритма неточного совмещения графов (*inexact matching algorithm*), который показывает, что элементарный паттерн гораздо ближе к структуре сегмента 2, алгоритм вычисления сходства (см. рис. 1) точно идентифицирует паттерн, который реализован в структуре сегмента 1. Матрица сходства  $NormScores_{pattern,seg2}$  также показывает сходство между классами (*a, 1*) и (*b, 2*), которые являются приемлемыми, поскольку матрицы обобщения сегмента 2 и элементарного паттерна на рис. 4 являются теми же самыми, но сила сходства ниже вследствие различия в их матрицах ассоциаций.

Полная методология обнаружения образцов паттернов проектирования в программных системах с учетом множественного наследования классов (*multiple inheritance*) описана в работе [21]. Разработан Java-па-

кет Design Pattern detection v.4.5, который автоматизирует эту методологию и генерирует список обнаруженных образцов паттернов проектирования. Этот пакет использует схему манипулирования Java-байткодом (*Java bytecode manipulation framework*), которая обеспечивает детальный анализ статической структуры программной системы [26]. Извлекаемая пакетом информация:

- абстракция (*abstraction*) — является ли класс конкретным (*concrete*), абстрактным (*abstract*) или интерфейсом (*interface*);
- наследование (*inheritance*) — родительский класс (*parent class*), реализованные интерфейсы (*implemented interfaces*);
- атрибуты класса (*class attributes*) — тип (*type*), видимость (*visibility*) и статические члены (*static members*);
- сигнатуры конструкторов (*constructor signatures*) — типы параметров (*parameter types*);
- сигнатуры методов (*method signatures*) — имя метода (*method name*), возвращаемый тип (*return type*), типы параметров, абстрактные или нет (*parameter types, abstract or not*);
- вызовы методов (*method invocations*) — класс-источник и сигнатура (*origin class and signature*), а также образцы объектов (*object instantiations*).

Примеры обнаружения образцов паттернов проектирования в программных системах описаны в работе [27].

#### Обнаружение паттернов посредством пакета Design Pattern detection v.4.5

В качестве первого примера рассматривается Java-код паттернов из коллекции Huston Design Pattern [28]. Результат обнаружения паттернов посредством пакета Design Pattern detection v.4.5 после анализа байткода классов, представленных на рис. 8 части 1 данной статьи (см. журнал "Программная инженерия" № 4, 2011), показан на рис. 5. Результат обнаружения паттернов после анализа байткода классов, представленных на рис. 12 части 1 данной статьи (см. журнал "Программная инженерия" № 4, 2011), показан на рис. 6. Результат обнаружения паттернов в примерах на языке Java из Huston Design Pattern [28] показан на рис. 7 (используются сокращенные обозначения паттернов проектирования из таблицы, см. часть 1 данной статьи, журнал "Программная инженерия" № 4, 2011.).

В качестве второго примера рассматриваются реализации паттернов на языке Java из работ [29, 30]. Аналогичные примеры реализации паттернов на C# представлены в работе [31]. Автор этих книг J. W. Соорет

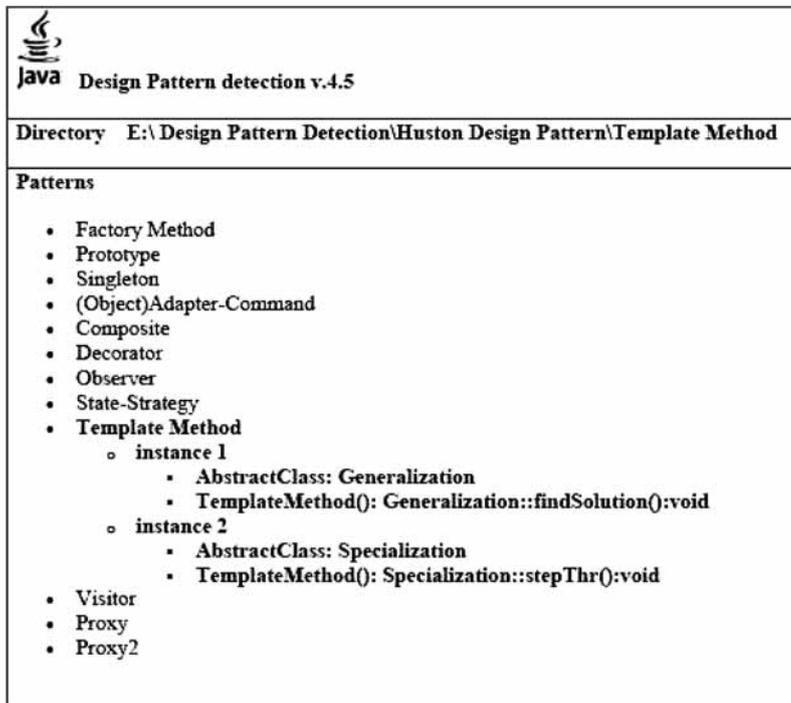


Рис. 5. Обнаружение паттерна Template Method

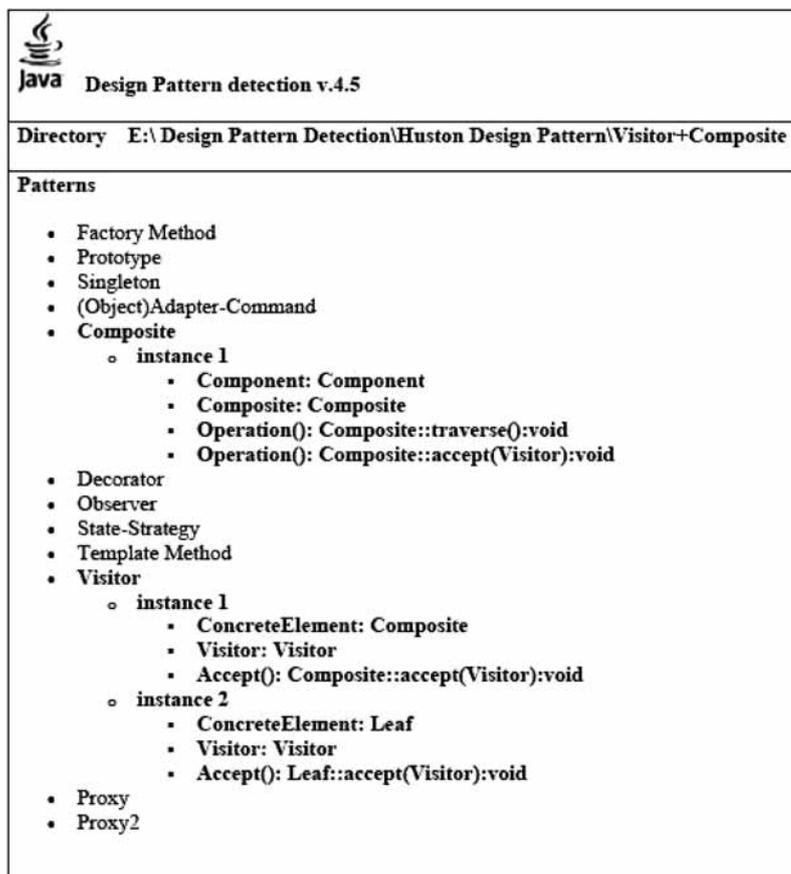


Рис. 6. Обнаружение паттернов Visitor + Composite

	A	CD	AF	BR	BU	CR	CP	D	FA	FM	FL	IN	IT	MD	MM	O	PT	PX	S	ST	SR	TM	V
A	+																				+		
AF																							
BR																							
BU																							
CD																							
CR																							
CP							+																
D								+															
FA																							
FM										+													+
FL																							
IN																							
IT																							
MD																							
MM																							
O																+							
PT																	+						
PX																		+					
S																			+				
ST																					+		
SR+TM																					+	+	
TM																						+	+
V+CP							+																+

Рис. 7. Результат обнаружения паттернов проектирования в примерах на языке Java из коллекции Huston Design Pattern [28]

	A	CD	AF	BR	BU	CR	CP	D	FA	FM	FL	IN	IT	MD	MM	O	PT	PX	S	ST	SR	TM	V
A	+																						
AF																							
BR	+																						
BU																							
CR	+						+																
CD	+																				+		
CP																							
D																							
FA																							
FM																							
FL	+																						
IN	+																						
IT																							
MD	+																				+		
MM	+																						
O																							
PT	+																						
PX																							
S																							
ST	+																						
SR	+																				+		
TM																						+	+
V																							+

Рис. 8. Обнаружение паттернов проектирования в примерах из работ [29, 30]

является научным сотрудником в Отделе передового поиска и анализа информации (*Advanced Information Retrieval and Analysis Department*) в Исследовательском центре им. Томаса Джона Ватсона компании IBM (*IBM Thomas J. Watson Research Center*). Результат обнаружения паттернов в примерах из работ [29, 30] показан на рис. 8.

Как видно на рис. 7 и 8, паттерны проектирования *Template Method* (Шаблонный метод) и *Visitor* (Посетитель), выбранные в качестве примера в настоящей статье, одинаково интерпретируются различными авторами.

### Список литературы

1. **Brown K.** Design Reverse-Engineering and Automated Design Pattern Detection in Smalltalk. Technical Report TR-96-07. Department of Computer Science, North Carolina State University, 1996.
2. **Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж.** Приемы объектно-ориентированного проектирования. Паттерны проектирования. СПб.: Питер, 2001.
3. **Prechelt L., Kramer C.** Functionality versus Practicality: Employing Existing Tools for Recovering Structural Design Patterns // *Journal of Universal Computer Science*. 1998. V. 4, No. 12. P. 866–882.
4. **Wendehals L.** Specifying Patterns for Dynamic Pattern Instance Recognition with UML 2.0 Sequence Diagrams // *Proc. of Sixth Workshop on Software Reengineering (WSR'04)*. 2004, May. P. 63–64.
5. **Heuzeroth D., Holl T., Hogstrom G., Lowe W.** Automatic Design Pattern Detection // *Proc. of 11th IEEE International Workshop Program Comprehension (IWPC'03)*. May 2003. P. 94–103.
6. **Antoniol G., Casazza G., Di Penta M., Fiutem R.** Object-Oriented Design Patterns Recovery // *Journal of Systems and Software*. 2001. V. 59, No. 2. P. 181–196.
7. **Balanyi Z., Ferenc R.** Mining Design Patterns from C++ Source Code // *Proc. of International Conf. on Software Maintenance (ICSM'03)*. 2003. Sept. P. 305–314.
8. **Vidacs L., Beszedes A., Ferenc R.** Columbus Schema for C/C++ Preprocessing // *Proc. of the 8th European Conf. on Software Maintenance and Reengineering (CSMR 2004)*. Tampere. Finland., 2004. P. 75–84.
9. **Costagliola G., De Lucia A., Deufemia V., Gravino C., Risi M.** Design Pattern Recovery by Visual Language Parsing // *Proc. of Ninth European Conf. on Software Maintenance and Reengineering (CSMR'05)*. Mar. 2005. P. 102–111.
10. **Bergenti F., Poggi A.** Improving UML Designs Using Automatic Design Pattern Detection // *Proc. of 12th International Conf. on Software Engineering and Knowledge Engineering (SEKE'00)*. July 2000. P. 336–343.
11. **Gueheneuc Y.-G., Sahraoui H., Zaidi F.** Fingerprinting Design Patterns // *Proc. of 11th Working Conf. on Reverse Engineering (WCRE'04)*. 2004. Nov. P. 172–181.
12. **Albin-Amiot H., Cointre R., Gueheneuc Y.-G., Jussien N.** Instantiating and Detecting Design Patterns: Putting Bits and Pieces Together // *Proc. of 16th Annual Conf. on Automated Software Engineering (ASE'01)*. 2001. Nov. P. 166–173.
13. **Tonella P., Antoniol G.** Object Oriented Design Pattern Inference // *Proc.s of IEEE Conf. on Software Maintenance (ICSM'99)*. 1999. P. 230–238.
14. **Smith J. M., Stotts D.** SPQR: Flexible Automated Design Pattern Extraction from Source Code // *Proc. of 18th IEEE International Conf. on Automated Software Engineering (ASE'03)*. 2003. Oct. P. 215–224.
15. **Smith J. M.** An Elemental Design Pattern Catalog. Technical Report TR-02-040. Department of Computer Science, University of North Carolina. 2002. Oct.
16. **Fowler M.** Refactoring: Improving the Design of Existing Code. Addison Wesley, 1999.
17. **Cirstea H., Faure G., Kirchner C.** A rho-Calculus of Explicit Constraint Application // *Proc. of Conf. Electronic Notes in Theoretical Computer Science — ENTCS*. 2005. V. 117. P. 51–67.
18. **Cirstea H., Liquori L., Wack B.** Rewriting calculus with fixpoints: Untyped and first-order systems // *Proc. of International Conf. Types for Proofs and Programs, (TYPES 2008)*. Torino, Italy. 2008. March 26–29. Revised Selected Papers. P. 147–161.
19. **Vokac M.** Defect Frequency and Design Patterns: An Empirical Study of Industrial Code // *IEEE Transactions on Software Engineering*. Dec. 2004. V. 30, No. 12. P. 904–917.
20. **Vokac M.** An Efficient Tool for Recovering Design Patterns from C++ Code // *Journal of Object Technology*. 2006. V. 5 No. 1. P. 139–157.
21. **Tsantalis N., Chatzigeorgiou A., Halkidis S. T.** Design Pattern Detection Using Similarity Scoring // *IEEE Transactions on Software Engineering*. 2006. V. 32, No 11. P. 896–909.
22. **Blondel V. D., Cajardo A., Heymans M., Senellart P., Van Dooren P.** A Measure of Similarity between Graph Vertices: Applications to Synonym Extraction and Web Searching // *SIAM Review*. 2004. V. 46, No. 4. P. 647–666.
23. **Ullman J. R.** An Algorithm for Subgraph Isomorphism // *Journal of the ACM*. Jan. 1976. V. 23, No. 1. P. 31–42.
24. **Bengoetxea E.** Inexact Graph Matching Using Estimation of Distribution Algorithms. PhD thesis. Ecole Nationale Supérieure des Telecommunications, France. Dec. 2002.
25. **Cook D. J., Holder L. B.** Substructure Discovery Using Minimum Description Length and Background Knowledge // *Journal of Artificial Intelligence Research*. 1994. Feb. Vol. 1. P. 231–255.
26. **Bruneton E., Lenglet R., Coupaye T.** ASM: a code manipulation tool to implement adaptable systems // *Adaptable and Extensible Component systems*. 2002. URL: <http://asm.objectweb.org/current/asm-eng.pdf>.
27. **Tsantalis N., Chatzigeorgiou A., Stephanides G.** Application of Graph Theory to OO Software Engineering // *Proc. of the 2006 international Workshop on interdisciplinary software engineering research (WISER'06)*. 2006. May 20. Shanghai, China. P. 29–35.
28. **Huston V.** Design Patterns in a Nutshell. Sebastopol, CA: O'Reilly & Associates, 2007.
29. **Cooper J. W.** The Design Patterns, with CD-Rom. Addison-Wesley Design Patterns Series. Addison-Wesley Professional, 1998.
30. **Cooper J. W.** Java Design Patterns: A Tutorial, with CD-Rom. Addison-Wesley Professional, 2000.
31. **Cooper J. W.** C# Design Patterns: A Tutorial, with CD-Rom. Addison-Wesley Professional, 2002.

## Оценка жизнеспособности программных проектов в условиях нечеткости исходных данных

*Рассматривается задача оценки жизнеспособности программных проектов в условиях неопределенности, порождаемой НЕ-факторами. Предлагаются нечетко-множественный и нечетко-интервальный подходы к решению данной задачи на основе сравнительного многокритериального анализа возможных альтернатив в условиях нечеткости исходных данных.*

**Ключевые слова:** программный проект, оценка жизнеспособности программного проекта, НЕ-факторы и неопределенность, нечетко-множественный и нечетко-интервальный подходы к оценке жизнеспособности программного проекта

Одним из приоритетных стратегических направлений модернизации отечественной экономики является разработка и внедрение эффективных программных проектов (ПП) во всех сферах человеческой деятельности. Здесь речь идет о сложных, наукоемких и дорогостоящих проектах в виде крупномасштабных комплексов взаимосвязанных программ на основе современных методов и средств программирования. Сложность таких проектов настолько велика, что стоимость и трудоемкость их разработки соизмеримы, а часто в несколько раз превышают стоимость и трудоемкость разработки физических, социальных и экономических систем, которые они призваны обеспечить [1].

Современная индустрия разработки ПП является развитой отраслью проектной деятельности, связанной со значительными затратами материальных, трудовых и финансовых ресурсов в условиях риска, обусловленного неопределенностью факторов внутренней и внешней среды проекта. Последние могут привести либо к ухудшению основных качественных показателей разрабатываемого ПП, либо к превышению бюджетного ассигнования и/или нарушению сроков осуществления проекта, либо просто к его провалу. В связи с этим на начальных этапах разработки ПП важная роль отводится анализу его жизнеспособности, направленному на недопущение провала проекта и снижение рисков его разработки, а также на прогнозирование его стоимости, сроков и качества разработки.

В настоящей работе рассматриваются нечетко-множественный и нечетко-интервальный подходы к оценке жизнеспособности ПП, т. е. оценке оправданности вложения в его разработку трудовых и материальных затрат на основе сравнительного многокритериального анализа возможных альтернатив в условиях нечеткой исходной информации.

### НЕ-факторы и нечеткие вычисления в задаче анализа жизнеспособности программного проекта

НЕ-факторы существовали всегда, "пронизывая" всю жизнедеятельность человека. Однако обратили на них внимание лишь при исследовании формальных аппаратов в сложных предметных областях. В задаче оценки жизнеспособности ПП НЕ-факторы играют важную роль, поскольку негативно влияют на ход выполнения проекта и, прежде всего, на планирование работ и принятие управленческих решений по проекту. Как подчеркивал А. С. Нариньяни [2], "игнорируя НЕ-факторы, мы расплачиваемся за это потерей решений или их качества, а во многих случаях и уверенности в их адекватности".

Понятие "НЕ-факторы" впервые введены А. С. Нариньяни в 1982 г. в целях обозначения комплекса свойств, характерных для человеческой системы знаний о реальном мире, но плохо представленных в формальных системах (неполнота, недоопределенность, некорректность и т. п.). Перед выбором метода

учета НЕ-факторов, характеризующих соответствующую проблемную область, необходимо содержательное их исследование.

В работе [3] дан обзор исследований в области моделирования НЕ-факторов и введена их классификация в условной трехмерной системе координат <НЕ-факторы, методы их моделирования, моделируемые объекты>.

На рис. 1 представлена предложенная в работе [4] классификация НЕ-факторов и методы их учета применительно к задаче оценки жизнеспособности ПП.

Здесь выделены следующие НЕ-факторы, порождающие неопределенность при анализе жизнеспособности ПП:

♦ *Неточность* — величина (неточное значение), которая может быть получена с точностью, не превышающей некоторый порог, определенный природой соответствующего параметра. Неточность является универсальным фактором и проявляется в тех случаях, когда разработчик в своих рассуждениях оперирует значениями каких-либо параметров (например, объем работ по проекту), полученных на основе опроса мнения группы экспертов в виде интервалов, являющихся неточным значением, задаваемым двумя более точными величинами — границами.

♦ *Нечеткость (размытость)* — величина, связанная с отсутствием точных границ соответствующего параметра. В большинстве случаев нечеткость в исходных данных возникает, когда разработчик пытается количественно охарактеризовать качественные понятия (например, "возможный объем проектных работ из интервала"), которые он использует в своих рассуждениях. Необходимо отметить, что любому измеримому параметру может соответствовать несколько нечетких значений, каждому из которых в свою очередь соответствует определенная функция принадлежности.

♦ *Неоднозначность* — величина, отражающая множество альтернатив оцениваемого параметра неравномерно с точки зрения некоторой конкретной семантики (реалистичность, желательность проекта и т. п.). Неоднозначное значение включает недоопределенный интервал и заданное на нем распределение, отражающее оценку каждого значения интервала. Заметим, что недоопределенный интервал может быть доопределен новой информацией, дополняющей знания о представляемом параметре (например объем и время выполнения работ).

В зависимости от типа неопределенности (вероятностная, интервальная, нечеткая) исходных данных о значениях располагаемых ресурсов (объем, стоимость, продолжительность работ и др.) при выполнении ПП необходимо различать класс НЕ-факторов, математический аппарат описания их модели, а также методы их учета. Заметим, что приведенное на рис. 1 разделение методов учета НЕ-факторов носит условный характер, поскольку на практике на различных этапах принятия решения они пересекаются и взаимодействуют между собой.

В последнее время в работах, связанных с управлением и обработкой информации в условиях неопределенности, все больше внимания акцентируется на проблемах моделирования НЕ-факторов в вычислительной среде. В традиционных, в частности вероятностно-статистических подходах к анализу жизнеспособности ПП (см., например, [5–7]), все неопределенности, обусловленные НЕ-факторами, трактуются в терминах случайности, т. е. в вероятностном смысле, и имеют событийно-частотную интерпретацию. Здесь неточность и нечеткость ассоциируются с распределением вероятностей, в связи с чем соответствующие модели неопределенности формулируются в терминах теории вероятности и математической статистики. Однако на практике это не всегда соответствует природе неопределенностей, часто являющихся следст-



Рис. 1. Классификация НЕ-факторов и методы их учета применительно к анализу жизнеспособности ПП

виями субъективных НЕ-факторов. Кроме того, частотные распределения будущих событий, используемые в данных подходах, известны недостаточно точно, причем наиболее типична следующая ситуация: частотные распределения некоторых хорошо изученных параметров известны и есть все основания считать, что в будущем они не изменятся; числовые значения же других параметров могут быть представлены лишь в форме нечетких или четких интервалов на основе имеющейся объективной информации и экспертных оценок.

В ряде работ обсуждается проблема совместного использования вероятностно-статистических и детерминированных (нечетких или интервальных) типов неопределенности. Очевидно, что на практике информативность (точность) результатов моделирования будет определяться точностью наиболее неопределенных параметров и поэтому искусственное преобразование интервалов в частотные распределения даст лишь иллюзию точности. Оно методологически не оправдано, поскольку не позволит оценить истинный характер неопределенностей результатов. В этой ситуации трансформация частотных распределений в интервалы может оказаться более методологически обоснованной.

Все это объясняет возрастающий интерес в последние годы к предложенным еще в 60–70-х гг. прошлого столетия *невероятностным моделям неопределенности*, обусловленной неточностью и нечеткостью информации: возможность Заде (L. A. Zadeh) и возможность Шейкла (G. L. S. Shackle), субъективная вероятность Севеджа (L. J. Savage), верхние и нижние вероятности Демпстера (A. P. Dempster), правдоподобие и доверие Шеффера (G. Shafer) и др.

Для преодоления трудностей, вызванных НЕ-факторами на ранних стадиях разработки ПП, весьма перспективно использовать провозглашенные в 1994 г. Л. Заде методы и технологии "мягких вычислений", позволяющие "устранить" неопределенности и придать задаче анализа жизнеспособности проекта количественную определенность.

*Мягкие вычисления (Soft Computing)* имитируют поведение и воспроизводят мышление человека и, в отличие от традиционных классических *жестких вычислений (Hard Computing)*, "терпимы" к НЕ-факторам и нацелены на максимальное приспособление к работе с неточными, недоопределенными или частично истинными данными. Применение в задачах анализа жизнеспособности проекта лишь жестких вычислений, особенностью которых является точность, определенность и строгость вычислительного результата, в условиях неопределенности оказывается малоэффективным или просто невозможным из-за недостаточного объема и качества исходной информации [5, 6].

