САНКТ-ПЕТЕРБУРГСКИЙ ЦЕНТР РАЗРАБОТОК ЕМС

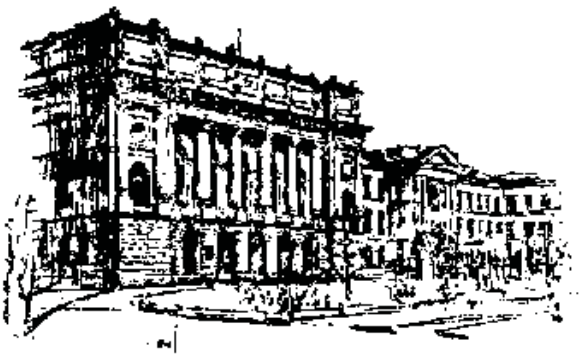
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

САНКТ-ПЕТЕРБУРГСКИЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых

28 апреля 2015 года



Санкт-Петербург
Издательство Политехнического университета
2015

УДК 004

Современные технологии в теории и практике программирования: Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада. СПб.: Изд-во Политехн. ун-та, 2015. 105 с.

В сборнике публикуются материалы докладов студентов ряда вузов, представленные на конкурс-конференцию, проводимую Санкт-Петербургским политехническим университетом Петра Великого и организованную институтом информационных технологий и управления при поддержке Санкт-Петербургского Центра Разработок EMC. Доклады отражают современный уровень подготовленности студентов политехнического университета и других вузов – участников конференции в области применения современных средств и технологий разработки программного обеспечения.

Представляет интерес для специалистов в области информационных технологий, методов разработки программных проектов различного назначения, систем и средств автоматизации инженерного проектирования, для учащихся и работников системы высшего образования.

Печатается по решению редакционно-издательского совета Санкт-Петербургского политехнического университета Петра Великого.

Приветствие Санкт-Петербургского Центра Разработок EMC

Уважаемые участники конференции
«Современные технологии в теории и практике программирования»!

Индустрия информационных технологий во всем мире и в России стремительно развивается. Ежедневное совершенствование технологий бросает серьезный вызов высшему образованию. Уровень технических вузов в нашей стране по-прежнему очень высок, однако для многих экспертов очевидно, что не всегда высшая школа успевает за взрывным ростом в ИТ-индустрии и справляется с задачей подготовки профильных специалистов в требуемом количестве. Важная роль успешного преодоления этого дисбаланса как раз отведена построению эффективных партнерских взаимоотношений между наукой и бизнесом, и в частности проведению и поддержке совместных образовательных инициатив между высшей школой и представителями ИТ компаний.

Корпорация EMC в третий раз поддерживает конференцию «Современные IT технологии в теории и практике программирования», проводя свою специальную секцию под названием «Подходы к разработке ПО на основе технологий EMC», а в этом году еще и став основным партнером конференции. Данная конференция зарекомендовала себя как отличная площадка для демонстрации молодыми учеными результатов их исследований.

В 2014-м году на базе кафедры «Информационные и управляющие системы» Санкт-Петербургского политехнического университета был открыт научно-образовательный центр «Политехник-EMC». За первый год работы центра нами было запущено четыре совместных образовательных проекта, а также организовано преподавание курса «Управление информацией и хранение данных», входящего в портфолио программы академического партнерства корпорации EMC, с последующей профессиональной сертификацией студентов. Помимо этого, в 2015-м году EMC уже в четвертый раз наградит успешных первокурсников института информационных технологий и управления именными грантами корпорации. Стоит отметить, что, работая со студентами, реализовывая проекты, мы не привязываем ребят к EMC. После завершения проекта у студентов остается понимание технологий, новые знания и навыки, которые они могут применить как в EMC, так и в любой другой компании или университете.

Целью как данной конференции в целом, так и в частности секции EMC является в первую очередь поддержка молодых ученых, их научных исследований в области информатики и информационных технологий; информирование молодых ученых о передовых технологических трендах в индустрии, о современных программных продуктах и решениях, представленных крупнейшими мировыми ИТ-корпорациями; знакомство молодых специалистов со стратегией развития ИТ компаний, обзор рынка в целом; выявление

талантливых и перспективных студентов и аспирантов для потенциального сотрудничества в будущем. Рынок стремительно развивается, поэтому нам хочется помогать вузам и студентам овладевать всеми необходимыми навыками и технологиями, в частности получая с помощью подобных научно-образовательных конференций фундаментальные знания и практические навыки.