Изначально концепция мягких вычислений в качестве первых и основных составляющих включает нечеткие вычисления. Кстати, следует заметить, что первые НЕ-факторы были определены и изучались именно в рамках нечеткой математики. В задачах,

связанных с анализом жизнеспособности ПП, широкое применение находят методы и технологии нечетких вычислений. В качестве примеров можно привести работу [8], в которой рассматривается *нечеткая оценка проектного риска*, а также работу [9], в которой рассматривается *нечеткая оценка длительности выполнения плановых проектных работ*.

Далее рассматриваются возможные подходы к оценке жизнеспособности ПП в условиях нечеткости исходных данных на основе объединения концепции нечетких и интервальных вычислений.

### **Нечетко-множественный подход к оценке жизнеспособности программного проекта в условиях нечеткой неопределенности**

В ситуации, когда исходные данные о времени и объемах выполнения проектных работ являются нечеткими, т. е. обладают "размытостью", предлагается использовать нечетко-множественный подход, основанный на использовании аппарата нечетких множеств и лингвистических переменных [4, 10, 11]. Здесь в качестве гипотезы информированности, позволяющей "устранить" исходную нечеткую неопределенность, принимаются функции принадлежности, которые, по утверждению Н. Н. Моисеева, "дают субъективное представление эксперта (исследователя) об особенностях исследуемой операции, о характерах ограничений и целей исследования. Это всего лишь новая форма утверждения гипотез, но она открывает и новые возможности".

Идея нечетко-множественного подхода к оценке жизнеспособности проекта в условиях нечеткой, неточной информации заключается в анализе возможных альтернатив разработки проекта в виде сетевых графиков выполнения проектных работ. Оценка каждой  $i$ -й альтернативы разработки проекта осуществляется на основе специально сформированного *нечеткого показателя жизнеспособности* в виде критического пути, длительность которого представляется нечетким числом:

$$\Delta_i = [T_i^-, T_i^+], i = \overline{1, m},$$

где  $T_i^-$ ,  $T_i^+$  — минимальное и максимальное время выполнения  $i$ -й альтернативы;  $m$  — число альтернатив разработки проекта.

Пусть заданы альтернативы сетевых графиков выполнения проектных работ с указанием объема каждой работы в виде нечеткого трапециевидного числа:

$$V_{i,j} = (V_{i,j,1}, V_{i,j,2}, V_{i,j,3}, V_{i,j,4}), i = \overline{1, m}, j = \overline{1, k_i},$$

где  $V_{i,j,1}$  — пессимистическая оценка выполнения объема работ,  $[V_{i,j,2}, V_{i,j,3}]$  — интервал ожидаемого (возможного) объема работ,  $V_{i,j,4}$  — оптимистическая оценка выполнения объема работ,  $k_i$  — число видов

работ в  $i$ -й альтернативе разработки проекта. Здесь объем работ по проекту может быть представлен также в виде нечетких треугольных чисел.

Полагаем, что задана производительность разработчиков проекта, которая известна неточно, поскольку строится на основе опроса мнения разработчиков, т. е. для каждого конкретного объема работ ставится не конкретное время работ, а нечеткое множество "возможное время выполнения работ".

Для расчета показателя  $\Delta_i$ , характеризующего жизнеспособность  $i$ -й проектной альтернативы, предлагается алгоритм, состоящий из следующих **четырёх этапов**:

- ♦ построение функции принадлежности  $\mu_{\tilde{B}}(V_{i,j})$  нечеткого множества  $\tilde{B}$  ожидаемого объема выполнения работ из интервала  $[V_{i,j,2}, V_{i,j,3}]$ ;

- ♦ построение функции принадлежности  $\mu_{\tilde{A}}(T_{i,j})$  нечеткого множества  $\tilde{A}$  ожидаемого времени выполнения работ с указанными нечеткими числами  $V_{i,j}$  и неточными производительностями разработчиков;

- ♦ расчет длительности критического пути  $\Delta_i$  для каждой  $i$ -й альтернативы разработки проекта [12];

- ♦ выбор приемлемой альтернативы разработки проекта на основании интуиции или опыта по интервалам времени  $[T_i^-, T_i^+]$ , либо по средним значениям этих интервалов на основе анализа множества характеристик  $\Delta_i$ .

Считаем, что в контексте рассматриваемой задачи роль времени выполнения работ  $T_{i,j}$  играет множество  $X$ , а роль объемов работ  $V_{i,j}$  — множество  $Y$ , значения которых будем обозначать соответственно через  $x$  и  $y$ . Полагаем, что функциональная зависимость  $T_{i,j} = \varphi(V_{i,j})$ , определяющая производительность разработчиков, задана неточно, она "размыта", т. е. отображение  $\varphi: x \rightarrow y$  является нечетким отображением  $\tilde{\varphi}$  множества  $Y$  во множество  $X$  с функцией принадлежности  $\mu_{\tilde{\varphi}}(y, x)$ .

Для нахождения ожидаемого времени выполнения работ по проекту используется следующая **четырёх-шаговая процедура**, графическая иллюстрация которой представлена на рис. 2.

На первом шаге строится функция принадлежности  $\mu_{\tilde{B}}(y)$  (кривая 1) нечеткого множества  $\tilde{B}$ .

На втором шаге на основе функции принадлежности  $\mu_{\tilde{B}}(y)$  (кривая 1) формируются интервалы достоверности  $[y_{pn}, y_{pk}]$  посредством задания  $\alpha$ -уровней, причем при достаточно малом шаге  $\alpha_p \in [0, 1]$ ,  $p = 1, 2, \dots$  изменения значения  $\alpha$  можно получить достаточное число интервалов достоверности.

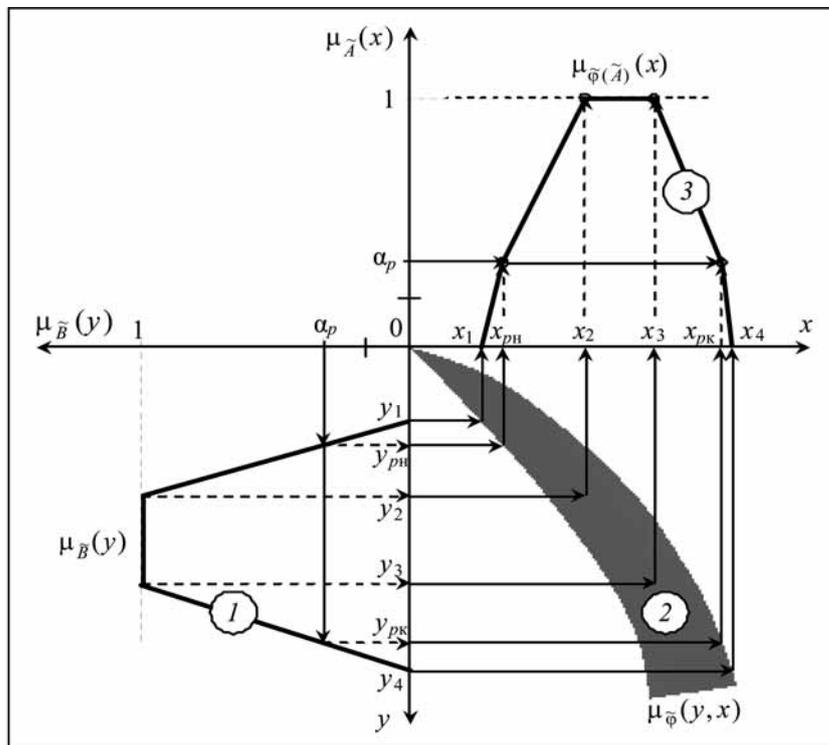


Рис. 2. Графическая иллюстрация построения функции принадлежности  $\mu_{\tilde{\varphi}}(x)$  при нечетком отображении

На третьем шаге формируются интервалы достоверности  $[x_{pn}, x_{pk}]$  посредством обращения нечеткого отображения (кривая 2).

На четвертом шаге осуществляется построение функции принадлежности  $\mu_{\tilde{\varphi}}(x)$  (кривая 3) по интервалам достоверности  $[x_{pn}, x_{pk}]$  и по значениям трапециевидного нечеткого числа  $x = (x_1, x_2, x_3, x_4)$ .

В основу данного алгоритма положена известная процедура кусочно-линейной аппроксимации функции принадлежности нечетких множеств семейством четких интервалов — так называемых  $\alpha$ -уровней, соответствующих значениям  $\alpha$  функции принадлежности. Очевидно, что чем больше число  $\alpha$ -уровней, тем точнее результат данной аппроксимации. Применение  $\alpha$ -уровней на втором шаге алгоритма обусловлено тем, что оно упрощает процедуру извлечения знаний от разработчиков для построения функции принадлежности  $\mu_{\tilde{\varphi}}(x)$ : разработчику необходимо формализовать свои представления о возможных значениях оцениваемой величины путем задания характеристической функции множества значений, которые она может принимать.

Заметим, что на рис. 2 функция принадлежности  $\mu_{\tilde{\varphi}}(x)$  (кривая 3), полученная посредством функции принадлежности  $\mu_{\tilde{B}}(y)$  нечеткого множества  $\tilde{B}$  и функции принадлежности  $\mu_{\tilde{\varphi}}(y, x)$  нечеткого множества  $\tilde{\varphi}$ , представляет собой нечеткое число  $\tilde{A}$  в виде

набора  $\alpha$ -сечений  $[A]^0 = [x_1, x_4]$ ,  $[A]^p = [x_{pH}, x_{pK}]$ ,  $[A]^1 = [x_2, x_3]$  при заданном наборе  $\alpha$ -уровней.

На рис. 2  $y = (y_1, y_2, y_3, y_4)$  — трапециевидное нечеткое число, где  $y_1$  — нижние границы интервала;  $[y_2, y_3]$  — интервалы наиболее ожидаемых значений анализируемых параметров;  $y_4$  — верхние границы интервала;  $[y_{pH}, y_{pK}]$  — интервалы достоверности нечеткого множества  $\tilde{B}$  на  $\alpha_p$  уровне, причем  $y_{pH}$  — начало, а  $y_{pK}$  — конец данных интервалов.

### Нечетко-интервальный подход к оценке жизнеспособности программного проекта в условиях интервальной неопределенности

Очевидно, что целесообразна не однокритериальная, а многокритериальная оценка жизнеспособности ПП на основе векторного критерия, характеризующего технические, финансовые, экономические и коммерческие показатели разрабатываемого проекта [5].

Анализ различных подходов к построению моделей многокритериального оценивания альтернатив проектов в условиях неопределенности показал, что параметры моделей (время выполнения проекта, объем работ по проекту, стоимость проекта и др.) достаточно часто являются интервальными величинами, что связано, во-первых, с применением методов параметрической идентификации моделей, с помощью которых зачастую можно определить только область допустимых значений параметров, а, во-вторых, с существующим разбросом мнений при получении значений параметров у разработчиков проекта.

В связи с этим целесообразно использовать нечетко-интервальный подход, предложенный автором в работе [11], идея которого заключается в следующем.

Пусть имеется ограниченное множество из  $m$  допустимых альтернатив разработки проекта  $X = \{x_1, x_2, \dots, x_m\}$ , где каждая альтернатива  $x_i \in X$ ,  $i = \overline{1; m}$  оценивается кортежем из  $n$  частных критериев  $K = \langle k_j(x_i) \rangle$ ,  $j = \overline{1; n}$ . Ставится задача формирования скалярной интервальной оценки обобщенной полезности, характеризующей жизнеспособность каждой  $i$ -й альтернативы разработки проекта из общего числа  $m$  альтернатив с помощью обычной интервальной арифметики:

$$P(x_i) = \sum_{j=1}^n w_j^H p_j^H(x_i), \quad i = \overline{1; m}, \quad j = \overline{1; n} \quad (1)$$

с возможностью последующего выбора приемлемой альтернативы:

$$x^0 = \arg \max_{x_i \in X} P(x_i), \quad i = \overline{1; m}, \quad (2)$$

где  $w_j^H$  — нормализованный интервальный коэффициент относительной важности  $j$ -го частного критерия альтернатив  $x_i \in X$ ;  $p_j^H(x_i)$  — нормализованные интервальные частные критерии альтернатив  $x_i \in X$ .

Для решения данной задачи предлагается следующий **четырёхэтапный алгоритм**.

На первом этапе осуществляется нормализация частных критериев  $p_j^H(x_i)$  для каждой  $i$ -й альтернативы разработки проекта:

$$p_j^H(x_i) = \frac{k_j(x_i) - k_j^-}{k_j^+ - k_j^-}, \quad 0 < p_j^H(x_i) < 1, \quad j = \overline{1; n}, \quad i = \overline{1; m},$$

где  $k_j(x_i)$  — интервал  $j$ -го частного критерия  $i$ -й альтернативы разработки проекта;  $k_j^-$ ,  $k_j^+$  — минимальный и максимальный интервал  $j$ -го частного критерия на данном множестве альтернатив  $x_i \in X$ .

Следует заметить, что интервальные частные критерии  $k_j(x_i)$ ,  $j = \overline{1; n}$  не должны пересекаться, поскольку только в этом случае устанавливаются отношения "больше" или "меньше". Так, например, согласно рекомендациям ряда авторов, для того чтобы интервалы  $a = [a_1, a_2]$  и  $b = [b_1, b_2]$  были сравнимы в отношении  $a \geq b$ , необходимо и достаточно выполнение условия  $a_1 \geq b_1$ ,  $a_2 > b_2$  [13], а также  $a_1 > b_2$  [14]. В случае пересечения интервалов предлагается использовать середины указанных интервалов частных критериев, что приводит к вырождению задачи (1)-(2) в детерминированную оптимизационную задачу.

Заметим, что если два интервальных критерия с одинаковой моделью описания имеют совпадающие границы, то в интервальном анализе справедливы следующие соотношения:  $a - a = 0$ ,  $a/a = 1$ , и  $a - b \neq 0$ ,  $a/b \neq 1$ . Следовательно, существуют обратные элементы и справедлив дистрибутивный закон [15].

Важно подчеркнуть, что необходимость нормализации частных критериев обусловлена тем, что их числовые значения отличаются единицами измерения и порядком величин, что затрудняет дальнейшие операции с этими критериями. Частные критерии могут быть представлены в виде интервальных или нечетких  $L-R$ -типа (например, треугольных, трапециевидных) чисел.

На втором этапе осуществляется нормализация интервальных коэффициентов  $w_j^H$ ,  $j = \overline{1; n}$  относительной важности частных критериев на основе известного метода Саати:

$$w_j^H = \frac{w_i}{\sum_{j=1}^n w_j}, \quad j = \overline{1; n}, \quad (3)$$

где  $w_j$  — интервальный коэффициент относительной важности  $j$ -го частного критерия;  $w_j^H = [\alpha_{j\min}, \alpha_{j\max}]$  — нормированный интервальный коэффициент относительной важности  $j$ -го частного критерия. При этом полагается, что  $\sum_{j=1}^n \alpha_{j\min} < 1$ ,  $\sum_{j=1}^n \alpha_{j\max} > 1$ , так как в противном случае задача (2) не имеет решения из-за

невозможности выполнения ограничения  $\sum_{j=1}^n \alpha_j = 1$ .

Если же  $\sum_{j=1}^n \alpha_{j \min} = 1$  или  $\sum_{j=1}^n \alpha_{j \max} = 1$ , то задача (2) вырождается в обычную детерминированную оптимизационную задачу [16].

Здесь интервальные коэффициенты относительной важности частных критериев могут быть сформированы на основе опроса мнения группы разработчиков и представлены в виде как интервальных, так и точных значений (детерминированный случай).

На третьем этапе рассчитываются на основе соотношения (1) скалярные интервальные оценки обобщенной полезности  $P(x_i)$  для каждой  $i$ -й альтернативы разработки проекта.

На четвертом этапе осуществляется выбор наилучшей альтернативы разработки проекта, для которой интервальная оценка имеет наибольшее значение.

Следует отметить, что на практике задача выбора наилучшей альтернативы проекта является гораздо более сложной, чем кажется на первый взгляд. Дело в том, что задача (2) корректна только в случае, если границы интервалов частных критериев не пересекаются, в противном случае задача (2) становится некорректной по Адамару, так как не имеет единственного решения и необходима ее регуляризация, т. е. трансформация на основе привлечения дополнительной, внешней по отношению к исходной задаче, информации. Носителем такой информации при решении задач многокритериальной оптимизации является руководитель проекта, поскольку он принимает решение. В настоящее время широко используются методы, позволяющие трансформировать исходную

задачу многокритериальной оптимизации в задачу однокритериальной скалярной оптимизации. К ним относятся принцип главного критерия; схема последовательной оптимизации; анализ иерархий и др.

Другим подходом к выбору наилучшей альтернативы проекта может служить упорядочивание интервальных оценок обобщенной полезности  $P(x_i)$ , представленных в виде трапециевидных нечетких чисел, в соответствии с приспосабливаемым им рейтингом. Для сравнения нечетких чисел предлагаются методы Чью Парка, Чанга, Кауфмана—Гупты [17].

### Пример оценки и выбора приемлемой проектной альтернативы в условиях интервальной неопределенности

Допустим, что число альтернатив разработки проекта конечно и частные критерии заданы в виде нечетких трапециевидных чисел, причем их границы не пересекаются.

Пусть имеется четыре альтернативы ( $x_1, x_2, x_3, x_4$ ) разработки проекта, описываемые совокупностью четырех частных критериев жизнеспособности:  $k_1$  — время выполнения проекта, мес.;  $k_2$  — стоимость проекта, тыс. руб.;  $k_3$  — срок окупаемости, мес.;  $k_4$  — вероятность успеха проекта в условиях возникновения ситуаций, при которых цели, поставленные в проекте, могут быть не достигнуты полностью или частично [5]. Ставится задача оценки альтернатив с возможностью последующего выбора приемлемой альтернативы разработки проекта для ее дальнейшего осуществления.

Расчетные значения частных критериев для каждой альтернативы представлены в табл. 1. На основе табл. 1 осуществляется нормализация частных критериев  $p_j^H(x_i)$ , результаты которых представлены в табл. 2.

Таблица 1

Расчетные значения частных критериев жизнеспособности альтернатив

Альтернативы проекта	Частные критерии жизнеспособности альтернатив			
	$k_1$	$k_2$	$k_3$	$k_4$
$x_1$	[9,2, 9,4, 9,6, 9,9]	[640, 660, 680, 690]	[3, 3,2, 3,3, 3,4]	[0,8, 0,82, 0,83, 0,83]
$x_2$	[12, 12,4, 12,7, 13]	[700, 703, 706, 710]	[2, 2,3, 2,4, 2,5]	[0,87, 0,88, 0,89, 0,9]
$x_3$	[8, 8,2, 8,4, 8,5]	[600, 610, 620, 630]	[3,5, 3,6, 3,7, 3,9]	[0,7, 0,72, 0,73, 0,74]
$x_4$	[10, 10,3, 10,4, 10,5]	[740, 750, 770, 800]	[4, 4,1, 4,3, 4,5]	[0,84, 0,85, 0,86, 0,87]

Таблица 2

Расчетные нормированные значения частных критериев жизнеспособности альтернатив

Альтернативы проекта	Нормированные значения частных критериев жизнеспособности альтернатив			
	$k_1$	$k_2$	$k_3$	$k_4$
$x_1$	[0,14, 0,22, 0,35, 0,54]	[0,05, 0,25, 0,54, 0,82]	[0,2, 0,4, 0,59, 0,93]	0
$x_2$	1	[0,35, 0,52, 0,74, 1]	0	[0,27, 0,44, 0,63, 0,93]
$x_3$	0	0	[0,4, 0,6, 0,82, 1]	1
$x_4$	[0,3, 0,42, 0,56, 0,71]	1	1	[0,45, 0,67, 0,88, 1]

Далее осуществляется нормализация интервальных коэффициентов относительной важности частных критериев. Предположим, что на основе опроса мнения группы разработчиков, известны следующие весовые коэффициенты относительной важности частных критериев в виде трапециевидных чисел:

$$w_1 = [3, 3,2, 3,5, 4], w_2 = [3, 3,5, 3,8, 4,5],$$

$$w_3 = [1, 1,3, 1,5, 2], w_4 = [2, 2,4, 2,6, 3],$$

тогда на основе соотношения (3) рассчитываются нормированные интервальные коэффициенты относительной важности частных критериев:

$$w_1^H = [0,22, 0,28, 0,34, 0,44]$$

$$w_2^H = [0,22, 0,31, 0,37, 0,5],$$

$$w_3^H = [0,07, 0,11, 0,14, 0,22],$$

$$w_4^H = [0,15, 0,21, 0,25, 0,33].$$

Затем рассчитываются скалярные интервальные оценки обобщенной полезности  $P(x_j)$  для каждой альтернативы проекта:

$$P(x_1) = [0,05, 0,18, 0,4, 0,86], P(x_2) = [0,24, 0,44, 0,7, 1,2],$$

$$P(x_3) = [0,18, 0,28, 0,37, 0,55], P(x_4) = [0,29, 0,53, 0,82, 1,37].$$

Поскольку интервалы частных критериев пересекаются, то выбор наилучшей альтернативы на основе выражения (2) осуществлять нельзя. В данном примере для выбора наилучшей альтернативы проекта применяется метод Чью-Парка, позволяющий каждому трапециевидному числу поставить в соответствие четкое число:

$$Cr(P(x_1)) = 0,666; Cr(P(x_2)) = 1,21;$$

$$Cr(P(x_3)) = 0,669; Cr(P(x_4)) = 1,43.$$

Таким образом, наилучшей альтернативой разработки проекта является альтернатива  $X_4$ , за которой следует альтернатива  $X_2$  (упорядочение проводится по убыванию значений).

Поскольку в обычной интервальной арифметике осуществляется быстрый рост ширины интервала результата по мере возрастания числа арифметических операций с интервалами, то предложенный алгоритм целесообразно применять для обобщенной интервальной арифметики и централизованных форм, целью которых является попытка препятствовать росту интервала.

1. Колоденкова А. Е. Ключевые проблемы концептуального проектирования перспективных систем в условиях неполноты исходной информации // Материалы Шестой Всероссийск. науч.-практ. конф. "Перспективные системы и задачи управления" и Третьей молодежн. школы-семинар. "Управление и обработка информации в технических системах" Таганрог: ТТИ ЮФУ, 2011. С. 193—196.
2. Нариньяни А. С. Введение в недоопределенность // Информационные технологии. Приложение. 2007. № 4. 32 с.
3. Валькман Ю. Р., Быков В. С., Рыхальский А. Ю. Моделирование НЕ-факторов — основа интеллектуализации компьютерных технологий // Системні дослідження та інформаційні технології. 2007. № 1. С. 39—61.
4. Колоденкова А. Е. НЕ-факторы и методы вычислительного интеллекта в концептуальном проектировании // Интегрированные модели и мягкие вычисления в искусственном интеллекте. Сб. тр. VI-й Междунар. науч.-практ. конф. 2011.
5. Колоденкова А. Е. Анализ жизнеспособности — важная стадия жизненного цикла инновационных программных проектов // Программная инженерия. 2010. № 1. С. 21—30.
6. Колоденкова А. Е. Статистический подход к оценке реалистичности программных проектов для автоматизированных информационно-управляющих систем // Мехатроника, автоматизация, управление. 2010. № 4. С. 52—60.
7. Архипенков С. Я. Лекции по управлению программными проектами. М., 2009. URL: [http://www.archipenkov.ru/resources/sw\\_project\\_management.pdf](http://www.archipenkov.ru/resources/sw_project_management.pdf).
8. Брежнев Е. В., Адаменко А. А. Метод прогнозирования оценки риска с использованием нечеткого вывода. URL: [http://www.nbu.gov.ua/portal/natural/SOI/2009\\_2/Bregph.pdf](http://www.nbu.gov.ua/portal/natural/SOI/2009_2/Bregph.pdf).
9. Шашкин А. И., Ширяев М. М. Календарное планирование работ по проекту на основе нечетких исходных данных. URL: <http://vestnik-samgu.samsu.ru/est/2008web3/math/200830112.pdf>.
10. Колоденкова А. Е. Нечетко-множественный подход к оценке реалистичности альтернатив программного обеспечения мехатронных систем // Мехатроника, автоматизация, управление. 2011. № 4. С. 45—53.
11. Колоденкова А. Е. Интеллектуальные технологии в задачах оценки жизнеспособности проектов сложных систем // Проблемы управления и моделирования в сложных системах. Тр. XIII Междунар. конф. Самара: Самарский НЦ РАН. 2011.
12. Акимов В. А., Балашов В. Г., Заложнев А. Ю. Метод нечеткого критического пути // Управление большими системами. 2003. Вып. 3. С. 5—10.
13. Левин В. И. Сравнение интервальных величин и оптимизация неопределенных систем // Информационные технологии. 1998. № 7. С. 22—32.
14. Бронз П. В. Разработка методов оценки экономической эффективности инвестиционных проектов электростанций по интервальному данным. Автореф. дис. ... канд. экон. наук. М., 2007.
15. Воцинин А. П. Интервальный анализ данных: развитие и перспективы // Заводская лаборатория. 2002. Т. 68, № 1. С. 118—126.
16. Бодянский Е. В., Петров К. Э. Определение экстремальных многофакторных оценок альтернативных вариантов решений в условиях интервальной неопределенности // Проблемы информационных технологий: сб. науч. пр. Херсонського національного технічного університету. 2008. № 2(4). С. 27—33.
17. Аньшин В. М., Демкин И. В., Никонов И. М., Царьков И. Н. Применение теории нечетких множеств к задаче формирования портфеля проектов // Проблемы анализа риска. 2008. Т. 5, № 3. С. 8—21.

**А. В. Шокуров**, канд. физ.-мат. наук, Лаборатория вычислительных методов Механико-математического факультета МГУ им. М. В. Ломоносова,  
e-mail: shokurov.anton.v@yandex.ru

## К созданию программных средств кодирования растровых изображений изменяемой детализации для их адаптивного интерактивного анализа

*Настоящая работа посвящена исследованию методов кодирования визуальной информации, разработке математических моделей таких данных, программных механизмов и средств управления, а также анализу результатов их тестовых испытаний. Представлено описание разработанного автором на базе положений традиционного метода SPIHT эффективного способа адаптивного интерактивного анализа растрового изображения изменяемой детализации, а также приведены результаты исследований свойств программной реализации на тестовых изображениях. Эффективность включает объемы получаемых кодированных потоков данных, скорость извлечения фрагментов изображения, количество соответствующих извлекаемым фрагментам данных и объем используемой программным обеспечением оперативной памяти.*

**Ключевые слова:** растровое изображение изменяемой детализации, кодирование на базе метода SPIHT, интерактивный анализ изображения, канал связи низкой пропускной способности, многомасштабный и постепенный режим передачи фрагмента изображения, адаптивный к размеру фрагмента изображения режим извлечения фрагмента

### Введение

Создание высокоэффективных программных средств визуализации и обработки изображений с использованием современных информационно-вычислительных технологий — одна из важнейших задач программной инженерии. Оно во многом определяет перспективы использования таких технологий — механизмов, средств и систем на их основе — во многих сферах национального хозяйственного комплекса. Программные средства кодирования растровых изображений изменяемой детализации в целях их адаптивного интерактивного анализа занимают особое положение. На сегодня они востребованы как в традиционных областях деятельности человека, таких как медицинская диагностика и дистанционное зондирование, так и в критически важных — охрана общественно-правового порядка и обеспечение государственной безопасности. Однако разработка отмеченных

программных средств представляет собой сложный итерационный процесс. Жизненный цикл такого программного продукта начинается с этапа предпроектных исследований и создания его макета, который в ходе тестовых испытаний должен продемонстрировать эффективность предлагаемых решений и инновационные перспективы создания промышленного продукта. В настоящей работе приведены результаты исследований.

С развитием средств вычислительной техники стало возможным формирование растровых изображений большого размера и разрешения. Возрастающие требования, которые предъявляются при этом к качеству изображений, приводят к постоянному увеличению размера и разрешения как самих растровых изображений, так и устройств, их отображающих. Последнее обстоятельство вызвало потребность в исследовании и разработке методов кодирования соответствующей визуальной информации в виде данных, что влечет за собой необхо-

димось разработки специальных моделей данных и новых принципов их проектирования. Кроме объективно возникающих вопросов о хранении данных, появляется необходимость в удаленном анализе (в рамках клиент-серверной архитектуры) растровых изображений, например, по каналу связи низкой пропускной способности. Заметим, что размер раstra запрашиваемых для анализа фрагментов изображений может существенно различаться в зависимости от специфики класса анализируемых изображений и от конкретных потребностей эксперта. Так, при отображении фрагмента изображения на стенде, состоящем из нескольких мониторов (или экранов), необходимо извлекать фрагменты большого размера, а при отображении на переносном устройстве, либо при изучении большого числа разрозненных фрагментов появляется потребность в извлечении фрагментов небольшого размера. Данные произвольного фрагмента при этом должны передаваться последовательно, в порядке, обеспечивающем постепенное улучшение качества восстанавливаемого фрагмента вплоть до точного восстановления. Как следствие, появляется необходимость в разработке и исследовании метода *адаптивного интерактивного анализа* растровых изображений.

Важной характеристикой растровых изображений является *степень детализации*, определяющая выбор оптимального разрешения, при котором не будут потеряны существенные детали растрового изображения. Во многих растровых изображениях степень детализации варьируется, т. е. *растровые изображения* имеют *области с изменяемыми степенями детализации*. Фотографии, например, содержат как попавшие, так и не попавшие в ГРИП (глубина резко изображаемого пространства) области. Соответственно, попавшие в ГРИП области обычно имеют более высокую степень детализации, чем не попавшие в нее. Использование информации о вариациях степени детализации позволило бы существенно увеличить степень сжатия алгоритма кодирования и существенно уменьшить объем извлекаемых данных соответствующего алгоритма декодирования. Существующий способ (основанный на понятии *региона интереса*) в недостаточной степени (только на стадии квантования) использует информацию об областях изменяемой детализации, что отрицательно влияет на объем кодированного потока данных и на процесс извлечения фрагмента. Последнее подчеркивает важность разработки новой модели изображения, которая позволяет фильтровать неактуальную визуальную информацию.

Задача интерактивного анализа растровых изображений в той или иной степени считается решенной. Однако известные решения рассматриваемой задачи достигаются за счет введения большого числа ограничений. Этого, в частности, можно добиться путем фиксации размера раstra отображающего устройства, на котором будет проводиться интерактивный анализ изображения, а также путем использования канала связи высокой пропускной способности.

Все изложенное выше позволяет сделать вывод, что проблема кодирования и адаптивного интерактивного анализа растровых изображений большого размера, а тем более растровых изображений изменяемой детализации, еще недостаточно исследована.

Метод SPIHT [1] является одним из ключевых в области теории кодирования растровых изображений. В частности, ввиду присущей ему высокой степени сжатия и оптимальной характеристике ПОСШ<sup>1</sup> данный метод очень часто применяется при сравнительном тестировании новых методов кодирования растровых изображений. Отметим, что в отличие от известных конкурирующих методов (JPEG [2], JPEG2000 [3], PNG и т. п.) данный метод построен на интуитивно понятных принципах (иерархическое представление и *нуль-дерево* [4]), применим к данным, имеющим разную природу происхождения (видео [5], аудио, распределение плотности вещества в объеме [6] и т. д.), его можно использовать при произвольном вейвлет-преобразовании [7, разделы 7.4, 7.5 и 7.7] и, вообще, при произвольном иерархическом преобразовании.

Это определило выбор темы настоящей статьи, которая актуальна в теоретическом и практическом плане и посвящена исследованию методов кодирования визуальной информации, разработке математических моделей таких данных и программных механизмов управления ими. Представленный в работе метод GISS-SPIHT кодирования построен на принципах, положенных в основу метода SPIHT преобразования визуальной информации в данные. Эффективность представленного метода включает объемы получаемых кодированных потоков данных, скорость извлечения фрагментов (размер раstra которых может составлять от нескольких десятков до нескольких тысяч пикселей по каждому из измерений), количество соответствующих извлекаемым фрагментам данных и объем используемой оперативной памяти.

## Краткий обзор исследований

**Блочное кодирование растрового изображения.** Задача интерактивного анализа растровых изображений в той или иной степени решается [8–10] путем разбиения изображения на квадратные блоки определенного фиксированного размера и кодирования каждого из них по отдельности тем или иным *основным методом* (JPEG, PNG, SPIHT и т. п.). Размер квадратных блоков, на которые разбивается исходное растровое изображение (вейвлет-представление) при кодировании, определяет грануляцию эффективного задания положения и размера извлекаемого фрагмента изображения. Последний факт означает, что при уменьшении грануляции, в частности, для возможности из-

<sup>1</sup> Пиковое отношение сигнала к шуму (в англоязычной литературе PSNR) является стандартной метрикой (измеряется в децибелах) измерения схожести между двумя растровыми изображениями.

влечения данных, соответствующих небольшому фрагменту изображения, необходимо уменьшить размер обозначенных квадратных блоков.

Однако с уменьшением размера квадратных блоков понижается эффективность декорреляции пикселей (вейвлет-коэффициентов) блока, что негативно влияет на объем кодированного потока данных и, соответственно, извлекаемой информации при восстановлении фрагмента изображения. Последнее обстоятельство в зависимости от используемого основного метода кодирования при сильной степени сжатия может привести к образованию визуально заметных артефактов.

Существенным недостатком является также то, что такой подход плохо масштабируется с ростом размера изображения, а точнее с ростом размера запрашиваемых фрагментов. При таком росте существенно увеличивается число блоков, данные которых необходимо передать клиенту, что сильно загружает сервер и, в конечном счете, существенно влияет на время, затрачиваемое на извлечение данных, относящихся к фрагменту.

С уменьшением размера квадратных блоков также уменьшается и количество масштабов (при кодировании блоков соответствующим методом, например, SPIHT), на которые эти блоки потенциально могут быть разбиты, а именно — уменьшается количество масштабов, при которых могут быть эффективно извлечены фрагменты изображения. Отметим, что при очень большом числе уровней (несоразмерном размеру блока) верхние уровни будут иметь очень малый размер (вплоть до  $1 \times 1$  пиксель при максимальном числе уровней). Следовательно, при необходимости восстановления сильно уменьшенной копии всего изображения придется извлекать верхние уровни всех блоков. Последнее обстоятельство означает, что будет извлечен существенно больший, чем в действительности необходим, объем данных, в частности, больше размера запрашиваемой копии изображения.

На завершающей стадии процесса извлечения фрагмента также возможно необходимо будет отдельно выполнить *домасштабирование*, а именно — уменьшение восстановленного фрагмента до требуемого размера. Последнее отмеченное действие при программной реализации алгоритма не всегда возможно как в силу нехватки памяти (как оперативной, так и внешней), так и по причине недостаточной мощности вычислительной системы. Как следствие, при применении данного подхода затруднено восстановление уменьшенной копии всего изображения, особенно при кодировании изображения большого размера.

Исходя из отмеченного выше, при использовании блочного подхода многомасштабность, необходимая для интерактивного анализа изображения, добавляется искусственным образом. Этот факт означает, что по растровому изображению вначале строится пирамида его масштабов, а потом к каждому уровню пирамиды применяется описанный ранее подход. Недостатком такого решения является избыточность

данных и вычислений, а также факт неиспользования связи между кодируемыми уровнями. Последнее обстоятельство означает отсутствие целостности представления, которое может оказаться крайне важным при решении более сложных задач (например, распознавание образов, поиск фрагмента в изображении по шаблону и т. п.) на базе данного представления.

Таким образом, необходимо разработать специальный механизм, устраняющий отмеченные недостатки тривиального разбиения изображения на блоки, что расширит его применимость.

Вместо того чтобы разрабатывать основной метод кодирования с нуля, рассмотрим известные традиционные методы кодирования растровых изображений и выберем тот, на базе принципов лежащих в основе которого целесообразно построить новый метод.

**Дискретное косинусное преобразование.** Для кодирования отдельных блоков, в частности, можно использовать программное средство, реализующее метод (стандарт) JPEG, который основан на *дискретном косинусном преобразовании* (ДКП). Однако большинство методов, основанных на ДКП, имеют существенные недостатки. Во-первых, ДКП фиксировано и, как следствие, не может быть адаптировано в зависимости от класса входных растровых изображений. Более того, существенно затруднено его применение к данным, происхождение которых отлично от аудиовизуальных. Во-вторых, данное преобразование не позволяет естественным образом воспользоваться многомасштабностью. Кроме того, при больших степенях сжатия визуально заметна блочность кодирования. Следует отметить, что основным преимуществом известных методов, основанных на методе ДКП, является их широкая распространенность и, соответственно, высокая скорость кодирования/декодирования.

В то же время при определенных условиях, характерных для решения задач, похожих на поставленную в настоящей статье, применение данного метода совместно с ранее отмеченным подходом разбиения изображения на блоки может оказаться целесообразным. Однако отмеченные выше недостатки в общем случае позволяют утверждать, что такой способ не является решением поставленной в работе задачи.

**Дискретное вейвлет-преобразование.** Альтернативный известный подход к кодированию растровых изображений основан на теории вейвлет-преобразований [11]. К преимуществам данного подхода можно отнести возможность адаптирования вейвлет-преобразования под входное изображение, что позволяет существенно повысить степень сжатия методов кодирования для определенных (специальных) классов растровых изображений, а также под альтернативные аудиовизуальные входные данные.

При применении отмеченного преобразования масштабируемость достигается естественным образом, так как вейвлет-преобразование, используемое для декорреляции данных растрового изображения, автоматически строит и его масштабы. Благодаря свойству компактизации энергии и ее более точному

соответствию зрительной системе человека [12], большинство методов, основанных на вейвлет-преобразовании, превосходят [13] метод JPEG по качеству восстановления растрового изображения как объективно (согласно метрике ПОСШ), так и субъективно (согласно средней оценке мнения группы экспертов). Следует также отметить, что при использовании небольшой начальной части потока данных, методом, основанным на вейвлет-преобразовании, присущи более благоприятные визуальные погрешности в сравнении с методом JPEG [11]. В частности, алгоритмы распознавания образов (лица, отпечатки пальцев и тому подобные) в среднем работают лучше на изображениях, которые были закодированы с использованием методов SPIHT и JPEG2000, основанных на вейвлет-преобразовании, чем закодированные с использованием метода JPEG.

Отмеченные выше преимущества подтверждают целесообразность использования вейвлет-преобразования для решения поставленной в настоящей статье задачи.

**Методы кодирования вейвлет-представления.** Рассмотрим методы кодирования вейвлет-представления. Можно выделить два класса методов кодирования вейвлет-представления растрового изображения, а именно — внутридиапазонный и междиапазонный [13].

Внутридиапазонные методы (например, EVCOT [14], SPECK и SWEET) основаны на устранении корреляции между соседними вейвлет-коэффициентами в каждом из поддиапазонов по отдельности. Заметим, что метод EVCOT был взят за основу стандарта JPEG2000, который также в той или иной степени решает задачу интерактивного анализа изображений. К недостатку метода EVCOT можно отнести сложный и, как следствие, фиксированный характер его преобразования под кодирование визуальной информации растровых изображений, что означает отсутствие возможности его существенной модификации. К недостаткам самого метода JPEG2000 можно отнести факт использования подхода на основе разбиения на блоки и несоответствие объема извлеченных данных размеру растра небольших фрагментов.

Междиапазонные методы основаны на устранении корреляции между вейвлет-коэффициентами, принадлежащими к разным вейвлет-поддиапазнам. Ключевым понятием в таких методах является нуль-дерево, в основе которого лежит гипотеза о том, что если вейвлет-коэффициент меньше определенного порога, то с высокой вероятностью и его *дету*<sup>2</sup> тоже меньше. Метод SPIHT<sup>3</sup> является ключевым в данной категории и часто применяется при сравнительном

<sup>2</sup> На вейвлет-представлении можно естественным образом задать иерархическую древовидную структуру.

<sup>3</sup> Отметим, что хотя в оригинальной статье, описывающей данный метод, используется конкретное вейвлет-преобразование, на самом деле данный метод применим к произвольному вейвлет-представлению.

тестировании новых методов кодирования растровых изображений.

Следует отметить, что методы кодирования текстур [15] также позволяют интерактивно анализировать изображения (определяющим свойством методов кодирования текстур как раз и является возможность случайного доступа к любой его части при произвольном масштабе). Однако они, например, не обладают (так как это не требуется) существенно важной для решаемой в настоящей статье задачи возможностью последовательной передачи данных с постепенным повышением визуального качества восстанавливаемого изображения (согласно метрики ПОСШ).

**Выбор базового метода.** Каждый из известных методов кодирования растровых изображений имеет свою область применения ввиду того, что в каждой области существуют соответствующие ей требования к программной реализации (например, скорость работы, используемый объем оперативной и дисковой памяти и качество восстанавливаемого изображения). Более того, со временем требования к качеству кодирования/декодирования решаемых задач только повышаются. В зависимости от области применения (для повышения эффективности) специфицируется также и класс кодируемых изображений.

Метод SPIHT активно исследуется, используется и финансируется применительно к задачам космического, медицинского и военного характера. Выбор метода SPIHT для решения этих задач был обусловлен в том числе и ранее перечисленными соображениями. С момента разработки метода SPIHT и по настоящее время на базе данного метода учеными было разработано большое число улучшений и расширений. Эти исследования касаются повышения качества восстановления изображения и степени сжатия [16, 17], устойчивости к ошибкам в канале связи [18], упрощениям [19] и реализации метода SPIHT на специализированных вычислительных машинах [20]. В частности, исследователями была разработана возможность извлечения всего растрового изображения при различных масштабах [21]. Однако необходимо отметить, что задача извлечения фрагмента растрового изображения при различных масштабах в общем случае решена не была.

Принимая во внимание перечисленные ранее результаты сравнительного анализа известных методов кодирования растровых изображений, было решено при разработке адаптивного интерактивного метода анализа растровых изображений изменяемой детализации использовать принципы, лежащие в основе метода SPIHT.

## Модель изображения

При первоначальном появлении аналого-цифровых устройств (к которым относится и устройство захвата растровых изображений) основная трудность заключалась в получении адекватного решаемой задаче качества (разрешающей способности) цифрового представления. С развитием средств вычислительной

техники эта трудность для широкого класса задач была устранена. Однако появилась новая трудность, которая заключается в том, что современные аналого-цифровые устройства формируют очень большой объем данных. Следует отметить, что, вообще говоря, отсутствует необходимость в кодировании всех входных аналоговых данных при одинаково высоком качестве. Более того, аналоговый сигнал (изображение) может состоять из частей, имеющих разные требования к качеству их представления.

Традиционное понятие "растровое изображение" не позволяет учитывать информацию о степени детализации различных областей в изображении. Отметим, что большие растровые изображения, которые рассматриваются в рамках решаемой в настоящей статье задачи, могут содержать области, востребованная степень детализации которых может существенно варьироваться. По этой причине, в случае кодирования изображений большого размера понятие растрового изображения является несодержательным.

Принимая во внимание отмеченный факт, автором разработана новая модель изображения, которая учитывает заданные пользователем степени детализации областей в изображении. Учитывая представленные недостатки растрового изображения, формализуем понятие растрового изображения с областями изменяемой детализации, а именно введем понятие фильтра  $\tau$ , который для каждой области задает ее степень детализации  $detail$  и положение, заданное в координатной системе растрового изображения. Причем, степень детализации равная 0 соответствует самому высокому разрешению, а с увеличением  $detail$  разрешение области понижается в  $2^{detail}$  раз.

Вейвлет-представление, будучи многомасштабным, естественным образом используется для задания на нем структуры, которая учитывает степень детализации областей изображения. Обозначим через  $\tilde{C}_N$  матрицу  $N$ -каскадного вейвлет-преобразования изображения  $I$ , где  $N := \max_{i=1}^M detail_i$ . Фильтр  $\tau$  используется для *фильтрации* вейвлет-коэффициентов: те вейвлет-коэффициенты, которым не соответствуют области необходимой степени детализации, полагаются равными  $\infty$ . Вейвлет-коэффициенты, не равные  $\infty$ , считаются *активными* и только они кодируются.

**Определение.** *Обобщенное растровое изображение*, отвечающее растровому изображению  $I$  и числу  $N \in \mathbb{N}$ , есть пара:  $\Theta := (N, \tilde{C}_N)$ , где  $\mathbb{N}$  — множество натуральных чисел.

На рис. 1 (см. вторую сторону обложки) показаны примеры растровых изображений с областями изменяемой детализации. В частности, фильтр  $\tau$  для изображения на рис. 1,  $a$  задан как

$$\tau := \{512, 512, Region_{лицо}, Region_{фон}\},$$

где область  $Region_{лицо}$  соответствует лицу и имеет степень детализации 0, а область  $Region_{фон}$  задает фон и имеет степень детализации 3.

Заметим, что фильтр  $\tau$  считается известным априори как серверу, так и клиенту. По этой причине фильтр не кодируется и не передается при запросе фрагмента изображения. Это обстоятельство является ключевым при разработке нового объекта — зачем кодировать и передавать то, что известно из каких-то других сообщений. Следующий раздел посвящен кодированию вейвлет-коэффициентов в предположении, что фильтр  $\tau$  известен.

## Схема кодирования

Схема кодирования состоит из алгоритма первичного кодирования обобщенного растрового изображения, вычисления дискретного вейвлет-преобразования обобщенного растрового изображения, кодирования соответствующих вейвлет-коэффициентов, сжатия пакетов данных и формирования целевого потока данных.

Сначала отметим специфику разрабатываемого метода. Результатом действия схемы кодирования является закодированный поток данных. Схема декодирования при восстановлении фрагмента изображения инициализируется сформированным таким образом потоком.

Автор использует специальный механизм при построении алгоритмов, позволяющий оценить объем оперативной памяти, используемый соответствующими программными реализациями. Благодаря последнему удается разработать такую схему кодирования, что объем оперативной памяти, используемой соответствующим программным обеспечением в процессе кодирования изображения, не зависит от размера его растра.

**Первичное кодирование.** Цель разработки алгоритма *первичного кодирования* обобщенного растрового изображения заключается в повышении эффективности разрабатываемой системы кодирования. В частности, для повышения корреляции между соседними вейвлет-коэффициентами было принято решение разбить поддиапазоны  $N$ -каскадного вейвлет-представления на блоки небольшого размера (например,  $64 \times 64$  вейвлет-коэффициентов), что улучшило кешируемость соседних вейвлет-коэффициентов [22]. Для обработки всех вейвлет-коэффициентов при этом достаточно обработать вейвлет-коэффициенты всех блоков. Используемый подход позволяет разработать такие алгоритмы, что используемый их программными реализациями объем оперативной памяти не зависел бы от размера изображения.

Автором предложен новый порядок хранения данных, соответствующих блокам, в закодированном потоке. Заметим, что в известных методах кодирования растровых изображений (например, в традиционном блочном подходе) порядок хранения блоков построочный. Такой порядок неэффективен при извлечении прямоугольных фрагментов растрового изображения. В новом подходе данные, соответствующие блокам, предложено хранить согласно кривой Гильберта, что

влечет равноценную корреляцию между блоками как по горизонтали, так и по вертикали. Последний факт означает, что данные соседних блоков эффективнее кешируются, т. е. предложенный порядок хранения блоков в сравнении с построчным порядком позволяет существенно сократить степень загрузки системы при извлечении фрагментов растрового изображения (обоснование дано при изложении схемы декодирования). Предложенные механизмы также актуальны при разработке алгоритмов, которые обрабатывают (например, вычисляют вейвлет-преобразование или кодируют) первичное представление изображения.