Желаю вам успешной работы на конференции, профессионального роста и творческих побед!

С уважением,

Вячеслав Михайлович Нестеров

д.ф-м.н.

Генеральный директор

Санкт-Петербургского Центра Разработок ЕМС

Информационное сообщение о проведении учебно-практической конференции студентов, аспирантов и молодых ученых «Современные технологии в теории и практике программирования»

1. В целях содействия подготовке к будущей работе в профессиональных программистских коллективах, обеспечивающих высокое качество программного продукта, в целях поддержки изучения современных информационных технологий и инструментальных средств в соответствии с мировыми стандартами и действующими международными сертификационными требованиями, а также для выявления талантливых молодых специалистов в области разработки и использования программных систем, инженерных проектов и моделей, компания EMC совместно с Санкт-Петербургским Государственным Политехническим Университетом объявляют учебно-практическую конференцию **«Современные технологии в теории и практике программирования».**
2. В рамках мероприятия планируется провести:
   - Конкурс-конференцию теоретических и практических работ - претендентов на именные дипломы компании EMC.
3. Сроки проведения мероприятия: **28 апреля 2015 года**.
4. Место проведения

   Институт информационных технологий и управления, 2 учебный корпус, а.359
   Санкт–Петербургский Государственный Политехнический Университет
   ул. Политехническая д.29
   195251, г. Санкт–Петербург

   Регистрация 28 апреля 9.30

   Предлагаемая тематика конкурса:

   - Программная инженерия: приложения, продукты и системы
   - Программная инженерия: инструментальные средства и технологии проектирования и разработки
   - Программная инженерия: методы и алгоритмы теории программирования
   - Подходы к разработке ПО на основе технологий EMC.

Данный конкурс рассчитан на студентов 1-6 курсов, аспирантов и молодых ученых, обучающихся или работающих по направлениям информатики, вычислительной техники, и по направлениям использования информационных технологий, владеющих основами современных промышленных технологий и инструментарием разработки программных продуктов и проектов. Предполагается, что участник должен проявить свои знания и умения не столько в области программирования различных математических головоломок, сколько в области разработки и использования программных продуктов и систем в условиях, максимально приближенных к реальным процессам проектирования и разработки современных систем различной степени сложности.

Темы работ могут быть взяты в соответствии с тематикой секции конференции.

На конкурс принимаются работы, оформленные в соответствии с заданными требованиями и представленные в организационный комитет конференции не позднее **20 марта 2015 года**. Требования к представлению и оформлению проектов представлены в документах «Порядок представления тезисов на конкурс-

конференцию» [0] и «Требования к проведению докладов и секционных выступлений на конкурсе-конференции» [II-V].

5. Количество работ конкурсантов не ограничено.
6. Конкурс принятых к участию работ проводится в один этап и протекает в виде представления презентаций и секционных докладов. Приглашение отобранным участникам будут рассылаться в период с 10 по 15 апреля 2015г. Решение о награждении участников конкурса принимается конкурсной комиссией. Требования к проведению докладов представлены в документе «Порядок проведения презентаций и выступлений на конкурсе-конференции» [IV].
7. Объявления об условиях проведения конкурса и вся дополнительная информация представлены на сайте ИИТУ СПбПУ в разделе Конференции. Приглашения на участие в конкурсе-конференции будут разосланы в ведущие университеты Северо-Запада.

ЖЕЛАЕМ УСПЕХОВ И ИНТЕРЕСНОЙ РАБОТЫ!

Научные руководители конференции:

проф. Черноруцкий Игорь Георгиевич,

проф. Котляров Всеволод Павлович

секретарь Эламик Татьяна Николаевна, тел. 552-76-66, часы приема: понедельник и среда с 10 до 17 часов.

Адрес: Политехническая ул., 29, 2 уч.корпус, к.358
E-mail: vpk@spbstu.ru

Порядок представления проектов на конкурс-конференцию.