**Алгоритм дискретного вейвлет-преобразования.** Вейвлет-преобразование обобщенного растрового изображения  $\Theta$  определено как соответствующее вейвлет-преобразование поддиапазона LL. Можно показать, что поддиапазон LL обобщенного растрового изображения  $\Theta$  состоит только из активных узлов, представляя собой таким образом по сути растровое изображение. Следовательно, вейвлет-преобразование обобщенного растрового изображения "де-факто" представляет собой обобщенное растровое изображение. В частности,  $N$ -каскадное вейвлет-преобразование растрового изображения есть обобщенное растровое изображение. По этой причине, при представлении алгоритма кодирования считается, что все необходимые (в рамках метода кодирования изображений) вейвлет-преобразования к изображению уже применены.

Автором предложен алгоритм вычисления дискретного вейвлет-преобразования, который основан на "оконном" подходе. Последний факт означает, что сначала вычисляется дискретное вейвлет-преобразование (поддиапазоны LL, LH, HL и HH) некоей окрестности (зависящей от типа вейвлет-преобразования) четверки блоков ( $2 \times 2$  блока) исходного растрового изображения. Затем из соответствующих вычисленных поддиапазонов LL, LH, HL и HH вырезаются блоки, которые являются искомыми блоками соответствующих поддиапазонов вейвлет-представления растрового изображения. Дискретное вейвлет-преобразование всего растрового изображения вычисляется путем выполнения таких действий для всех четверок блоков. Данный алгоритм использует первичное представление обобщенного растрового изображения, что позволяет ему реализовывать качественные преимущества первичного представления. Отметим, что авторский алгоритм обеспечивает порядок вычисления блоков поддиапазонов, согласованный с кривой Гильберта. Последний факт означает, что как при каскадном применении данного алгоритма (именно таким образом предлагается вычислять  $N$ -каскадное вейвлет-преобразование), так и при кодировании вычисленных вейвлет-коэффициентов эффективность первичного представления сохраняется.

**Основной алгоритм кодирования.** В ходе решения задачи кодирования обобщенного растрового изображения применялся существенно модифицированный алгоритм кодирования растровых изображений, лежа-

щих в основе метода SPIHT. Алгоритм кодирования SPIHT основан на том, что вейвлет-коэффициенты кодируются побитно в порядке их вклада в ПОСШ-характеристику. Последовательность, согласно которой кодируются вейвлет-коэффициенты, при этом явно не кодируется, а используется процесс разбиения множества коэффициентов на подмножества.

Первоначально имеется некое разбиение множества узлов  $\Lambda_{width,height}$  вейвлет-преобразованного растрового изображения:

$$\Lambda_{width,height} = \bigsqcup_m \tau_m,$$

где  $\bigsqcup$  — объединение непересекающихся множеств.

Далее выполняется проверка значимости порядка  $n \in N$  каждого из этих подмножеств:

$$\max_{(i,j) \in \tau_m} (|c_{ij}|) \geq 2^n, \quad (1)$$

где  $c_{ij}$  является элементом матрицы  $\tilde{C}_N$ , и если проверка дает ложь, то это означает, что все вейвлет-коэффициенты, соответствующие узлам подмножества  $\tau_m$ , незначимы. Если же проверка дает истину, то этот факт означает, что хотя бы какой-то узел подмножества  $\tau_m$  соответствует значимому вейвлет-коэффициенту. В данном случае значимое подмножество  $\tau_m$  разбивается на подмножества  $\tau_{m,p}$  по заранее принятым правилам, и та же самая проверка (1) выполняется для каждого из этих  $\tau_{m,p}$  новых подмножеств. Такое разбиение выполняется до тех пор, пока не будут выявлены все одноэлементные значимые множества, и, тем самым, пока не будут известны все значимые вейвлет-коэффициенты.

Чтобы уменьшить число сравнений при сортировке, а значит и число выводимых в поток битов, методы кодирования используют такие правила разбиения на подмножества, при которых незначимые подмножества содержали бы большее число узлов, а значимые подмножества были бы одноэлементными. В методе SPIHT правила разбиения основаны на понятии нуль-дерева, а именно — подмножества представляют собой поддеревья (в том числе и усеченные) вейвлет-преобразования. Если предположить, что гипотеза об нуль-дереве верна, то поддеревья остаются неразбитыми достаточно долго.

В традиционном методе SPIHT используются три глобальных списка для хранения множеств различных типов (для каждого из которых существуют свои правила разбиения): *список незначимых множеств* ( $\mathcal{L}_{LIS}$  — list of insignificant sets), *список незначимых пикселей* ( $\mathcal{L}_{LIP}$  — list of insignificant pixels), *список значимых пикселей* ( $\mathcal{L}_{LSP}$  — list of significant pixels). Данные списки обрабатываются схемами кодирования/декодирования в одинаковом порядке.

Далее автором решается задача пространственно-масштабного кодирования растровых изображений (обобщенных растровых изображений, все

вейвлет-коэффициенты которых активны). Авторский метод кодирования растровых изображений SS-SPIHT (*Spatially Scalable SPIHT* [23]) концептуально базируется на SPIHT, а именно — метод SS-SPIHT основан на том факте, что в нем локализуются глобальные списки метода SPIHT. Последнее означает, что для каждого из блоков вейвлет-преобразования вводятся свои три локальные списка, а именно —  $\mathcal{L}_{LIP}^{l,rc}$ ,  $\mathcal{L}_{LIS}^{l,rc}$  и  $\mathcal{L}_{LSP}^{l,rc}$ , где  $l$  — это номер вейвлет-уровня, а  $rc$  — индекс блока в данном уровне.

Кроме отмеченной выше модификации автором значительно изменен алгоритм добавления элемента в список. В методе SPIHT элементы соответствующим алгоритмом добавляются в глобальные списки. В отличие от традиционного алгоритма автором предложен алгоритм, который сначала вычисляет индекс  $\hat{l}, \hat{kr}$  того блока, в список которого необходимо добавить элемент. Далее рассматриваемый алгоритмом элемент добавляется в список того блока, индекс  $\hat{l}, \hat{kr}$  которого ранее был вычислен.

Предложенный автором метод SS-SPIHT позволяет локализовать закодированные данные (биты) относительно их масштаба и положения. Последний факт означает, что биты, выводимые при обработке каждого из блоков, обособливаются в *пакеты данных*. Кодированный поток данных (алгоритма кодирования метода SPIHT) фактически разбивается на пакеты данных. Благодаря разбиению потока на пакеты, представленная далее схема декодирования метода SS-SPIHT позволяет восстановить произвольные фрагменты растрового изображения при востребованных экспертом масштабах. Последнее достигается путем извлечения пакетов данных (из закодированного потока), которых достаточно для декодирования требуемого фрагмента.

Кратко отметим авторское расширение GI-SS-SPIHT (*General Image SS-SPIHT*) метода SS-SPIHT на случай обобщенных растровых изображений. Отличительной особенностью расширенного алгоритма является то обстоятельство, что при кодировании учитывается активность узлов соответствующих вейвлет-коэффициентов. Последний факт означает, что кодируются только активные вейвлет-коэффициенты, в то время как известные методы (например, подход, основанный на понятии региона интереса) кодируют все вейвлет-коэффициенты, в том числе и неактивные. Заметим, что в работе множество активных узлов *ActiveGrid* строится таким образом, что если родитель неактивен, то и его дети неактивны. Такой подход гарантирует корректность работы алгоритма GI-SS-SPIHT, т. е. все активные вейвлет-коэффициенты будут обработаны (закодированы).

В дополнение к указанным модификациям структура алгоритма кодирования метода GI-SS-SPIHT также претерпевает существенное изменение: блоки обрабатываются (кодируются соответствующие вейвлет-коэффициенты) согласно кривой Гильберта, что согласуется с порядком хранения блоков вейвлет-ко-

эффициентов в первичном представлении обобщенного изображения. В результате кодирования формируются пакеты данных, которые также хранятся в потоке согласно кривой Гильберта. Более того, в алгоритме кодирования авторского метода используется согласованность кривых Гильберта с разных вейвлет-поддиапазонов, т. е. после обработки блока-родителя обрабатываются блоки-дети (в соответствующем, следующем далее поддиапазоне). Представленный подход позволяет программной реализации алгоритма кодирования метода GI-SS-SPIHT использовать объем оперативной памяти, который не зависит от размера обобщенного растрового изображения (но зависит, например, от числа вейвлет-уровней).

**Повышение степени сжатия.** Согласно принятому подходу считается, что закодированный поток данных состоит только из 8-битных<sup>4</sup> чисел. Для этого разобьем последовательность из 0 и 1 на восемь подряд идущих битов. Отметим, что такая группировка битов при этом должна учитывать семантические границы в потоке данных. В частности, последние биты в каждом из пакетов данных не объединяются с битами следом идущих пакетов данных (относящихся к другому блоку). Последний факт позволяет путем извлечения пакетов (из 8-битного массива) получать данные, относящиеся к тому или иному блоку.

С каждым алгоритмом свяжем численную характеристику, которая является обратной к длине потока данных, а точнее к размеру массива. Данную характеристику будем называть *степенью сжатия*, и будем считать, что более эффективен тот алгоритм, который имеет более высокую степень сжатия (меньший поток) данных. Заметим, что при программной реализации алгоритмов 8-битные числа будут соответствовать байтам, а последовательность чисел — последовательности байтов. Отметим, что одно из преимуществ первоначального алгоритма SPIHT (а значит и GI-SS-SPIHT) заключается в том, что он и без механизмов энтропийного кодирования компактно (в смысле только что упомянутой степени сжатия) кодирует растровые изображения. Тем не менее, далее показано, как добавить энтропийные механизмы кодирования.

Рассмотрим простейший способ повышения степени сжатия пакетов данных, который заключается в уплотнении объема для их хранения. При использовании байтового потока пакеты необходимо выравнивать на байтовые границы, что влечет потерю неопределенных битов, которые расположены в последнем байте каждого пакета. В целях устранения таких потерь было решено при формировании очередного пакета (при очередном проходе), относящегося к данному блоку, записывать биты (недостающие до байта) в последний байт текущего пакета.

Авторская вероятностная модель (процесса вывода битов) метода GI-SS-SPIHT основана на том, что при инициализации вероятностной модели текущего бло-

<sup>4</sup>Для упрощения изложения материала. Вместо 8-битных чисел можно использовать любую другую битность.

ка используется соответствующая модель блока-родителя. Благодаря представленной модели сохраняется эффективность использования метода арифметического кодирования [24] даже при использовании блоков небольшого размера (например,  $8 \times 8$  пикселей). Заметим, что с уменьшением размера блока у известных методов (например, в стандарте JPEG2000) существенно снижается степень сжатия, причем в наибольшей степени это происходит из-за снижения эффективности метода арифметического кодирования.

Далее предложен новый механизм завершения прохода при кодировании очередного пакета данных. Положения, лежащие в основе алгоритма арифметического кодирования, при декодировании влекут за собой необходимость опережающего чтения данных из потока. Последний факт означает, что при декодировании пакета необходим механизм, препятствующий чтению байтов из других (следующих за текущим) пакетов. По этой причине в известных методах для определения момента завершения декодирования используется длина пакета данных, которая хранится вместе с пакетом данных.

Автором предложен механизм, который не опирается на хранение длины пакета, а основан на определении состояния (значения внутренних переменных алгоритма) алгоритма декодирования по текущему состоянию алгоритма кодирования. Последнее означает, что в процессе кодирования вычисляется тот объем данных, который алгоритм декодирования считает в опережающем режиме. Вычисленное число байтов добавляется в конец каждого пакета для обеспечения корректной работы алгоритма декодирования. Эффективность представленного подхода актуальна при сравнении объема передаваемых клиенту данных, а именно — в отличие от других методов (например, JPEG2000) информация о длине пакетов клиенту не передается, что уменьшает общий объем передаваемых данных. Отметим, что информация о длине пакетов методу GI-SS-SPIHT необходима только при формировании ответа на запрос клиента.

**Целевой поток данных.** В завершении описания схемы кодирования необходимо отметить способ формирования целевого кодированного потока данных для его эффективного использования соответствующей схемой декодирования. Его суть в том, что пакеты данных переупорядочиваются таким образом, чтобы они хранились согласно следованию кривой Гильберта (отдельно для каждого уровня  $l$  и битовой плоскости  $n$ ). Обработка пакетов согласно кривой Гильберта осуществляется путем рекурсивного спуска по соответствующим квадрантам данного вейвлет-уровня. Построим квадродерево  $\Upsilon_{n,l}$ , значения в узлах которого будут равны соответствующим суммам длин пакетов данных, относящихся к обрабатываемым квадрантам. Тогда корневой узел квадродерева  $\Upsilon_{n,l}$  будет иметь значение, равное сумме длин всех пакетов битовой плоскости  $n$  и вейвлет-уровня  $l$ , а его дети — сумме длин пакетов соответствующих квадрантов. Заметим, что в квадродереве достаточно сохранить дли-

ну трех из четырех квадрантов, так как сумма длин пакетов, которые соответствуют последнему квадранту, вычисляется как сумма длин всех пакетов данных (значение родителя) за вычетом суммы длин пакетов, которые соответствуют трем первым квадрантам. По дереву  $\Upsilon_{n,l}$  строится вектор длин, который и кодируется.

Алгоритм кодирования длин (*вложенных*) пакетов совпадает с алгоритмом, используемым в стандарте JPEG2000.

## Схема декодирования

Сначала необходимо вычислить *достаточные* для восстановления фрагмента узлы вейвлет-представления [25, 26]. Достаточность узлов означает, что по соответствующим им вейвлет-коэффициентам можно восстановить требуемый фрагмент растрового изображения. Легко вывести формулы, которые по координатам запрашиваемого фрагмента вычисляют множество достаточных узлов.

Схема декодирования фрагмента изображения состоит из серверной части (алгоритма извлечения данных из целевого потока) и клиентской части (алгоритма восстановления фрагмента по этим данным) [23]. Отметим, что на запрос пользователя передаются не сами вейвлет-коэффициенты, а пакеты данных (рис. 2, см. вторую сторону обложки), соответствующие блокам, которым эти вейвлет-коэффициенты принадлежат.

Алгоритм серверной части использует тот факт, что пакеты данных в целевом потоке хранятся согласно кривой Гильберта, что позволяет извлекать данные, соответствующие фрагменту, путем рекурсивного спуска по соответствующим квадрантам вейвлет-уровня. В частности, если среди пакетов данных, которые соответствуют квадранту, не содержится ни одного необходимого клиенту пакета, то рекурсивного спуска не проводится. Аналогично, если все пакеты данных клиенту необходимы, то все данные, соответствующие квадранту, передаются (для этого используется квадродерево  $\Upsilon_{n,l}$ ) клиенту целиком и рекурсивного спуска не проводится. Рекурсивный спуск проводится только в том случае, когда квадранту соответствуют пакеты как отмеченные, так и не отмеченные на передачу. Представленный способ хранения пакетов позволяет существенно сократить число обрабатываемых пакетов при извлечении данных, относящихся к фрагменту растрового изображения, что, в свою очередь, позволяет существенно снизить нагрузку вычислителя. На основе работ [27, 28] можно показать, что вычислительная сложность извлечения

равна  $O\left(\frac{w}{B} \log\left(\frac{D}{B}\right)\right)$ , где  $D = \max(\text{width}, \text{height})$  — линейный размер изображения;  $B$  — размер блока, используемого при кодировании, и  $w$  — ширина вырезаемого квадратного фрагмента.

Далее представлен авторский алгоритм *активной адаптации* к размеру растра извлекаемого фрагмента. При извлечении фрагмента размером  $1000 \times 1000$  пик-

селей нет смысла в ответ клиенту пересылать пакеты, относящиеся к блокам размера  $16 \times 16$  пикселей. В данном случае целесообразнее дать ответ пакетами, соответствующими блокам (квадрантам) размером  $64 \times 64$  пикселя (что естественно увеличит общий объем передаваемых данных). При вырезании фрагмента растрового изображения, во-первых, удастся сократить число рекурсивных вызовов, что уменьшит степень загрузки вычислительной системы, и, во-вторых, пакеты данных удастся объединить в непрерывные фрагменты кодированного потока, что уменьшит общее число сессий доступа к хранилищу данных. В целом, такой подход снижает нагрузку вычислительной системы.

Подводя итог изложенному выше, можно констатировать, что представлены механизмы, которые играют существенную роль в уменьшении времени извлечения фрагмента растрового изображения. Отметим, что за счет использования хранилищем данных механизмов кэширования (кэшируются близко расположенные данные, в частности ветви кривой Гильберта) алгоритм извлечения фрагментов адаптируется к размеру фрагмента неявно. Клиентский алгоритм строится обращением алгоритма кодирования, при этом объем оперативной памяти, используемый программной реализацией данного алгоритма, зависит только от размера растра и масштаба фрагмента, а не от размера растрового изображения.

### Теорема

Показатель эффективности метода GI-SS-SPIHT кодирования обобщенных растровых изображений заключается в том, что объем извлекаемых данных, соответствующих восстанавливаемым фрагментам изображения, с уменьшением размера блока при кодировании не увеличивается. В данном разделе предполагается, что в схемах кодирования/декодирования алгоритм арифметического кодирования не используется. Данное условие не является недостатком, так как на практике алгоритм арифметического кодирования использовать непринято в силу его большой вычислительной сложности. В данном разделе под *значимыми* данными (битами) понимаются только те биты, которые были выведены в результате работы основного алгоритма кодирования. Последнее означает, что при вычислении их числа не учитываются биты, которые используются для выравнивания пакетов данных на байтовые (например, 8-битные) границы.

Представленный в работе метод GI-SS-SPIHT базируется на основных положениях метода SPIHT. В связи с тем обстоятельством, что данные методы построены на схожих положениях, алгоритм кодирования метода GI-SS-SPIHT по существу переупорядочивает биты, выводимые алгоритмом кодирования метода SPIHT. Можно доказать инвариантность объема кодированного потока данных, а именно то, что число значимых битов  $z$ , выводимых в кодированный поток данных алгоритмом кодирования GI-SS-SPIHT,

не зависит от выбранного при кодировании размера блока. Можно также показать, что при кодировании растрового изображения число значимых битов, выводимых в кодированный поток данных алгоритмом кодирования SS-SPIHT (частный случай алгоритма кодирования GI-SS-SPIHT для случая растровых изображений), совпадает с числом битов, выводимых в поток алгоритмом кодирования SPIHT. Более того, число выводимых битов 1 и 0 по отдельности также будет совпадать. Для этого можно показать, что в методе SS-SPIHT изменен только порядок обработки элементов и, следовательно, факт вывода битов сохранился.

Кроме выводимых основным алгоритмом кодирования метода GI-SS-SPIHT значимых битов, целевой кодированный поток данных содержит заголовок, длины вложенных пакетов и биты, используемые для выравнивания пакетов на естественные байтовые границы. При описании механизма организации потока данных решено использовать оставшиеся биты текущего пакета данных при формировании следующего пакета, который относится к тому же блоку. Последний факт означает, что число битов, необходимых для выравнивания, зависит только от числа используемых блоков и не зависит от числа битовых плоскостей. Можно вычислить оценку общего числа блоков  $\delta$ , на которое разбито вейвлет-преобразованное изображение. При выводе оценки можно считать, что  $w, h \gg N$ , где  $w$  и  $h$  это размеры изображения, а  $N$  — число используемых вейвлет-уровней. На практике более содержательна оценка относительной избыточности:

$$\epsilon_{\%} = \frac{\delta}{wh} \lesssim \frac{1}{3s^2}, \text{ где } s \text{ — линейный размер квадратного}$$

блока, используемого при кодировании. Для блока размером  $8 \times 8$  пикселей получаем оценку 0,5 %, которая является незначительной. Следовательно, размер целевого потока в наибольшей степени зависит от способа хранения длин пакетов.

Можно доказать, что число значимых битов  $q$ , по которым можно восстановить фрагмент обобщенного растрового изображения, не зависит от выбранного при кодировании размера блока. Напомним, что при восстановлении фрагмента не используются длины пакетов данных. В этой связи объем извлекаемых данных определяется количеством значимых данных. Можно также показать, что соответствие бита фрагменту зависит только от расположения соответствующего вейвлет-коэффициента в вейвлет-преобразованном обобщенном изображении и не зависит от факта принадлежности конкретному блоку. Следовательно, общее количество выводимых значимых данных в поток, относящихся к текущему масштабу фрагмента изображения, инвариантно относительно используемого размера блока.

Исходя из отмеченных выше утверждений можно доказать справедливость утверждений следующей теоремы.

**Теорема.** *С уменьшением размера блока при кодировании общий объем данных, передаваемый клиенту на запросы фрагментов, не увеличивается, а в невырожденном случае — уменьшается.*

## Результаты и программа

Из результатов представленных тестовых испытаний следует, что метод GI-SS-SPIHT имеет явное преимущество в сравнении с JPEG2000<sup>5</sup> по степени сжатия растровых изображений, в том числе и с областями изменяемой детализации. Причем с уменьшением размера блока при кодировании преимущество метода GI-SS-SPIHT становится все более существенным. В сравнении с JPEG-LS предложенный метод показывает несколько худшие результаты. Однако отметим, что метод JPEG-LS не обладает никакими другими важными свойствами (ПОСШ-характеристика и возможность извлечения фрагментов), кроме превосходящей степени сжатия.

Использование понятия обобщенного растрового изображения (рис. 3, см. вторую сторону обложки) действительно позволяет существенно сокращать объем данных (с контролируемой пользователем степенью качества во всем изображении), необходимых для хранения растровых изображений. Отметим, что преимущество использования обобщенного растрового изображения сохраняется в том числе и при использовании других методов кодирования (например, JPEG2000). При этом заметим, что с уменьшением размера блока при кодировании преимущество метода GI-SS-SPIHT становится все более значимым.

Далее приведен анализ процесса извлечения фрагмента изображения из потока. Предложенный метод GI-SS-SPIHT имеет существенное преимущество в сравнении с методом JPEG2000. Скорость извлечения фрагмента у метода GI-SS-SPIHT существенно выше, чем у метода JPEG2000 (рис. 4), а именно — средняя скорость извлечения выше как минимум на

порядок. Например, на извлечение квадратного фрагмента размером  $1000 \times 1000$  пикселей предложенный метод GI-SS-SPIHT потратит 5 мс, а JPEG2000 — более 100 мс.

Объем извлекаемых данных с использованием метода GI-SS-SPIHT в сравнении с методом JPEG2000 в среднем ниже и он действительно уменьшается с уменьшением размера блока (согласуется с основной теоремой, имея в виду и то обстоятельство, что используется энтропийное кодирование). Более того, представленный в настоящей работе метод GI-SS-SPIHT обладает возможностью адаптации под размер извлекаемого фрагмента. Последний факт означает, что за счет объединения соседних блоков существенно повышается скорость извлечения. Например, на извлечение фрагмента размером  $1000 \times 1000$  пикселей время извлечения понизится с 8 до 2 мс. Процесс восстановления фрагмента методом GI-SS-SPIHT имеет сравнимое с методом JPEG2000 визуальное качество, а использование небольших блоков заметно его повышает. На рис. 5 представлен пример восстановления фрагмента, наглядно показывающий результат сравнения. Легко заметить, что визуальное качество изображения на рис. 5, б лучше, чем на рис. 5, а.

В настоящее время задача интерактивного анализа растровых изображений решена при определенных (существенных) ограничениях. По результатам тестовых испытаний можно сделать вывод о том, что представленный в настоящей работе метод GI-SS-SPIHT решает задачу интерактивного анализа изображений без ограничений на размеры используемых отображающих устройств и в режиме последовательной передачи данных.

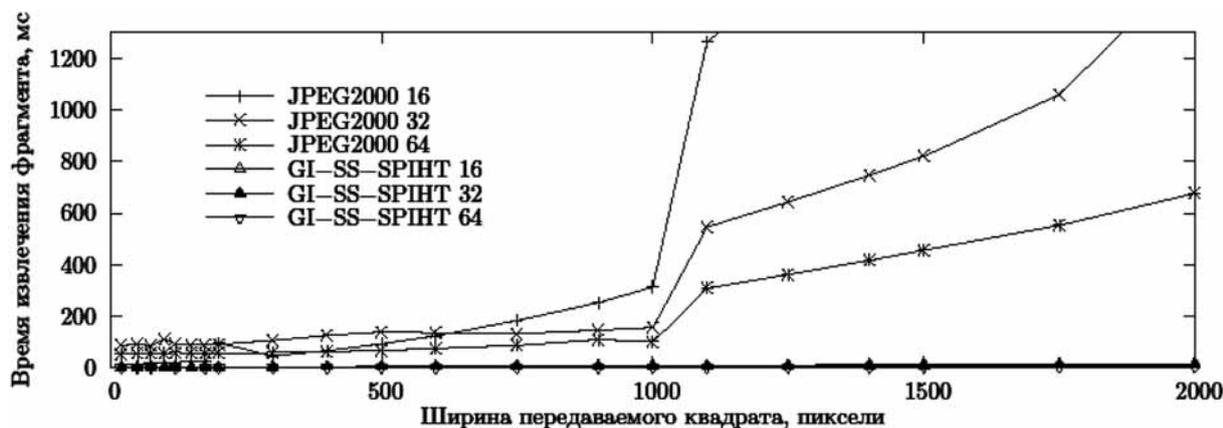
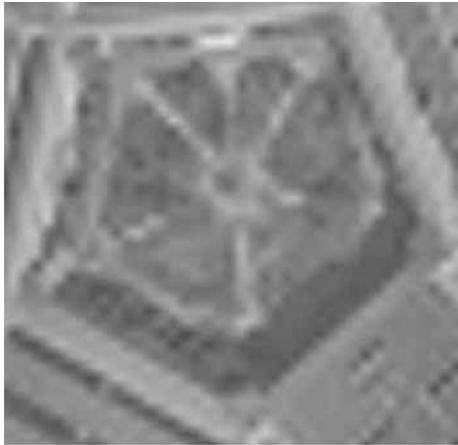


Рис. 4. Сравнение среднего времени, затраченного методами GI-SS-SPIHT и JPEG2000 на извлечение фрагмента из большого (размер  $37049 \times 8500$  пикселей) растрового изображения. Числа 16, 32 или 64 в названии элемента легенды указывают на линейный размер квадратного блока, при котором кодировалось изображение

<sup>5</sup>Для численного сравнения используется реализация Kakadu. Корректность проводимого тестового испытания гарантируется тем, что выбор данной реализации был осуществлен путем сравнительного анализа различных программных реализаций стандарта JPEG2000: Lurawave и LEAD.



а)



б)

Рис. 5. Сравнение восстановленного методами JPEG2000 (а, 25,21 дБ) и GI-SS-SPIHT (б, 26,71 дБ), фрагмента (размер  $324 \times 324$  пикселя) традиционного изображения 3.2.25 (pentagon). Восстановление фрагмента проводится при использовании кодированного потока данных, размер которого равен  $0,3 \times 324 \times 324/8$  байта (0,3 бит на пиксель). Числовые значения (дБ) даны согласно традиционной метрике ПОСШ

## Заключение

Принимая во внимание представленные выше результаты исследований, можно сделать следующие выводы.

1. Разработан новый метод кодирования и адаптивного интерактивного анализа растрового изображения изменяемой детализации, который включает эффективные модели, алгоритмы и программные механизмы кодирования/декодирования растрового изображения, использующие основные положения метода SPIHT.

2. Доказательно показано, что при формировании кодированного потока данных, оптимизированного для извлечения небольших фрагментов изображения, объем данных, достаточных для их восстановления, не увеличивается.

3. Аналитические оценки алгоритмов кодирования/декодирования растрового изображения свидетельствуют о том, что объем используемой для их реализации оперативной памяти не зависит от размера этого изображения.

4. Тестовые испытания программной реализации нового метода показали, что она позволяет последовательно извлекать из кодированного потока данные, относящиеся к определенному масштабу фрагмента изображения; адаптивно уменьшать степень загрузки вычислительной системы.

Теоретическая значимость полученных в ходе выполнения работы результатов подтверждается тем, что применение методов на базе метода SPIHT не ограничивается исключительно двумерным пространством, использованием вейвлет-преобразования и кодированием визуальных данных. Существует большое число статей, посвященных альтернативному применению метода SPIHT, в частности, применительно к трехмерному пространству, к совместному с ДКП и EBCOT кодированию и к кодированию аудиоданных. По этой причине задачу, решение которой представлено в настоящей публикации, можно рассматривать как заявку на решение аналогичной задачи при альтернативных применениях метода SPIHT.

Практическая значимость выполненной работы подтверждается тем, что программная реализация описываемого в работе метода, кроме решения поставленной цели, при кодировании растровых изображений сравнима по степени сжатия и оптимальности характеристики ПОСШ с известными методами, которые активно используются на практике. В остальных случаях (специальные классы изображений изменяемой детализации, при извлечении больших и больших фрагментов) предложенный метод показывает существенно лучший результат.

Результаты представленных в настоящей статье исследований демонстрируют эффективность предложенных моделей, методов, реализующих их программных средств кодировании растровых изображений изменяемой детализации для их адаптивного интерактивного анализа. Макет такового программного комплекса может использоваться в качестве архитектурно-технологической основы для подготовки и реализации проекта создания программного комплекса, удовлетворяющего заданным требованиям по функциональным возможностям и качеству их выполнения.

*Автор выражает благодарность д-ру физ.-мат. наук, проф. В. А. Васенину за помощь в подготовке статьи к публикации и канд. физ.-мат. наук, ст. науч. сотр. Д. В. Иванову за внимание к работе.*

## Список литературы

1. Said A., Pearlman W. A. A New, Fast and Efficient Image CODEC Based on Set Partitioning in Hierarchical Trees // IEEE Transactions on Circuits and Systems for Video Technology. 1996. June. Vol. 6, No 3. P. 243–250.

2. **ITU-T.** Information technology — Digital compression and coding of continuous-tone still image — Requirements and guidelines. Part 1. 1992. T. 81.
3. **ISO/IEC.** JPEG2000 image coding system. Part 1. 2000. 15444-1.
4. **Shapiro J. M.** Embedded image coding using zerotrees of wavelet coefficients // IEEE Transactions on Signal Processing. 1993. December. Vol. 41, No 12. P. 3445—3462.
5. **Khan E., Ghanbari M.** An efficient and scalable low bit-rate video coding with virtual SPIHT // Signal processing. Image communication. 2004. March. Vol. 19, No 3. P. 267—283.
6. **Christophe E., Mailhes C., Duhamel P.** Hyperspectral Image Compression: Adapting SPIHT and EZW to Anisotropic 3-D Wavelet Coding // IEEE Transactions on Image Processing. 2008. Vol. 17, No 12. P. 2334—2346.
7. **Малла С.** Вейвлеты в обработке сигналов. М.: Мир, 2005. 672 с.
8. **Burt P. J., Adelson E. H.** Image Data Compression With The Laplacian Pyramid // In Proc. of the conf. on pattern recognition and image processing. IEEE Computer Society Press, 1981. P. 218—223.
9. **Burt P. J., Adelson E. H.** The Laplacian pyramid as a compact image code // IEEE Transactions on Communications. 1983. Vol. 31, N 4. P. 532—540.
10. **Vitter J. S., Howard P. G.** Fast Progressive Lossless Image Compression // Image and Video Compression. Proc. SPIE. 1994. Vol. 2186. P. 98—109.
11. **Mallat S. G.** A Theory for Multiresolution Signal Decomposition: The Wavelet Representation // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1989. Vol. 11, No 7. P. 674—693.
12. **Vetterli M., Kovacevic J.** Wavelets and Subband Coding. Prentice Hall PTR, 1995. 488 p.
13. **Sudhakar R., Karthiga R., Jayaraman S.** Image Compression Using Coding of Wavelet Coefficients: A Survey // Graphics, Vision and Image Processing, 2005. Vol. 5, No 6. P. 25—38.
14. **Taubman D.** High Performance Scalable Image Compression with EBCOT // IEEE Transactions on Image Processing. 2000. Vol. 9, No 7. P. 1158—1170.
15. **Krause P. K.** Ftc — floating precision texture compression // Computers and Graphics. 2010. Vol. 34, No 5.
16. **Meng W., Qi-rui H.** An Improved Algorithm of SPIHT based on the Human Visual Characteristics // World Academy of Science, Engineering and Technology. 2006. Vol. 17. P. 119—122.
17. **Wang K., Wu C., Kong F., Zhang L.** An improved partial SPIHT with classified weighted rate-distortion optimization for interferential multispectral image compression // Chinese Optics Letters. 2008. Vol. 6. No 5. P. 331—333.
18. **Yang S. H., Cheng P. F.** Robust Transmission of SPIHT-Coded Images Over Packet Networks // IEEE Transactions on Circuits and Systems for Video Technology. 2007. May. Vol. 17, No 5. P. 558—567.
19. **Jun-Ren-D., Jar-Ferr-Y.** A Simplified SPIHT algorithm // Journal of the Chinese Institute of Engineers. 2008. Vol. 31, No 4. P. 715—719.
20. **Chew L. W., Chia W. C., Ang L., Seng K. P.** Very Low-Memory Wavelet Compression Architecture Using Strip-Based Processing for Implementation in Wireless Sensor Networks // EURASIP Journal on Embedded Systems. 2009. P. 16. URL: doi:10.1155/2009/479281.
21. **Danyali H., Mertins A.** Fully Spatial and SNR Scalable, SPIHT-Based Image Coding for Transmission Over Heterogeneous Networks // Journal of Telecommunications and Information Technology. 2003. Vol. 2. P. 92—98.
22. **Prokop H.** Cache-Oblivious Algorithms. Ph. D. thesis. MIT. 1999.
23. **Шокуров А. В.** Кодирование изображений с последующим возможным оптимальным декодированием // Фундаментальная и прикладная математика. 2007. Т. 13. С. 225—255.
24. **Witten I. H., Neal R. M., Cleary J. G.** Arithmetic coding for data compression // Communications of the ACM. 1987. June. Vol. 30, No 6. P. 520—540.
25. **Шокуров А. В., Михалев А. В.** Оптимальное использование вейвлет-компонент // Успехи математических наук. 2007. Т. 62. С. 171—172.
26. **Шокуров А. В.** Оптимальное использование компонент двоичного вейвлет-представления // Вестник Московского университета. 2009. Т. 5. С. 3—6.
27. **Proietti G.** An optimal algorithm for decomposing a window into maximal quadtree blocks // Acta Informatica. 1999. Vol. 36, No 4. P. 257—266.
28. **Moon B., Jagadish H. V., Faloutsos C., Saltz J. H.** Analysis of the Clustering Properties of the Hilbert Space-Filling Curve // IEEE Transactions on Knowledge and Data Engineering. 2001. January. Vol. 13, No 1. P. 124—141.

---

## ИНФОРМАЦИЯ

### *Продолжается подписка на журнал "Программная инженерия" на второе полугодие 2011 г.*

Оформить подписку можно через подписные агентства  
или непосредственно в редакции журнала.

Подписные индексы по каталогам:

Роспечать — 22765; Пресса России — 39795

*Адрес редакции* 107076, Москва, Стромьинский перд., д. 4,  
редакция журнала "Программная инженерия"

Тел. (499) 269-53-97. Факс: (499) 269-55-10. E-mail: prin@novtex.ru

**Ю. К. Язов**, д-р техн. наук, проф., **С. В. Соловьев**, канд. техн. наук, доц.,  
**А. В. Ступников**, стар. науч. сотр., Государственный научно-исследовательский  
испытательный институт проблем технической защиты информации Федеральной службы  
по техническому и экспортному контролю,  
e-mail: gniii@fstec.ru

## Некоторые аспекты обеспечения совместного функционирования прикладных систем поддержки принятия решений с системами электронного документооборота

*Рассматриваются два способа интеграции прикладных систем поддержки принятия решений с внедряемыми системами электронного документооборота в рамках единого автоматизированного комплекса. Раскрываются варианты реализации указанных способов сопряжения.*

**Ключевые слова:** система электронного документооборота, прикладная система, способы интеграции

В соответствии с документами [1, 2] в федеральных органах государственной власти Российской Федерации в настоящее время создано большое число систем, предназначенных для обеспечения информационно-аналитической и инструментальной поддержки принятия и выработки управленческих решений должностными лицами органов власти в различных сферах деятельности. Данные информационные системы различного назначения разработаны и введены в эксплуатацию в целях реализации требований единой государственной политики в сфере формирования информационных ресурсов и информатизации, которые определены нормативно-правовыми актами Российской Федерации, например, такими как [3–6].

Наряду с развернутыми информационными системами в органах власти в последнее время широко внедряются системы электронного документооборота (СЭД). Необходимость автоматизации делопроизводственных процессов и организации электронного документооборота на настоящее время осознана органами государственной власти и местного самоуправления практически во всех субъектах Российской Федерации. Преимущества электронного документооборота, организованного на высоком техническом уровне и эффективно используемого в деятельности органов власти, перед традиционным "бумажным" делопроизводством очевидны и сомнений не вызывают.

Согласно Концепции [1] одним из основных результатов ее реализации должно стать внедрение СЭД во всех федеральных органах государственной власти (ФОГВ), в том числе и на межведомственном уровне, а также перевод более 70 % всего объема используемых документов в электронный вид.

Одним из основных вопросов на пути успешного внедрения в практику новых систем электронного документооборота является вопрос их совместимости с уже существующими в органах власти автоматизированными системами различного назначения (информационными, расчетными, информационно-аналитическими и др.). Существующие прикладные системы и СЭД слабо интегрированы между собой, что не позволяет оптимизировать процессы осуществления документооборота, затрудняет обмен документами между подразделениями органов власти, приводит к неоправданному дублированию функций в этих системах. Таким образом, задача разработки эффективных способов интеграции прикладных систем поддержки принятия решений с СЭД является в настоящее время весьма актуальной.

Элементами функциональной структуры многих распределенных информационных систем, в том числе и прикладных систем поддержки принятия решений, могут являться разработанные отдельно друг от друга автоматизированные системы, обеспечивающие деятельность в сфере ответственности федеральных

органов государственной власти. В настоящее время существует большое разнообразие прикладных систем. К их числу относятся системы управления базами данных (СУБД), системы сбора и обработки информации, информационно-справочные системы, информационно-аналитические системы, системы общего назначения, узко специализированные системы. Все эти виды прикладных систем могут быть построены на основе различных инструментальных средств, технологий, сервисов, языков программирования.

Проведенный авторами анализ существующих платформ построения СЭД [7] показал возможность интеграции прикладных систем с СЭД и, соответственно, объединения этих систем в группы двумя способами. Первая такая группа объединяет системы посредством единого хранилища данных, вторая группа — посредством web-сервисов и единой системы нормативно-справочной информации (НСИ). В соответствии с таким подходом к классификации в состав информационной системы, кроме прикладных подсистем, необходимо также включить один или несколько вспомогательных компонентов (модулей), которые будут обеспечивать сопряжение разработанных средств с СЭД в интересах обеспечения их совместного функционирования на единой платформе. Таковыми подсистемами являются:

- подсистема web-сервисов;
- подсистема единой системы нормативно-справочной информации;
- подсистема единого хранилища данных.

Подсистема web-сервисов обеспечивает обмен данными всех прикладных подсистем с единым хранилищем путем связывания содержащихся в нем данных с программными модулями, базами данных и/или непосредственно с автоматизируемыми процессами. Она предоставляет также интерфейсы для обмена данными с единым хранилищем, функционирующие на базе web-сервера (например, Apache) и сервера приложений (например, Apache Tomcat).

Единая система НСИ обеспечивает предоставление нормативной и справочной информации, которая используется всеми автоматизированными подсистемами, интегрируемыми в единую информационную систему.

Единое хранилище данных (ХД) предназначено для консолидации информации, хранящейся и обрабатываемой в базах данных отдельных автоматизированных подсистем. Таким образом, в рамках интегрированного комплекса создается единое информационное пространство, в рамках которого обеспечивается хранение, импорт и экспорт данных интегрируемых подсистем.

На основе перечисленных выше способов сопряжения могут быть применены следующие варианты интеграции ранее созданных автоматизированных систем с СЭД:

- посредством размещения документов отдельных автоматизированных систем в системе электронного документооборота (вариант 1);

- путем создания web-сервисов и программного обеспечения автоматизированных систем, поддерживающего взаимодействие с web-сервисами (вариант 2);
- поддержкой экспорта данных автоматизированных систем в единое хранилище (вариант 3);
- путем создания web-приложений, реализующих функциональные возможности автоматизированных систем (вариант 4).

Объединение автоматизированных систем на основе варианта 1 предполагает обработку всех типов документов, используемых при совместной работе прикладных систем с СЭД. Такая работа организуется на базе единого ХД с предоставлением пользователю интерфейсов для размещения документов в едином хранилище, организации логических маршрутов работы с документами и их экспорта на твердые носители.

Реализация варианта 2 предполагает:

- создание приложений, реализующих обмен данными всех подсистем с единым хранилищем путем связывания содержащихся в нем данных с программами, базами данных и/или непосредственно с автоматизируемыми процессами, а также предоставляющих интерфейс для обмена с единым хранилищем;
- модификацию специального программного обеспечения интегрируемой автоматизированной системы для взаимодействия с приложениями, реализующими обмен данными.

Интеграция систем с использованием варианта 3 предполагает создание механизма импорта/экспорта данных автоматизированных систем на базе единого хранилища данных. При этом существующее программное обеспечение прикладной системы, которая интегрируется с использованием данного варианта, остается практически без изменений.

Использование варианта 4 предполагает реализацию функций интегрируемой автоматизированной системы путем создания набора сценариев, функционирующих на базе сервера приложений.

Использование перечисленных вариантов реализует следующие принципы интеграции разработанных прикладных систем с СЭД: однородность операционной среды; открытость приложений; однородность данных; взаимодействие автоматизируемых процессов; единое управление доступом к приложениям и данным. В зависимости от специфики интегрированной системы могут применяться различные сочетания рассмотренных вариантов интеграции.

Схема функциональной структуры и возможный состав интегрированной системы, которая создается с применением всех перечисленных выше вариантов интеграции СЭД с прикладными подсистемами поддержки принятия решений, представлены на рис. 1.

Интеграция прикладных систем с СЭД посредством web-сервисов предполагает запуск подсистем из пользовательского интерфейса программного обеспечения электронного документооборота при условии, что системы находятся в одной локальной сети, либо имеют между собой физическое соединение, которое

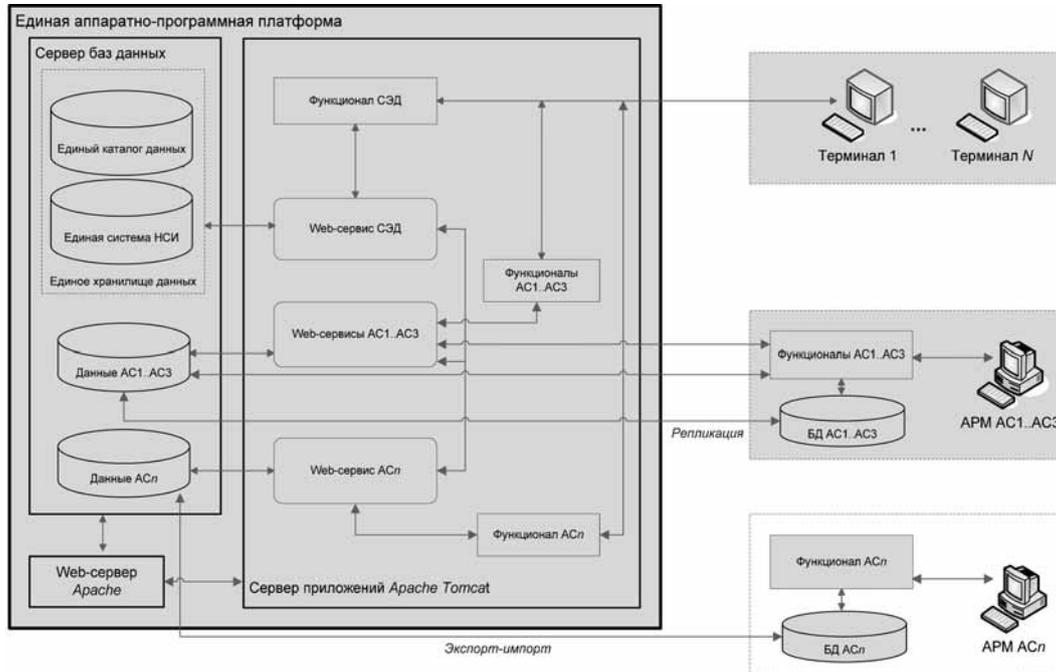


Рис. 1. Функциональная структура интегрированной системы

может быть реализовано по технологии создания защищенных туннелей (*Virtual Private Network — VPN*). Реализация этого варианта интеграции систем строится на современных технологиях многоуровневых вычислений по модели клиент/сервер и на разработке корпоративных приложений на серверной части согласно модели вычислений J2EE (*Kava 2 Enterprise Edition*). Подсистема web-сервисов реализует функцию обмена данными всех подсистем, интегрируемых в информационную систему, с единым хранилищем данных путем их связывания.

Результатом разработки web-сервиса является программный модуль или набор программных сценариев (написанных, например, на языке XML), который осуществляет связывание данных с программами, базами данных или непосредственно с процессами, реализуемыми с помощью ранее разработанных автоматизированных систем. Web-сервисы представляют собой эффективные платформонезависимые механизмы передачи данных и интерактивного взаимодействия приложений.

Возможным примером реализации такого способа интеграции является использование среды выполнения web-сервисов Apache Axis 2 (AXIS2), основанной на новой архитектуре, объединяющей опыт ее предшественников — Apache SOAP и Apache Axis. Эта среда является уже третьим поколением средств выполнения web-служб, обеспечивающим высокую скорость, устойчивость, эффективность и надежность. Общий вид AXIS2 представлен на рис. 2.

Одним из достоинств AXIS2 является объектная модель XML-документа — AXIOM. Суть подхода к интеграции на основе AXIOM заключается в том, чтобы формировать не полное представление XML-документа, а частичное, поскольку полное представление зачастую не требуется. Это обстоятельство позволяет экономить время как на обработку запроса, так и на затрачиваемые ресурсы (оперативную память).

Подсистема единого ХД обеспечивает реализацию следующих функций:

- консолидация информации, хранящейся в базах данных и обрабатываемой в автоматизированных подсистемах;
- создание единого информационного пространства в рамках системы поддержки принятия решений;
- хранение, импорт и экспорт данных интегрируемых автоматизированных систем, разработанных для обеспечения деятельности ФОГВ.

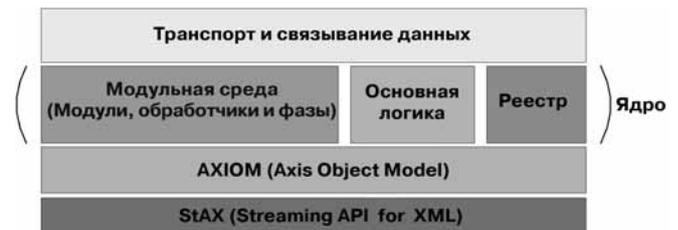


Рис. 2. Общий вид платформы AXIS2



Рис. 3. Схема подсистемы ХД в рамках СЭД

Подсистема единого хранилища данных содержит:

- структуры данных, обеспечивающих хранение информации;

- интерфейс подсистемы ХД, предназначенный для работы пользователя с данными подсистемы.

Структура подсистемы единого ХД представлена на рис. 3. Примером применения единого хранилища данных является СЭД на базе платформы Documentum. Такая система обеспечивает интерфейсы размещения документов в едином хранилище, организацию логических маршрутов работы с документами и их экспорт

на твердые носители. Платформа Documentum спроектирована для создания web-приложений по управлению контентом в средах интранет, экстранет и/или Интернет. Она может открывать часть или весь набор своих функциональных возможностей, реализованных в виде web-сервисов. Платформа Documentum позволяет работать с произвольными типами контента (без каких-либо ограничений на формат файла). Структурно контент может быть одним документом, частью документа или папкой, пополняемой документами по мере работы с ними сотрудников организации.

Подсистема ХД организуется следующим образом:

- структура данных подсистемы ХД входит в хранилище СЭД наравне со стандартными структурами данных Documentum и доступна для традиционных средств администрирования и разработки;
- пользовательский интерфейс подсистемы ХД разрабатывается как составная часть пользовательского интерфейса СЭД. В то же время данные подсистемы СЭД остаются доступными для пользователя через стандартный пользовательский интерфейс СЭД;
- взаимодействие пользовательского интерфейса подсистемы ХД и структур данных происходит через универсальный механизм вызовов методов подсистемы web-сервисов.

Структуру данных подсистемы СЭД можно изобразить в виде диаграммы классов, представленной на рис. 4. Под классом понимается часть программного кода, написанного на объектно-ориентированном языке.

Классы `dm_sysobject`, `dm_folder`, `dm_document` относятся к стандартным классам Documentum. Классы с префиксом `<udss>` относятся к подсистеме ХД. Класс `udss_area` создан для хранения информации о существующих и вновь создаваемых областях приложений. Такая область будет создаваться для каждой

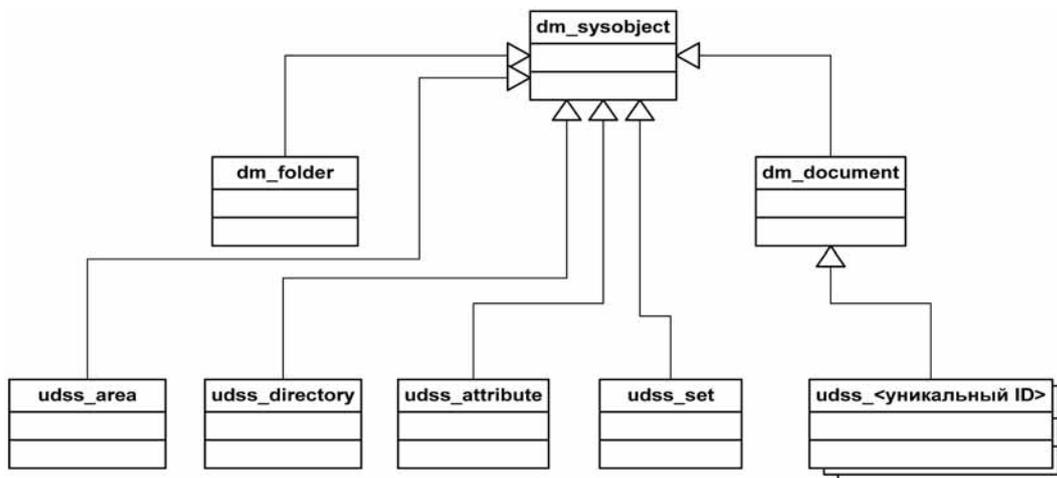


Рис. 4. Диаграмма наследования классов структуры хранения подсистемы ХД

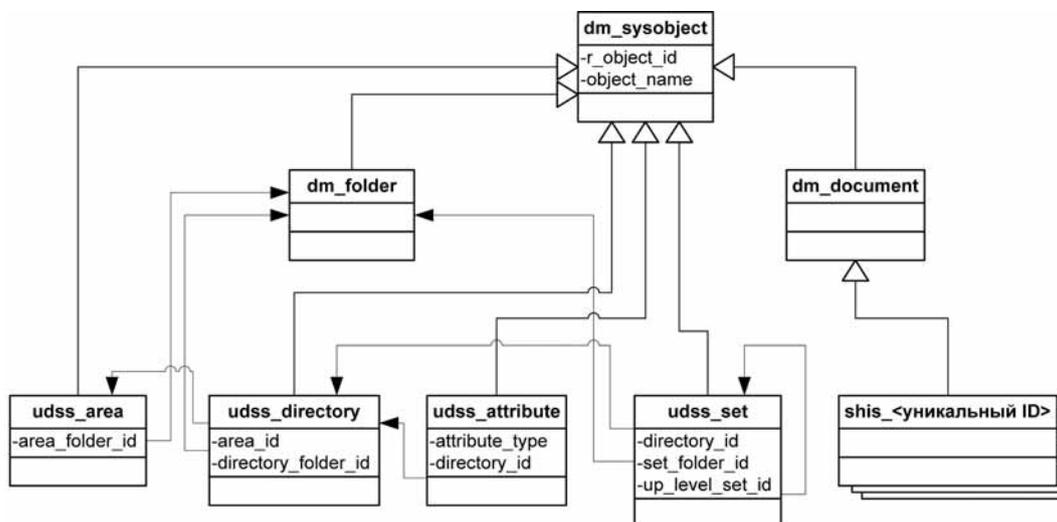


Рис. 5. Диаграмма связей структуры хранения подсистемы ХД и стандартных структур данных СЭД Documentum

автоматизированной системы, предоставляющей информацию в единую информационную систему.

Класс `udss_directory` создан для хранения информации о существующих и вновь создаваемых описаниях информационных объектов в рамках конкретной области приложения. При создании объекта `udss_directory` создается дополнительный класс, обозначенный на диаграмме как `udss_<уникальный ID>`.

Класс `udss_attribute` предназначен для хранения информации об атрибутах информационных объектов, описанных в `udss_directory`. При создании объекта `udss_attribute` создается дополнительный атрибут в классе `udss_<уникальный ID>`.

Класс `udss_set` обеспечивает возможность создания древовидной иерархии информационных объектов.

Класс `udss_<уникальный ID>` создается при добавлении нового объекта `udss_directory`. В качестве уникального ID в данном случае используется значение атрибута `r_object_id` нового объекта `udss_directory`. Этот идентификатор является уникальным в рамках всего хранилища Documentum.

Создание новых классов направлено на поддержку возможности расширения набора информационных объектов, используемых автоматизированными системами. Такое решение позволит быстро проводить поиск необходимой информации за счет ее разделения по соответствующим таблицам в зависимости от требуемого типа информационного объекта.

Для каждого класса определен набор атрибутов (рис. 5), связывающий данные подсистемы НСИ и стандартные структуры данных Documentum.

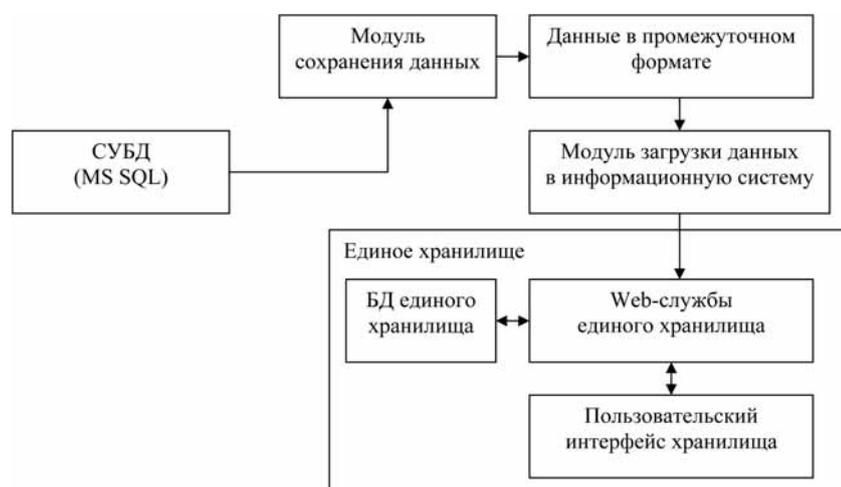


Рис. 6. Схема интеграции прикладной системы в единой информационной системе

В качестве примера на рис. 6 представлена схема интеграции прикладной системы (СУБД) в единой информационной системе. Интеграция осуществляется посредством экспорта данных в единое хранилище. Для этого разрабатываются два программных модуля. Первый модуль обращается к СУБД (MS SQL) и сохраняет данные в некотором промежуточном формате. Второй модуль работает на стороне информационной системы и размещает данные из промежуточного формата в едином хранилище путем вызова соответствующих web-служб. Таким образом, осуществляется односторонняя передача данных из СУБД в информационную систему. Просмотр данных в едином хранилище обеспечива-

ется с помощью стандартных средств пользовательского интерфейса единого хранилища.

Сложность выбора языка программирования для разработки такого специального математического программного обеспечения, предназначенного для интеграции прикладных систем с СЭД, обусловлена следующими требованиями к средству разработки:

- наличие в языке программирования реализованной концепции объектно-ориентированного программирования (в интересах реализации единых структур данных и экспорта данных автоматизированных систем в единое хранилище);
- наличие в языке программирования реализованной кроссплатформенности на уровне выполнения (с учетом того обстоятельства, что исходные автоматизированные системы могут быть построены на различных программно-аппаратных платформах, а также в интересах создания полнофункциональных web-сервисов и web-приложений);
- обеспечение простого и доступного пользователю режима взаимодействия приложений, написанных на языке программирования, с выбранной СЭД.

В качестве такого языка целесообразно использовать язык программирования Java, удовлетворяющий всем определенным в работе [8] требованиям к языку программирования для разработки средств интеграции.

С использованием языка программирования Java разрабатывается специальный механизм, который направлен на поддержку совместного функционирования систем. В его основе реализация технической, информационной, лингвистической и организационной совместимости прикладных систем с СЭД. Эти требования обеспечиваются проведением работ по интеграции автоматизированных систем на уровне платформ, данных, приложений, доступа пользователей к приложениям, автоматизируемых процессов.

Результатом такой интеграции является повышение эффективности использования прикладных систем совместно с СЭД за счет:

- обеспечения функционирования на единой аппаратно-программной платформе;
- создания единой "точки входа" для получения всей необходимой пользователям информации;
- использования единых входных и выходных данных.

Таким образом, в статье предложены несколько эффективных способов интеграции прикладных систем поддержки принятия решений, которые уже разработаны и введены в эксплуатацию во многих информационных системах ФОГВ, с внедряемыми системами электронного документооборота. Раскрыты

варианты реализации этих способов, обеспечивающих сопряжение автоматизированных систем, разработанных для поддержки деятельности должностных лиц ФОГВ, в едином автоматизированном комплексе на базе СЭД, функционирующей на программной платформе Documentum. Рассмотрены два способа группировки этих систем в едином автоматизированном комплексе на основе создания общего хранилища данных, а также интегрируемых посредством web-сервисов и подсистемы НСИ.

Применение предлагаемых подходов к интеграции эксплуатируемых автоматизированных систем с внедряемыми СЭД обеспечит удобство их использования, сокращение затрат на дальнейшее развитие автоматизированных систем путем исключения дублирующих функций и преемственность в использовании и развитии информационных технологий в деятельности работы органов власти, ведомств и организаций.

Предложенные способы и соответствующие им варианты реализации, обеспечивающие сопряжение систем поддержки принятия решений в едином автоматизированном комплексе на базе СЭД, функционирующей на программной платформе Documentum, апробированы и на практике реализованы в информационно-аналитической системе специального назначения.

#### Список литературы

1. **Концепция** использования информационных технологий в деятельности федеральных органов государственной власти до 2010 года, утверждена распоряжением Правительства Российской Федерации от 27 сентября 2004 г. № 1244-р, г. Москва.
2. **Стратегия** развития информационного общества в Российской Федерации, утверждена Указом Президента Российской Федерации от 7 февраля 2008 г. № 212, г. Москва.
3. **Федеральная** целевая программа "Электронная Россия", утверждена постановлением Правительства Российской Федерации от 28 января 2002 г. № 65.
4. **Федеральный** закон Российской Федерации от 27 июля 2010 г. № 210-ФЗ "Об организации предоставления государственных и муниципальных услуг".
5. **Федеральный** закон Российской Федерации от 27 июля 2006 г. № 149-ФЗ "Об информации, информационных технологиях и о защите информации".
6. **Распоряжение** Правительства Российской Федерации от 12 февраля 2011 г. № 176-р "Об утверждении плана мероприятий по переходу федеральных органов исполнительной власти на безбумажный документооборот при организации внутренней деятельности".
7. **Системы** электронного документооборота в органах государственной власти Российской Федерации федерального уровня. Аналитический отчет. М.: CNews Analytics, 2008 г.
8. **ГОСТ Р ИСО/МЭК 12207—99.** Информационная технология. Процессы жизненного цикла программных средств.

## Механизмы аутентификации Grid в web-приложениях и сервисах

*Рассматривается подход к построению средств аутентификации web-приложений и сервисов, основанный на механизмах, реализованных в инфраструктуре безопасности Grid. Такой подход позволяет интегрировать средства, построенные на основе современных технологий, в уже существующие распределенные вычислительные Grid-системы.*

**Ключевые слова:** распределенные вычислительные системы, Grid-системы, аутентификация, прототип программного средства

### Введение

В последнее время значительно вырос интерес к распределенным информационно-вычислительным системам, в первую очередь к построенным на основе методологии Grid [1–3]. Такие системы призваны обеспечить среду для поддержки исследований в самых разных научных направлениях, для автоматизации процессов в промышленности, бизнес-среде и др. При этом пока наблюдается существенный технологический разрыв между информационной и вычислительной составляющими таких систем. С одной стороны, вычислительная составляющая существующих и активно используемых Grid-систем основана на технологиях, которые, с точки зрения современных требований к ним, уже можно рассматривать как устаревшие. К таким, например, можно отнести реализованные в пакете инструментальных средств Globus Toolkit механизмы обеспечения безопасности GSI<sup>1</sup>, созданные под влиянием технологий, традиционно используемых в корпоративных интранет-системах, а также протоколы, поддерживающие основные сервисы промежуточного уровня. С другой стороны, эволюционный путь развития Grid-технологий привел к тому, что такие средства уже получили

широкое распространение. Как следствие, потребность дальнейшего совершенствования программной платформы, возникающая из необходимости решения новых задач, сталкивается с существенным препятствием в виде требований совместимости программного обеспечения платформы с уже существующими Grid-средами.

Основным направлением развития Grid сейчас является усовершенствование сервисно-ориентированной архитектуры промежуточного программного обеспечения. Этот путь уже отмечен несколькими масштабными изменениями. Упомянутый ранее пакет программных средств Globus Toolkit, который является одним из основных при создании современных, эксплуатируемых в промышленном режиме Grid-систем, уже дважды (в своей третьей и четвертой версиях) предоставлял новую реализацию платформы для построения сервисно-ориентированной среды распределенных вычислений. Однако этот процесс еще не завершен. Перед выпуском пятой версии пакета его разработчики в очередной раз были вынуждены отказаться от дальнейшего развития целого ряда ранее используемых в его составе технологий<sup>2</sup>. Во

<sup>1</sup> <http://www.globus.org/security/overview.html>

<sup>2</sup> <http://lists.globus.org/pipermail/announce/2009-March/000054.html>, <http://lists.globus.org/pipermail/gt-user/2009-October/008578.html>

многим такое положение дел вызвано слишком быстрым развитием технологий на этом направлении, что не позволяет надежно опереться на какие-либо программные средства и обеспечить поддержку этих средств на достаточно продолжительном промежутке времени. Однако учитывая актуальность и важность этих технологий, есть основания полагать, что в некоторой перспективе решение задачи выбора технологий будет найдено.

При этом уже сейчас понятно, что основную роль в этом процессе будут играть web-технологии, так как именно они являются основным средством реализации современных распределенных систем, обеспечивающих взаимодействие различных участников информационного обмена.

В этой связи актуальной представляется интеграция web-технологий с уже существующими Grid-комплексами в целях отработки новых технологических возможностей и как следствие расширение функциональности этих комплексов. В первую очередь, такая интеграция сводится к распространению одного из ключевых механизмов Grid — единого входа в систему — на протокол HTTP [4], являющийся основой web-взаимодействия.

## 1. Методы аутентификации для web-приложений в Grid

Аутентификация в Grid строится на основе инфраструктуры открытых ключей. Каждый пользователь системы обладает персональным X.509-сертификатом [5, 6] продолжительного срока действия, который позволяет получать доступ к ресурсам распределенной Grid-среды. Предоставление доступа к ресурсам на основе сертификатов обеспечивает возможность единого входа в систему (*single sign-on*), поддерживая таким образом одно из основополагающих положений концепции Grid.

В целях безопасности персональный сертификат не используется при доступе напрямую, поскольку его утечка может дать злоумышленнику все возможности, доступные для пользователя. Вместо этого на основе персонального сертификата строится другой сертификат с ограниченными возможностями и сокращенным сроком действия, на основе которого обеспечивается доступ к ресурсам. Такой сертификат называется прокси-сертификатом.

Сложность интеграции описанных механизмов для доступа к web-ресурсам состоит в том, что web-браузеры не поддерживают возможность непосредственного использования прокси-сертификатов Grid. Это обстоятельство вынуждает использовать "обходные" варианты, отличающиеся по своим характеристикам

и вносящие различные ограничения на функциональные возможности.

В числе наиболее распространенных методов можно отметить два. Первый из них основан на использовании X.509-сертификатов Grid для установления защищенного соединения по протоколу HTTPS [7]. Для реализации этого метода пользователю необходимо импортировать свой персональный сертификат в браузер, с которого будет осуществляться доступ к ресурсам. В связи с тем, что импорт требует дополнительных действий со стороны пользователя, в браузер устанавливается основной сертификат, а не прокси, используемый для входа в систему. Таким образом, при доступе через браузер к web-приложению отсутствует информация о тех правах доступа, которыми пользователь обладает в данный момент, а для передачи этих данных используется внешний механизм делегирования. Отмеченный способ применяется в широко распространенном средстве GridSite<sup>3</sup>.

Вторым методом является использование традиционных для web способов аутентификации, таких как OAuth [8] или OpenID с автоматизированной трансляцией данных о пользователе во временный сертификат для доступа. При таком подходе Grid-инфраструктура в значительной степени изолируется от пользователя, что ограничивает ее потенциальные возможности. Более того, в этом случае хотя и обеспечиваются более широкие возможности по интеграции с web-приложениями, но процесс сопровождается практически полным отказом от непосредственного доступа к Grid-ресурсам.

Предлагаемое в настоящей статье решение качественным образом отличается от изложенных выше. Отличие связано с тем, что механизмы GSI предоставляют реализацию обобщенного интерфейса сервисов безопасности GSSAPI [9]. В связи с этим обстоятельством можно использовать метод аутентификации HTTP Negotiate [10], изначально предполагавшийся к применению в корпоративных интранет-системах. Встроенная поддержка механизма аутентификации Negotiate на основе интерфейса GSSAPI в браузер Mozilla Firefox, который широко используется пользователями Grid-систем, позволяет прямо обращаться к механизмам GSI. Кроме того, настройка этого механизма может быть выполнена на уровне операционной системы без установки дополнительных программных средств на стороне клиента. Этот факт выгодно отличает данный метод от перечисленных ранее.

Недостатком подхода можно считать необходимость защиты канала средствами HTTPS и относи-

<sup>3</sup><http://www.gridsite.org>

тельно большое число дополнительных запросов, которые обрабатываются при каждом обращении клиента к серверу. Однако учитывая общую постановку задачи, эти недостатки не являются существенными и не ограничивают возможность применения предлагаемого решения.

## 2. Реализация и практические испытания

Испытания программного средства, реализующего предлагаемое автором и описанное выше решение, проводились на специально созданном для этой цели макете. Макет был построен на вычислительных установках под управлением Debian GNU/Linux, входящих в состав экспериментального Grid-полигона [11], включая средства разработки и оперативного запуска программного обеспечения, в том числе — инфраструктуру X.509-сертификатов.

Основу механизма представляет модуль аутентификации `mod_auth_gss`, разработанный автором на основе модуля аналогичного назначения для протокола аутентификации Kerberos (`mod_auth_kerb`) [12, 13]. В обоих случаях используется библиотека, реализующая интерфейс GSSAPI, вследствие чего значительная часть кода модуля остается без изменений.

Так как для обслуживания запроса необходимо несколько последовательных обращений клиента к серверу, потребовалось ассоциировать контекст безопасности с соединением, что не являлось необходимым для Kerberos и не было изначально реализовано в модуле `mod_auth_kerb`.

Модуль оформлен в виде установочного пакета для операционной системы Debian GNU/Linux и использует существующую в этой операционной системе инфраструктуру поддержки web-сервера Apache.

**Конфигурация клиента.** На клиентской машине было установлено программное обеспечение Debian GNU/Linux версии 6.0. Для поддержки описанного метода аутентификации потребовались следующие компоненты: элементы инфраструктуры открытых ключей Grid, включая набор сертификатов используемых удостоверяющих центров и утилиты для создания прокси-сертификатов пользователя; web-браузер Icesweasel 3.5.13 и 3.6.12<sup>4</sup>; библиотека `libglobus_gssapi-gsi4`, входящая в состав дистрибутива.

Настройка браузера заключалась в изменении трех конфигурационных переменных, которые были

<sup>4</sup>Debian Icesweasel — полный аналог Mozilla Firefox, переименованный авторами по требованиям лицензионных ограничений из-за наличия специфических для Debian изменений.

установлены с помощью системного файла `/etc/icesweasel/pref/gsi.js` следующего содержания:

```
pref("network.negotiate-auth.gsslib",
"/usr/lib/libglobus_gssapi_gsi.so.4");
pref("network.negotiate-auth.using-native-gsslib", false);
pref("network.negotiate-auth.trusted-uris", "https://");
```

Здесь явным образом задается имя библиотеки, реализующей интерфейс GSSAPI, и разрешается использование этого метода для ресурсов, доступных по протоколу HTTPS. Следует отметить, что неправильная настройка доверия к HTTPS-ресурсам может приводить к неприятным последствиям из-за возможности атак типа "человек посередине". В этой связи при практическом использовании предлагаемого средства возникает необходимость ограничить список доверенных корневых сертификатов или уточнить список ресурсов, к которым может быть предоставлен доступ. Однако для предлагаемого решения на его нынешнем этапе отмеченное обстоятельство не представляет серьезной угрозы. Более того, указанная рекомендация применима к использованию протокола HTTPS вообще, а не только в данном случае.

В процессе тестирования автором была обнаружена ошибка в реализации метода аутентификации в браузере Icesweasel версии 3.5.11 и предложен способ ее устранения<sup>5</sup>, который был принят разработчиками и включен в версию готовящегося на тот момент к выпуску дистрибутива. Браузер Mozilla Firefox 3.6 этой ошибки не содержит, поэтому есть основания полагать, что в других современных дистрибутивах метод также будет работать.

**Конфигурация сервера.** На стороне сервера использовался точно такой же дистрибутив, как и на стороне клиента. Установленное программное обеспечение включало: web-сервер Apache 2.2 с модулем аутентификации GSI и необходимыми библиотеками Globus Toolkit; элементы инфраструктуры сертификатов Grid; тестовое приложение на основе web-фреймворка Django с необходимыми модулями; модуль `mod_wsgi` для подключения Django-приложения к серверу Apache.

У Apache была создана стандартная конфигурация сервера с поддержкой SSL. При этом параметр `SSLVerifyClient` был выставлен в значение `none`, что означает отсутствие необходимости аутентификации со стороны клиента на уровне протокола HTTPS.

Контроль доступа к ресурсам обеспечивался в зависимости от их типа. Значение по-умолчанию, указанное явным образом в директиве `<Directory/>`,

<sup>5</sup><http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=590040>

запрещало обращение к ресурсам, не указанным в конфигурации сервера. Доступ к вспомогательным данным приложения (стилевым файлам и изображениям) был открыт для всех пользователей, чтобы снизить нагрузку на клиента и сервер.

Механизм аутентификации на основе GSI использовался при обращении к скриптам CGI и коду Django-приложения. Настройка доступа выглядела следующим образом:

```
ScriptAlias /CGI-bin/srv/https/bin/CGI-bin
WSGIScriptAlias / /srv/ https/bin/WSGI.py
<Directory/srv/https/bin>
Options +ExecCGI
GssServiceName "host"
Require valid-user
AuthType GSSAPI
AuthName "grid.su"
< /Directory >
```

Заметим, что здесь существует возможность явно указать имя пользователя в директиве Require. В этом случае указывается DN (*Distinguished Name*) сертификата пользователя, при том, что аутентификация проводится на основе прокси-сертификата. Пример такой директивы:

```
Require user "/C=RU/ST=Moscow/O=MSU
MRI/OU=Grid/CN=Alexander Inyukhin"
```

Это же имя устанавливается сервером Apache в переменную REMOTE\_USER, передаваемой скриптам CGI и приложениям WSGI таким же образом, как и в случае аутентификации с помощью других методов.

Отдельного упоминания заслуживает необходимость присутствия двух X.509-сертификатов для обеспечения работы сервера. Первый из них необходим для аутентификации сервера по протоколу HTTPS. Значение DN этого сертификата должно содержать элемент вида CN = *hostname*, где *hostname* — сетевое имя сервера, к которому осуществляется соединение. Второй сертификат необходим для аутентификации сервера через GSI. У этого сертификата формат CN-компонента должен выглядеть следующим образом: CN = *host/hostname*. Здесь *hostname* имеет то же значение, что и у предыдущего сертификата, а предшествующее ему *host* указывает на тип сервиса. Этот же тип указан в директиве GssServiceName конфигурационного файла.

С точки зрения протокола, в качестве типа сервиса ожидается значение http. Однако из-за особенностей обработки имен сертификатов в библиотеке GSSAPI GSI это значение распознается некорректно.

Следует отметить, что с теоретической точки зрения возможно совмещение двух сертификатов в од-

ном, используя расширенный атрибут subjectAltName, определенный в спецификации X.509 и позволяющий указывать в сертификате альтернативные имена идентифицируемых объектов. Однако такая возможность на практике не проверялась.

### 3. Дальнейшее развитие

Результаты тестовых испытаний на полигоне в представленной выше конфигурации демонстрируют, что предлагаемые механизмы аутентификации GSI для web-приложений реализуемы и эффективны. Однако востребованные на практике сценарии аутентификации и авторизации пользователей могут быть более сложными, чем допускает существующая реализация подхода. Одной из таких возможностей является делегирование полномочий пользователя приложению, что необходимо для выполнения процессов от имени пользователя.

Предлагаемое решение допускает реализацию указанных выше функций, однако его применение на практике требует более глубокой проработки. Во-первых, необходимо обеспечить контроль использования делегированного прокси-сертификата на стороне сервера, поскольку делегирование должно проводиться только при доступе к тем ресурсам, для которых оно необходимо. Во-вторых, нужно разработать метод передачи делегированного сертификата приложению, которое будет с ним работать дальше. В-третьих, параметры делегирования, задаваемые браузером пользователя, такие как срок действия прокси-сертификата, могут не соответствовать ожиданиям пользователя и потребностям приложений.

Другой важной возможностью является проверка принадлежности пользователя виртуальной организации [1—3]. Приложение может использовать эту информацию для реализации групповых политик доступа к ресурсам. С практической точки зрения эта информация может быть извлечена из сертификата, участвующего в установлении соединения, в том числе и из сертификата, полученного через процедуру делегирования. Однако реализация такого механизма в приложении может оказаться затруднительной. В этой связи возможности модуля должны включать не только аутентификацию, но и основные механизмы авторизации.

Еще одним важным применением предлагаемого метода аутентификации является возможность трансляции Grid-идентификатора пользователя в идентификаторы OpenID и OAuth, которые широко применяются в современных web-приложениях и сервисах. Такой способ позволит подключать к Grid-системам разнообразные сторонние сервисы, число которых постоянно увеличивается.

## Заклучение

Несмотря на ряд технологических ограничений, предлагаемое решение обладает существенными преимуществами, основное из которых заключается в простоте развёртывания программного средства как на стороне клиента, так и на стороне сервера. Настройка клиентской части не требует установки дополнительных программных средств, а конфигурирование может быть выполнено как на уровне пользователя, так и на уровне системы, к ресурсам которой будет осуществляться доступ.

Предлагаемое программное средство прошло тестовые испытания на специально созданном для этой цели макете в рамках работ по развитию экспериментального Grid-полигона [11], имея в виду применение в его составе современных web-технологий.

## Список литературы

1. **Foster I.** What is the Grid? A three point checklist // GRID today. 2002. July. Vol. 1. No 6.
2. **Foster I., Kesselman C., Nick J. M., Tuecke S.** The physiology of the grid: An open grid services architecture for distributed systems integration: Tech. rep. 2002. URL: <http://www.globus.org/research/papers/ogsa.pdf>.
3. **Foster I. T.** The anatomy of the grid: Enabling scalable virtual organizations // Proc. of the 7th International Euro-Par Conf.

Manchester on Parallel Processing. Euro-Par'01. London. UK. 2001. P. 1—4.

4. **Fielding R., Gettys J., Mogul J.** et al. Hypertext Transfer Protocol — HTTP/1.1. RFC 2616 (Draft Standard). 1999. June. Updated by RFCs 2817, 5785. URL: <http://www.ietf.org/rfc/rfc2616.txt>.

5. **Foster I., Kesselman C., Tsudik G., Tuecke S.** A Security Architecture for Computational Grids // ACM Conference on Computers and Security. 1998. P. 83—91.

6. **Welch V., Foster I., Kesselman C., Mulmo O., Pearlman L., Tuecke S., Gawor J., Meder S., Siebenlist F.** X.509 Proxy Certificates for Dynamic Delegation // 3rd Annual PKI R&D Workshop, 2004.

7. **Rescorla E.** HTTP Over TLS. RFC 2818 (Informational). 2000. May. Updated by RFC 5785. URL: <http://www.ietf.org/rfc/rfc2818.txt>.

8. **Hammer-Lahav E.** The OAuth 1.0 Protocol, RFC 5849 (Informational). 2010. April. URL: <http://www.ietf.org/rfc/rfc5849.txt>.

9. **Linn J.** Generic Security Service Application Program Interface Version 2, Update 1. RFC 2743 (Proposed Standard). 2000. January. Updated by RFC 5554. URL: <http://www.ietf.org/rfc/rfc2743.txt>.

10. **Jaganathan K., Zhu L., Brezak J.** SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows. RFC 4559 (Informational). 2006. June. URL: <http://www.ietf.org/rfc/rfc4559.txt>.

11. **Васенин В. А., Инюхин А. В., Шевелев М. В.** Фрагмент территориально-распределенной информационно-вычислительной среды на основе методологии grid // Информационные технологии. 2010. № 1. С. 63—64.

12. **Neuman C. and Ts'o T.** Kerberos: An Authentication Service for Computer Networks // IEEE Communications. 1994. 32(9). P. 33—38.

13. **Neuman C., Yu T., Hartman S., Raeburn K.** The Kerberos Network Authentication System. RFC 4120. July 2005. URL: <http://www.ietf.org/rfc/rfc4120.txt>

## ИНФОРМАЦИЯ

19—21 октября 2011 г. в Москве в ИПУ им. В. А. Трапезникова РАН состоится

### XI Международная конференция "Системы проектирования, технологической подготовки производства и управления этапами жизненного цикла промышленного продукта" (CAD/CAM/PDM—2011)

Тематика конференции:

- ❖ Организация структур технических и программных средств проектирования и управления
- ❖ Средства взаимодействия, структуры данных, международные стандарты
- ❖ Компьютерная графика и CAD/CAM/PDM-системы в учебных процессах
- ❖ Средства виртуальной реальности в промышленных системах
- ❖ Интегрированные производственные системы и управление технологическими процессами
- ❖ PDM-системы, проектирование в машиностроении, строительстве и радиоэлектронике

Подробную информацию см. на сайте конференции: <http://lab18.ipu.rssi.ru>

## Метод генерации тестов для отладки баз знаний экспертных систем

*Рассмотрены основные типы ошибок в базах знаний экспертных систем, построенных на основе продукции и искусственных нейросетей. Приведены методы статической и динамической отладки баз знаний. Предложен метод, позволяющий строить полное множество тестов для баз знаний, показано его использование для тестирования продукционных баз знаний и искусственных нейросетей, основанных на трехслойном персептроне.*

**Ключевые слова:** искусственные нейросети, продукционные системы, ошибки в базах знаний, отладка экспертных систем, ошибка типа "забывание об исключении", извлечение правил, логические схемы

При создании экспертных систем (ЭС) особое место занимает работа со знаниями. Так как база знаний (БЗ) наполняется человеком-экспертом, то в силу своих психологических особенностей он может допускать различного рода ошибки, поэтому этап отладки приобретает особо важное значение. Цель этого этапа — проверка БЗ для выявления различного рода ошибок и их устранения.

Будем использовать понятие отладки ЭС в узком смысле слова [1, 2].

Отладкой ЭС в узком смысле называется процесс выявления, локализации ошибок в БЗ, а также коррекции БЗ, не связанной с выбором нового способа представления знаний.

Для отладки БЗ ЭС используются две основных группы методов:

- методы статического анализа (верификация), осуществляющие проверку БЗ на уровне формального контроля качества, не требующие запуска интерпретатора системы;

- тестирование, заключающееся в прогоне ЭС на заданном множестве тестовых данных и сравнение результатов вывода ЭС с эталонными, определяемыми экспертами.

Для статического анализа обычно используют представление базы знаний в виде модели И/ИЛИ графа. Анализ графовой модели позволяет выявить такие ошибки как избыточность, неполноту, проти-

воречивость, т. е. структурные ошибки. Отметим, что методы статического анализа хорошо формализуемы и поэтому сравнительно легко могут быть реализованы в виде программных модулей. Кроме того, они не нуждаются в оценке решения экспертом. Но в то же время методы статического анализа не обеспечивают достаточную проверку БЗ. Например, БЗ может быть непротиворечивой, но, тем не менее, включать содержательные ошибки.

Отметим, что статически корректная БЗ (т. е. не содержащая структурных ошибок), тем не менее, может делать ошибочные выводы за счет содержательных ошибок в самой базе знаний.

Универсальным средством обнаружения ошибок в БЗ является тестирование. В ходе этапа тестирования осуществляется заключительная оценка выбранного способа представления знаний и ЭС в целом. Как только ЭС оказывается в состоянии обработать несколько примеров от начала до конца, необходимо провести процедуру тестирования, заключающуюся в запуске интерпретатора ЭС на определенном множестве входных данных, называемом тестовым множеством, и в сравнении полученных результатов с эталонными, полученными от эксперта.

Важным является подбор тестового множества данных, на котором будет проверена ЭС. Отметим, что одной из причин неудачной работы ЭС в дальнейшем являются недостаточно показательные тестовые

примеры. В худшем случае они могут оказаться вообще вне проблемной области, на которую рассчитана ЭС, однако чаще всего множество тестовых примеров принадлежит рассматриваемой проблемной области, но является однородным и не позволяет охватить ее всю [3].

Существующий богатый опыт тестирования и оценки обычных программ позволяет частично перенести методы тестирования программ на тестирование БЗ. Классическое определение тестирования как процесса исполнения программы в целях обнаружения ошибок [4] распространяется и на тестирование БЗ: ЭС тестируется не для того, чтобы доказать, что она работает, а наоборот, тестирование начинается с предположения, что в ней есть ошибки, а затем уже обнаруживается максимальное их число. Тестирование заключается в прогоне ЭС на некотором множестве исходных данных, называемом тестовым множеством, в целях обнаружения ошибок. Тестовый прогон считают удачным, если в результате его выполнения обнаруживается ошибка, и неудачным, если получен корректный результат.

Одним из способов тестирования программ, который может быть применен к тестированию БЗ, является стратегия "черного ящика", называемая также тестированием с управлением по данным или с управлением по входу-выходу. ЭС в этом случае рассматривается как "черный ящик". Тесты выбираются из так называемого входного домена (множества, определяемого экспертом). Результаты выполнения сравниваются с эталонными (полученными от эксперта). По результатам сравнения делается вывод о факте наличия ошибок в БЗ. Такая оценка может быть названа качественной оценкой достоверности БЗ. Подобные тесты получили название тестов Тьюринга.

Отметим, что большинство реально действующих ЭС основаны на базах знаний больших объемов и построение вручную тестовых примеров требует больших человеческих и материальных затрат.

К настоящему времени наибольшее распространение для представления знаний в ЭС получили продукции или правила, а также нейросетевой механизм принятия решения. В данной статье описана отладка таких экспертных систем.

Рассмотрим продукционное представление знаний. В англоязычной литературе чаще используется термин "*rule-based knowledge*", т. е. знания, основанные на правилах.

Продукционные системы универсальны, т. е. любая формальная система, оперирующая символами, может быть реализована в виде одной из продукционных систем Поста [5].

Продукционная база знаний (ПБЗ) определяется кортежем  $P$ :

$$P = (F, R, G, C, D), \quad (1)$$

где  $F$  — конечное множество фактов о решаемой проблеме; каждый факт может быть установленным или неустановленным, совокупность установленных фактов задает некоторую ситуацию в предметной области;  $R$  — множество продукций или правил, включающее правила вида

$$r_m: \text{ЕСЛИ } f_i \text{ И } f_j \dots \text{ И } f_n \text{ ТО } f_k, \quad (2)$$

где  $r_m$  — имя правила,  $r_m \in R$ ;  $f_i, f_j \dots f_n$  — факты, стоящие в условии выполнения правила;  $f_k$  — следствие правила,  $f_i, f_j \dots, f_n, f_k \in F$ ;  $G$  — множество целей или терминальных фактов ЭС;  $I$  — интерпретатор правил, реализующий процесс вывода;  $C$  является множеством разрешенных комбинаций фактов. В множество  $C$  не входят, например, комбинации, в которых установлены одновременно какие-либо два факта, взаимно исключающие друг друга. База правил  $R$  и множество целей  $G$  образуют базу знаний.

Введем понятие тестового примера  $T$  для БЗ экспертной системы.

Под тестовым примером  $T$  для базы знаний понимают набор фактов с приписанными им истинными значениями. Если, основываясь на этом наборе фактов, интерпретатор выдает заключение  $g \in G$ , где  $g$  — результат выполнения тестового примера  $T$  над БЗ, то  $g$  — результат выполнения тестового примера  $T$  над БЗ.

Под тестом понимается пара  $(T, g')$ , где  $g'$  — определение пользователем правильного результата выполнения тестового примера  $T$ .

Существует тривиальный метод для проверки статической корректности для генерации тестовых данных, удовлетворяющих критерию покрытия операторов — достаточно сгенерировать все возможные комбинации фактов. Число сгенерированных таким образом тестов растет экспоненциально от числа возможных фактов. Несмотря на то, что время генерации одного теста здесь постоянно, суммарное время генерации тестов растет также экспоненциально от числа фактов.

Методы так называемого "белого ящика" основаны на проверке содержимого БЗ, т. е. конкретных продукций. Для обычных программ тестирование по принципу "белого ящика" характеризуется степенью, в какой из них тесты выполняют или покрывают логику (исходный текст) программы.

В классическом тестировании программного обеспечения (ПО) часто используется такой критерий, как критерий покрытия операторов (КПО), требующий выполнения каждого оператора программы хотя бы один раз. Обычно этот критерий является довольно слабым, так как это требование является необходимым, но недостаточным условием для приемлемого тестирования. Более сильным критерием является критерий покрытия решений, который требует, чтобы

каждое решение имело результатом TRUE или FALSE и при этом каждый оператор выполнялся по крайней мере один раз.

Такой критерий контроля качества ПО как критерий комбинаторного покрытия всех решений и путей в программе в большинстве случаев очень трудоемок как для обычных программ, так и для ЭС.

Отметим, что эти требования к критериям тестирования справедливы для БЗ, не содержащих вероятностей в продукциях.

Для БЗ, содержащих вероятностные или нечеткие продукции, решение обычно принимается на основе аккумуляции всей используемой информации. Например, правило может быть выполнено на заданных тестовых данных, но его вклад в процесс принятия решения, возможно, будет минимальным. В этом случае КПО является явно недостаточным критерием и целесообразно рассмотреть достижение каждой цели в БЗ. Это также необходимый критерий эффективного тестирования, но не достаточный.

Для получения эталонных значений  $G'$  могут быть использованы дополнительные эксперты или те же, кто участвовал в формировании базы знаний. Такой подход также позволяет обнаруживать ошибки в правилах. Дело в том, что эксперт обладает несколькими дублирующими системами рассуждений. Допуская ошибки при формировании ПБЗ большой размерности, он, как правило, дает правильные решения для конкретной постановки задачи.

Проанализируем возможности КПО с точки зрения полноты обнаружения ошибок БЗ. Если логическое следствие в правиле действительно выполняется всегда, независимо от установки других фактов, то данная продукция является правильной, т.е. не содержит ошибок. Наиболее грубая ошибка в продукции имеет место, если факт  $f_i$  никогда не бывает истинным в случае, когда установлены факты  $f_1, f_2, \dots, f_k$ .

Легко убедиться что тесты, полученные по КПО, обнаруживают именно ошибки такого рода. Если (1) может выполняться или не выполняться в зависимости от состояния некоторых других фактов, то обнаружение подобных ошибок не гарантируется тестами КПО.

Наиболее грубая ошибка в правиле (2) имеет место, если факт  $f_i$  в данной предметной области никогда не может быть установлен в присутствии фактов  $f_1, \dots, f_k$ . Именно такие ошибки обнаруживают тесты, построенные по критерию покрытия операторов. Действительно, данный критерий предусматривает однократную активизацию каждого правила. Если факт — следствие некоторой продукции на самом деле не должен устанавливаться при выполнении условия продукции, то ЭС сообщит об истинности какой-либо цели, которая не должна удовлетворяться в данной ситуации.

Более реален случай, когда факт — следствие  $f_i$  иногда должен устанавливаться при установленных фактах  $f_1, \dots, f_k$ , входящих в условие продукции, а иногда не должен. Это зависит от состояния других фактов предметной области (ПО). Очевидно, что тестирование по критерию покрытия операторов не гарантирует обнаружения ошибок этого наиболее общего класса.

В общем случае, вследствие ошибок эксперта множество фактов ПБЗ  $F$  может содержать не все факты, существенные для рассматриваемой ПО. В результате некоторые правила из  $P$  могут оказаться неверными при установке некоторых фактов, не входящих в  $F$ . Анализ ПБЗ не позволяет сформировать тестовый пример, обеспечивающий обнаружение подобных ошибок. В дальнейшем предполагается, что множество  $F$  содержит все факты, установка которых может влиять на справедливость правил  $R$ . Соответственно, класс обнаруживаемых ошибок ограничивается этим допущением.

Рассмотрим подробнее наиболее сложную для выявления ошибку в БЗ, связанную с так называемым "забыванием об исключении" в предметной области.

При формировании правил ПБЗ вследствие психологических особенностей эксперт практически всегда безошибочно выделяет для данного факта  $f_i$  те факты, которые способствуют его установке. Однако факты, "мешающие" установке рассматриваемого факта  $f_i$ , зачастую упускаются экспертом. В результате возникают ошибки, имеющие характер "забывания" о границах применимости правил (классический пример летучей мыши при классификации птиц и животных).

Применение перенесенных из тестирования ПО критериев контроля качества для тестирования ПО с достаточной степенью точности выявляет противоречия вида:

ЕСЛИ  $f_1$  И  $f_2 \dots$  И  $f_k$  ТО НИКОГДА  $f_n$ . (3)

Однако применение подобных методов не в состоянии обнаружить скрытые противоречия вида:

ЕСЛИ  $f_1$  И  $f_2 \dots$  И  $f_k$  ТО НЕ ВСЕГДА  $f_n$ . (4)

Формальная модель ошибок типа, который можно назвать также "забывание об исключении" [1, 2, 6], состоит в следующем: правило выполняется всегда, за исключением того случая, когда в ПБЗ установлен набор фактов  $S$ :

$$\{f_1, f_2, \dots, f_k\} = S \in C. \quad (5)$$

Данный класс ошибок является наиболее общим и покрывает ошибки типа "забывание о нескольких исключениях".

## Метод построения тестов для обнаружения ошибок типа "забывание об исключении"

Если БЗ может быть представлена И/ИЛИ-графом  $\Gamma$ , то после проведения статического анализа ее логика может быть задана более наглядно соответствующей  $\Gamma$  связной логической сетью (ЛС). Переход к логико-сетевому представлению ПБЗ позволяет использовать для построения тестов методы технической диагностики цифровых устройств.

Распространенной моделью неисправностей в диагностике цифровых устройств является так называемая константная неисправность (см., например, [7]). В схеме, содержащей константную неисправность, выход одного из вентилях всегда находится в состоянии "0" или всегда находится в состоянии "1" независимо от состояний его входов. Ошибку типа (5) можно рассматривать как неисправность "константный 0" логической сети, которая проявляется только при каком-то одном наборе значений сигналов в ЛС.

В терминах ЛС тестом, обнаруживающим неисправность типа (5), является набор установленных и неустановленных фактов, которые обеспечивают активизацию правила  $r_i$  и активизацию пути в ЛС от линии  $r_i$  до одной из выходных линий, которым соответствуют цели ПБЗ [7]. При этом должны быть установлены факты  $f_1, \dots, f_k$ . Если активизация  $r_i$  и путей невозможна при установленных фактах  $f_1, \dots, f_k$ , то тест не существует.

Любой непротиворечивый набор фактов  $S$  может быть установлен в  $P$  путем задания по крайней мере одного набора входных фактов  $S \in C$ . При этом множество  $C$  также должно быть задано относительно входных фактов ПБЗ. Такое представление  $C$  можно получить, выполняя обратный логический вывод для всех запрещенных комбинаций, содержащих внутренние факты  $P$ .

Сказанное является основанием для построения тестов ПБЗ в виде наборов ее входных фактов.

В силу того, что множество фактов  $f_1, \dots, f_k$ , при одновременной установке которых правило  $r_i$  перестает выполняться, неизвестно, для выполнения  $r_i$  необходимо перебрать все варианты активизации правила  $r_i$  и все возможные способы транспортировки соответствующего  $r_i$  сигнала логической сети до одного из ее выходов. Все факты, состояние которых несущественно для активизации  $r_i$  и путей от  $r_i$  до выходов сети, должны при этом устанавливаться. Заметим, что правило  $r_i$ , в общем случае, может быть неверным при нескольких комбинациях фактов. Данный подход к построению множества тестов для  $r_i$  обеспечит обнаружение всех таких ситуаций.

Возможно также наличие в ПБЗ кратных ошибок, когда неверными могут быть несколько правил. Полный тест ПБЗ  $P$  может быть получен путем объединения тестовых множеств всех правил из множества  $R$ .

Обозначим через  $T$  полученное таким образом множество тестов для  $P$ .

Несложно убедиться, что  $T$  обеспечит обнаружение любых кратных ошибок  $P$  типа (5). Действительно, так как логическая сеть, соответствующая  $\Gamma$ , не содержит инверсий, кратные ошибки не могут маскировать друг друга.

*Определение 1.* Продукции, в условиях которых присутствуют только входные факты логической сети ПБЗ, будем называть листовыми.

Для построения тестов ПБЗ можно использовать методы активизации путей, развитые в технической диагностике. При этом необходимо строить все возможные тесты для каждой листовой продукции. В теории логических сетей принято представлять множества соседних двоичных наборов в виде так называемых кубов. Куб представляет собой набор символов, в котором кроме "0" и "1" могут присутствовать символы "X" и "D". Каждый символ куба обозначает состояние одной соответствующей линии ЛС. Если какой-нибудь линии приписан символ "X", то ее состоянием может быть как "0", так и "1". В каждом случае единичное состояние линии соответствует установленному факту, нулевое — неустановленному. Символ "X" означает, что соответствующий факт может быть как установленным, так и неустановленным. Символ "D" приписывается линии, если она принимает состояние "1" в исправной ЛС и состояние "0" в неисправной ЛС. Кубы, осуществляющие активизацию рассматриваемого правила и активизацию путей, пересекаются с множеством допустимых кубов входных фактов  $C$ . В кубах-результатах все символы "X" заменяются символом "1". Наиболее целесообразно использовать принятый в технической диагностике алгоритм PODEM [8]. Данный алгоритм предусматривает возврат по входам ЛС после каждого акта транспортировки сигнала через очередной элемент. В случае генерации тестов для ПБЗ этот подход особенно эффективен. Полученный в результате возврата куб входных фактов пересекается с кубом множества  $C$ . Если это пересечение пусто, то дальнейшая активизация рассматриваемого пути не проводится.

При построении тестов ЛС используется так называемая операция D-пересечения кубов, выполняемая по следующим правилам:

$$1. X \cap b_i = b_i; a_i \cap X = a_i,$$

где  $a_i$  и  $b_i$  — значения  $i$ -х координат кубов.

$$2. \text{Если } a_i \neq X \text{ и } b_i \neq X, \text{ то}$$

$$a_i, \text{ если } b_i = a_i,$$

$$a_i \cap b_i = 0, \text{ в противном случае.}$$

Результатом пересечения двух кубов  $A$  и  $B$  будет являться куб

$$A \cap B = (a_1 \cap b_1, a_2 \cap b_2, \dots, a_n \cap b_n).$$

Если хотя бы для одной координаты результат пересечения не определен, то  $A \cap B = \emptyset$ .

Для построения тестов множество  $S$  допустимых наборов входных фактов удобно представить в виде наборов кубов. Заметим, что эксперту психологически проще сформировать кубы, описывающие запрещенные, а не допустимые комбинации фактов. После приведения этих ограничений к виду кубов входных фактов в результате обратного логического вывода, множество  $S$  разрешенных наборов может быть получено с помощью операции вычитания кубов.

Пусть, например,  $C2 \cap C1$ , где  $C1 = 1X0X10X$ ,  $C2 = 100X101$ , (куб  $C2$  покрывается кубом  $C1$ , т. е. множество двоичных наборов, которые представляет куб  $C2$ , является подмножеством множества наборов куба  $C1$ ). Тогда результатом вычитания  $C1/C2$  являются кубы  $C3 = 110X10X$  и  $C4 = 1X0X100$ .

В результате вычитания получается столько кубов, сколько раз символ "X" в уменьшаемом кубе приходится против "0" или "1" в кубе вычитаемом. Куб-результат получается заменой в уменьшаемом рассматриваемого символа "X" на инверсное значение соответствующего символа из куба вычитаемого.

### Алгоритм построения полного множества тестов ПБЗ

Для построения полного множества тестовых наборов для обнаружения неисправностей типа (4) ПБЗ предлагается следующий алгоритм.

1. Установить искомое множество тестов  $T$  пустым:  $T = \emptyset$ .

2. Выбрать очередную продукцию.

3. Установить факты, необходимые для ее активизации. Если их установка противоречит множеству  $S$ , то сообщить об ошибке и перейти к шагу 7.

4. Воспользоваться алгоритмом PODEM для получения множества кубов  $T1$ , которое является тестовым для выбранной продукции. Кубы  $T1$  пересекаются с кубами множества  $S$  по мере их формирования, причем  $T2 = T1 \cap S$ .

5. В кубах  $T2$  заменить все символы "X" на "1" и в полученном списке исключить повторяющиеся кубы. Результат есть множество кубов  $T3$ .

6. Выполнить объединение множеств двоичных наборов  $T$  и  $T3$ . Результат назначить новым множеством  $T$ .

7. Если есть еще продукции, то перейти к шагу 2.

Так как PODEM обеспечивает построение теста всегда, когда он существует, то теми же свойствами обладает и предложенный алгоритм.

При тестировании для получения эталонных ответов используются эксперты. Для того, чтобы тестирование было реально применено на практике, нужно, чтобы число тестов не превышало несколько десятков. Несложно убедиться в том, что тестовое множество, которое строится разработанным алгоритмом, является минимальным.

Тесты строятся для листовых продукций. Ни один из наборов тестового множества не может быть удален, так как в этом случае тестовое множество по крайней мере для одной из листовых продукций окажется неполным.

### Построение тестового множества для трехслойного персептрона

Наряду с продукционной формой представления знаний, сегодня все большую популярность приобретают экспертные системы, принимающие решения на основе искусственных нейросетей (ИНС). Наибольшее число систем принятия решения, использующих нейросетевой механизм, представляет собой трехслойный персептрон.

Трехслойный персептрон имеет входной слой, один скрытый (промежуточный) слой и выходной слой. Для каждой предсказываемой переменной имеется один нейрон во входном слое. В случае категоризованных переменных  $N - 1$  нейрон представляют  $N$  категорий переменных.

Рассмотрим трехслойный персептрон  $PR$ , обученный на множестве  $L$ . Входной вектор  $X$  соответствует множеству  $In$  значений входных нейронов. Персептрон  $PR$  осуществляет отображение множества входных векторов  $X$  в множество выходных векторов  $Y$ :  $PR = X \rightarrow Y$ .

Так как обучающее множество формируется экспертами, то не существует гарантии того, что обученная нейросеть не содержит ошибок. Введем понятие ошибки нейросети. Ошибка нейросети имеет место при наличии такой комбинации значений входных нейронов  $\{In_1, In_2, \dots, In_m \in In\}$ , при установке которой отображение  $PR$  становится неверным.

Формальная модель ошибки типа "забывание об исключении" заключается в том, что отображение  $X \rightarrow Y$  корректно всегда, кроме случая установки такой комбинации значений входных нейронов  $\{In_1, In_2, \dots, In_m \in In\}$ , при которой отображение  $PR$  становится неверным.

Для использования предложенного метода генерации тестового набора ИНС должна быть преобразована к продукционной базе знаний, затем к виду И/ИЛИ-графа, и затем логической сети. Преобразование ИНС к логической сети требует, во-первых, дискретизации степени взаимовлияния нейронов, во-вторых, представления значений входных, про-

межуточных и выходных признаков дискретными значениями 0 или 1; в-третьих, прореживания структуры сети путем определения и исключения наименее значимых нейронов и связей.

Алгоритмы преобразования нейросети к системе продукций можно подразделить на поисковые алгоритмы и алгоритмы, основанные на прямой интерпретации весовых коэффициентов связей. Главным недостатком алгоритмов первого типа является их высокая вычислительная сложность, которая экспоненциально растет при увеличении числа входных и выходных признаков [9]. Поэтому для извлечения правил предлагается использовать алгоритм GLARE, основанный на прямой интерпретации весовых коэффициентов связей между нейронами входного и промежуточного слоя и связей между нейронами промежуточного и выходного слоя [9]. Важным является тот факт, что при использовании GLARE для извлечения правил отсутствует необходимость модификации метода обучения или структуры самой нейронной сети, а также набор правил, сгенерированный с помощью алгоритма GLARE, имеет точность прогноза не меньшую, чем точность прогноза самой нейронной сети [10, 11].

### Алгоритм построения полного множества тестов для нейросетевой экспертной системы, основанной на трехслойном персептроне

Введем следующие обозначения:  $N_1$  — число нейронов на первом слое,  $St_{i,j}$  — активационное значение  $j$ -го нейрона на  $i$ -м слое,  $w_{i,j}^k$  — весовой коэффициент связи нейрона  $j$  на слое  $i$  с нейроном  $k$  ( $i - 1$ )-го слоя.

Для нейронов первого слоя примем дополнительные обозначения:  $m$  — номер входного признака,  $n$  — номер значения входного признака  $m$ , которое кодирует данный нейрон. Таким образом,  $St_{1,i}^{m,n}$  означает  $i$ -й нейрон первого слоя, соответствующий значению  $n$  атрибута  $m$ .

Приведем девятишаговый алгоритм построения полного множества тестов для экспертной системы на основе трехслойного персептрона:

**Шаг 1.** Выбор значения параметра  $NWIN$ , удовлетворяющего условию:

$$1 < NWIN < N_1.$$

**Шаг 2.** Построение упорядоченной последовательности скрытых нейронов выходного нейрона  $St_{3,1}$ .

2.1. Формирование для нейрона  $St_{2,1}$  последовательности  $RWH_1$ , состоящей из нейронов входного

слоя, упорядоченных по убыванию абсолютного значения весового коэффициента связи с нейроном  $St_{2,1}$ :

$$RWH_1 = \left\{ St_{1,i}, St_{1,j}, \dots, |Abs(w_{2,1}^i > w_{2,1}^{i+1}) \right\},$$

где  $Abs(w_{2,1}^i)$  — абсолютное значение весового коэффициента связи нейрона 1 на слое 2 с нейроном  $i$  слоя ( $i - 1$ ).

2.2. Формирование для нейрона  $St_{2,1}$  упорядоченной последовательности  $RRWH_1$  путем усечения длины последовательности  $RWH_1$ , полученной на предыдущем шаге, и включения первых  $NWIN$  ее элементов:

$$RRWH_1 = \left\{ St_{1,i}, St_{1,j}, \dots, St_{1,NWIN} | Abs(w_{2,1}^i > w_{2,1}^{i+1}) \right\}.$$

2.3. Получение множества последовательностей  $RRWH$  путем повторения шагов 2.1—2.2 для всех нейронов промежуточного слоя:

$$RRWH = \{RRWH_1, RRWH_2, \dots, RRWH_n\}.$$

2.4. Выделение из обучающего множества  $L$  подмножества  $L^1$ , каждый входной вектор которого активирует нейрон выходного слоя  $St_{3,1}$ .

2.5. Расчет индекса значимости  $II_{2,1}$  для нейрона  $St_{2,1}^j$  путем подачи на вход сети элементов обучающего подмножества  $L^1$  мощности  $n$  и расчета среднего активационного значения  $AAL_{2,1}$  нейрона  $St_{2,1}$ :

$$AAL_{2,1} = \frac{1}{n} \sum_{i=1}^n St_{2,1}^i,$$

$$II_{2,1} = |AAL_{2,1} w_{3,1}^1|.$$

2.6. Повторение шага 2.5 для каждого нейрона скрытого слоя и получение множества индексов значимостей  $II$ :

$$II = \{II_{2,1}, II_{2,2}, \dots, II_{2,p}\}.$$

2.7. Построение упорядоченной (по возрастанию соответствующих индексов значимости) последовательности скрытых нейронов  $RWHO_1$  для выходного нейрона  $St_{3,1}$ , причем знаки элементов последовательности берутся равными знакам соответствующих

весовых коэффициентов связей данного нейрона с нейронами скрытого слоя:

$$RWHO_1 = \left\{ \begin{array}{l} \text{sign}(w_{3,1}^i) | St_{2,i}, \text{sign}(w_{3,1}^j) | St_{2,j} | \dots \\ \Pi_{2,i} < \Pi_{2,j} \end{array} \right\},$$

где  $\text{sign}(x) = \begin{cases} 1, & x \geq 0; \\ -1, & x < 0. \end{cases}$

**Шаг 3.** Построение матрицы  $ATTR$ , строками которой являются последовательности  $RRWIH$ , полученные на шагах 2.1—2.3 и отсортированные по порядку следования соответствующих элементов последовательности  $RWHO_1$ , полученной на шаге 2.7. При этом знаки элементов матрицы  $ATTR$  инвертируются в тех строках, которые соответствуют отрицательным элементам последовательности  $RWHO_1$ :

$$ATTR = \left[ \begin{array}{l} \text{sign}(w_{3,1}^l) RRWIN_l \\ \text{sign}(w_{3,1}^m) RRWIN_m \\ \text{sign}(w_{3,1}^n) RRWIN_n \\ \dots \end{array} \right], \Pi_{2,l} < \Pi_{2,m} < \Pi_{2,n} \dots$$

**Шаг 4.** Построение матрицы  $RATTR$  путем формирования и заполнения нулевой матрицы размером  $i \times j$ , где  $i$  — число входных атрибутов,  $j$  — наибольшее число возможных значений атрибутов.

4.1. Если элементом  $ATTR_{1,1}$  является нейрон  $St_{1,h}^g$ , то  $RATTR_{h,g}$  принимает следующие значения:

$$RATTR_{h,g} = \text{sign}(St_{1,h}^g).$$

4.2. Пара  $\langle h, g \rangle$  заносится в список  $U$  использованных координат.

4.3. Повторение шагов 4.1 и 4.2 для  $ATTR_{1,2}, \dots, ATTR_{1,NWIH}$ , причем, если текущие значения пары  $\langle h, g \rangle$  присутствуют в списке использованных координат  $U$ , то  $RATTR_{h,g}$  новое значение не присваивается.

4.4. Повторение шагов 4.2—4.4 для всех строк  $RATTR$ .

**Шаг 5.** Построение классифицирующего правила  $RULE_1$  на основе матрицы  $RATTR$ . Обозначая через  $St_{1,j}^{m,n}$  входной нейрон  $j$ , соответствующий значению  $n$  атрибута  $m$  и учитывая, что при  $RATTR_{i,j} = 1$  атрибут  $i$  принимает  $j$ -е значение (в противном случае

$RATTR_{i,j} = -1$ ), классифицирующее правило имеет вид:

$$\begin{aligned} & \text{if} \left( St_{1,i}^{m_1, n_1} = 1 \right) \text{ or } \left( St_{1,j}^{m_1, n_2} = 1 \right) \dots \text{ and} \\ & \left( St_{1,k}^{m_2, n_1} = 1 \right) \text{ or } \left( St_{1,h}^{m_2, n_2} = 1 \right) \dots \text{ and} \\ & \dots \text{ then class}_1. \end{aligned}$$

**Шаг 6.** Получение набора классифицирующих правил путем повторения шагов 2—5 для всех остальных выходных нейронов  $St_{3,2}, St_{3,3}, \dots, St_{3,p}$ :

$$\begin{aligned} & \text{if} \left( St_{1,i}^{m_1, n_1} = 1 \right) \text{ or } \left( St_{1,j}^{m_1, n_2} = 1 \right) \dots \text{ and} \\ & \left( St_{1,k}^{m_2, n_1} = 1 \right) \text{ or } \left( St_{1,h}^{m_2, n_2} = 1 \right) \dots \text{ and,} \\ & \dots \text{ then class}_1 \end{aligned}$$

$$\begin{aligned} & \text{if} \left( St_{1,i}^{m_1, n_1} = 1 \right) \text{ or } \left( St_{1,j}^{m_1, n_2} = 1 \right) \dots \text{ and} \\ & \left( St_{1,k}^{m_2, n_1} = 1 \right) \text{ or } \left( St_{1,h}^{m_2, n_2} = 1 \right) \dots \text{ and,} \\ & \dots \text{ then class}_2 \end{aligned}$$

.....

$$\begin{aligned} & \text{if} \left( St_{1,i}^{m_1, n_1} = 1 \right) \text{ or } \left( St_{1,j}^{m_1, n_2} = 1 \right) \dots \text{ and} \\ & \left( St_{1,k}^{m_2, n_1} = 1 \right) \text{ or } \left( St_{1,h}^{m_2, n_2} = 1 \right) \dots \text{ and,} \\ & \dots \text{ then class}_p \end{aligned}$$

**Шаг 7.**

7.1. Формирование при помощи экспертов валидационного множества  $V$ .

7.2. Оценка на множестве  $V$  точности классификации полученного набора правил. В случае удовлетворительной точности набора правил закончить работу алгоритма. В противном случае вернуться к шагу 1, выбрав большее значение для параметра  $NWIH$ .

**Шаг 8.**

8.1. Построение для множества правил, полученных на шаге 7, соответствующей логической схемы  $S$ .

8.2. Формирование экспертным методом множества запрещенных кубов  $C1$ .

8.3. Получение множества разрешенных кубов  $C$  путем вычитания из куба всевозможных комбинаций  $R$  значений запрещенных кубов  $C_1, \dots, C_h$ .

$$C = R \setminus C_1 \setminus C_2 \setminus C_3 \dots \setminus C_h.$$

#### Шаг 9.

9.1. Установление множества тестов  $T$  пустым:  $T = \emptyset$ .

9.2. Формирование множества  $T$  путем применения алгоритма построения полного множества тестов.

Разработанный метод генерации тестов для отладки баз знаний ЭС был успешно применен для тестирования продукционной ЭС KORDEX, осуществляющей прогноз развития нестабильной стенокардии [2], а также для ЭС *Glaukoma Complaint*, основанной на механизме трехслойного перцептрона, реализующей прогноз комплаентности для офтальмологических больных [10, 11]. Следует отметить, что при извлечении правил из ИНС ЭС *Glaukoma Complaint* число *NWIN* было принято равным 40 при числе нейронов входного слоя 136.

#### Список литературы

1. Долинина О. Н. Обнаружение ошибок типа "забывание об исключении" в продукционных базах знаний экспертных систем. Саратов: СГТУ, 1997. Деп. в ВИНТИ, № 678-В97.

2. Долинина О. Н. Информационные технологии в управлении современной организацией. Саратов: Изд. Сарат. гос. техн. ун-та, 2006. 160 с.

3. Искусственный интеллект: в 3 кн. Кн. 2. Модели и методы: Справочник / под ред. Э. В. Попова. М.: Радио и связь, 1990. 304 с.

4. Майерс Г. Искусство тестирования программ / Пер. с англ. под ред. Б. А. Позина. М.: Финансы и статистика, 1982. 176 с.

5. Post E. L. Formal Reductions of the General Combinatorial Decision Problem // American Journal of Mathematics V. 65. 1943. P. 197—215.

6. Поспелов И. Г., Поспелова Л. Я. Динамическое описание систем продукции и проверка непротиворечивости продукционных экспертных систем. // Известия АН СССР. Техническая кибернетика. 1987. № 1. С. 184—192.

7. Основы технической диагностики. В 2 кн. Кн. 1. Модели объектов, методы и алгоритмы диагноза / Под ред. Пархоменко П. П. М.: Энергия, 1976. 464 с.

8. Goel P. An implicit enumeration algorithm to generate tests for combinational logic circuits // IEEE Trans. Computers. 1981. Vol. C-30. No 3. P. 215—222.

9. Gupta A., Park S., Lam S. Generalized Analytic Rule Extraction for Feedforward Neural Networks // IEEE Transactions on Knowledge and Data Engineering. 1999. Vol. 11, No 6. P. 985—992.

10. Долинина О. Н., Кузьмин А. К. Применение методов технической диагностики для отладки баз знаний нейросетевых экспертных систем // Информационные технологии. 2009. № 2. С. 34—38.

11. Долинина О. Н., Кузьмин А. К. Метод генерации тестов для отладки нейросетевых экспертных систем // Вестник ТГТУ, 2010. Т. 16, № 3. С. 519—528.

2—3 декабря 2011 г. в Москве пройдет

## ЮБИЛЕЙНАЯ Десятая Международная конференция в области обеспечения качества ПО "Software Quality Assurance Days"



На конференции планируется обсуждение ключевых вопросов по следующим тематикам:

- функциональное тестирование;
- интеграционное тестирование;
- тестирование производительности;
- автоматизация тестирования и инструментальные средства;
- конфигурационное тестирование;
- тестирование удобства использования (usability);
- тестирование защищенности (security);
- статические методы обеспечения качества;
- внедрение процессов тестирования на предприятии;
- управление процессами обеспечения качества ПО;
- менеджмент команд тестировщиков и инженеров качества ПО;
- аутсорсинг тестирования;
- тестирование системных приложений (не Web), а также тестирования игр и мобильных приложений;
- мотивация проектной команды и сертификация специалистов в области обеспечения качества ПО.

Подробную информацию см. на сайте конференции <http://it-conf.ru/ru/content/379.htm>

---

---

## CONTENTS

**Kukhareno B. G.** Open Closed Principle in Program Engineering and Design Patterns. . . . . 2

As shown, Open Closed Principle in object oriented programming appears on program system microarchitecture level. Design patterns represent class hierarchies, which form a general solution of program system design problem. Techniques are under study, which are able to detect design pattern modified versions in program systems, differing from standard representations by additional inheritance level. An efficiency of program system component graph and design pattern graph similarity scoring method in use is demonstrated by detecting canonic and demonstrative examples of patterns written in Java code.

**Keywords:** object oriented programming, open-closed principle, microarchitecture, design patterns, pattern detection, graph algorithms

**Kolodenkova A. E.** The Program Project Viability Estimation in the Conditions of Fuzzy Initial Data . . 10

The problem of program projects viability estimation in the conditions of uncertainty generated by Not-factors is being considered. Multi-fuzzy and fuzzy-interval approaches to the decision of the given problem on the basis of comparative multicriterial analysis of possible design alternatives in the conditions of fuzzy initial data is offered.

**Keywords:** the program project, project viability estimation, Not-factors and uncertainty multi-fuzzy and fuzzy-interval approaches to program project viability estimation

**Shokurov A. V.** Irregularly Detailed Raster Image Coding Software Design with Respect to Adaptive and Interactive Image Analysis . . . . . 17

A research of visual data encoding methods, designs of mathematical models of such data, software and management tools, as well as the analysis of the corresponding test results are given in the paper. The author develops an efficient, adaptive and interactive irregularly detailed raster image analysis method that is based on the propositions of the traditional SPIHT algorithm and gives an analysis of the software implementation properties on test images. This efficient method is developed with respect to the encoded data stream sizes, the image fragments extraction speed, the

amount of data relevant to the fragments being extracted and the amount of random access memory used by the software.

**Keywords:** irregularly detailed raster image, SPIHT-based image coding method, interactive image analysis, low-bandwidth channel, multiscale and progressive image fragment transmission modes, size adaptive image fragment retrieval mode

**Yazov Y. K., Soloviev S. V., Stupnikov A. V.** Some Aspects of Maintenance Joint Functioning of Applied Decision Support System with Systems of Electronic document Circulation . . . . . 29

Some approaches to synthesis of applied systems support of decision-making with introduced systems of electronic document circulation are offered. Variants of realization of these ways of interface reveal. Two ways of grouping these systems in the uniform automated complex are considered.

**Keywords:** system of electronic document circulation, applied system, ways of integration

**Inyukhin A. V.** Grid Authentication for the Web . . 35

An approach for Web authentication using Grid single sign-on capabilities is presented in the paper. This approach is based on HTTP GSSAPI authentication implemented in popular browsers, and it requires no additional software for clients. A functional prototype of a server-side component is provided for evaluation.

**Keywords:** distributed systems, Grid-systems, authentication, functional prototype

**Dolinina O. N.** Method of Test Generation for Debugging of Knowledge Bases for Artificial Intelligence Systems . . . . . 40

There are considered main types of errors in knowledge bases of expert systems, based on rules and on artificial neural networks. There are described methods of static and dynamic debugging of knowledge bases. Method of building of full set of tests for knowledge bases is suggested. There is described its using for testing of rule-based systems and for artificial intelligence systems based on 3-level perceptron.

**Keywords:** artificial neural networks, rule-based systems, errors in knowledge bases, debugging of expert systems, type of error "forgetting about exception", knowledge extraction, logic schemes

---

---

ООО "Издательство "Новые технологии". 107076, Москва, Стромьинский пер., 4

Дизайнер *Т.Н. Погорелова*. Технический редактор *Е.М. Патрушева*. Корректор *Т.В. Пчелкина*

Сдано в набор 19.06.2011 г. Подписано в печать 26.07.2011 г. Формат 60×88 1/8. Бумага офсетная. Печать офсетная.  
Усл. печ. л. 5,88. Уч.-изд. л. 6,88. Цена свободная.

---

Отпечатано в ООО "Белый ветер", 115407, г. Москва, Нагатинская наб., д. 54, пом. 4