Для представления разработанных проектов в Конкурсный центр устанавливаются следующие правила:

- Желающие принять участие в конкурсе – конференции на этапе предварительного сбора материалов на почтовый адрес vpk@spbstu.ru присылают анкету участника и тезисы разрабатываемого **проекта, приложения или модели** в виде секционного доклада. Тезисы докладов, представленные на конкурс - конференцию, будут напечатаны в Сборнике материалов конкурса – конференции.
- На заочном этапе – первом туре - конкурсная комиссия на основе тезисов отбирает работы для непосредственного участия в конкурсе, после чего высылает приглашение по электронной почте (при наличии) или по телефону.
- Очный этап проводится в **виде секционных докладов для теоретических или прикладных поисковых работ.**

  Конкурсная комиссия оценивает представленные доклады по представлению руководителей секций и отбирают наиболее значимые из них для награждения.
- Все участники награждаются специальными дипломами ЕМС.

I. Требования к содержанию конкурсных работ.

Предполагается, что конференция будет проходить в рамках четырех секций:

С1.Программная инженерия: приложения, продукты и системы

Задача конкурса в разделе С1: продемонстрировать в тезисах и презентации знания и умения в создании программного продукта на основе доклада, содержащего обоснование предлагаемого решения и анализа преимуществ по сравнению с существующими.

(Например, программные проекты, представляющие завершенный продукт или незавершенный, но исполняемый в соответствии с требованиями к промышленному продукту)

Рекомендуется особо отмечать передовые технологии, поддержанные в программных продуктах и оценку эффективности их применения.

С2.Программная инженерия: инструментальные средства и технологии проектирования и разработки

Задача конкурса в разделе С2: продемонстрировать знания и умения в создании и/или применении перспективных методов и технологий разработки программного обеспечения на основе доклада, содержащего обоснование предлагаемого решения и анализа преимуществ по сравнению с существующими. Наряду с технологиями разработки программных продуктов на секции рассматриваются программные средства систем управления жизненным циклом информации.

(Например, автоматизация разработки спецификации, доказательства корректности спецификаций, их использование при генерации кода, автоматизация тестирования, средства обеспечения качества программного продукта, средства управления разработкой программного продукта, средства идентификации и информационной безопасности, управляющие и встроенные применения, беспроводная телекоммуникация, моделирование контроллеров и других аппаратно-программных решений, средства управления качеством реализации программного

продукта, системы управления электронным документооборотом, безопасностью хранения данных, виртуализация вычислительных ресурсов.)

Рекомендуется особо отмечать передовые технологии, поддержанные в программных продуктах и оценку эффективности их применения.

С3. Программная инженерия: методы и алгоритмы теории программирования

Задача конкурса в разделе С3: продемонстрировать знания и умения в разработке и применении формальных методов при создании и/или улучшении либо оптимизации характеристик программного обеспечения на основе доклада, содержащего обоснование предлагаемого решения и анализа преимуществ по сравнению с существующими.

(Например, алгоритмы и методы для проверки корректности программного продукта, исполнимые спецификации, формальные методы для применения в программировании и т.п.)

Оценка применимости предлагаемых подходов на практике и оценка эффективности применения.

С4.Подходы к разработке программного обеспечения на основе технологий ЕМС

Задача конкурса в разделе С4: продемонстрировать знания и умения в разработке ПО с применением новых технологий ЕМС. на основе доклада, содержащего обоснование предлагаемого решения и анализа преимуществ.

II. Требования к оформлению студенческих работ на конкурс-конференцию.

Для участия в конкурсе необходимо представить в электронном варианте:

1. **Анкету участника,** составленную в произвольной форме (Ф.И.О., название вуза, факультета, №группы, домашний адрес, телефон, адрес электронной почты, предполагаемая секция, предполагаемое название работы или доклада; сведения о руководителе: Ф.И.О., место работы, адрес электронной почты (если возможно).
2. **Тезисы доклада и презентацию,** содержащие название работы, постановку задачи, краткое описание проекта, оценку характеристик демонстрационной версии. Объем тезисов 1 страница напечатанного текста.

III. Требования к оформлению тезисов доклада:

электронный вариант текста набирать в редакторе **Microsoft Word версии 2007**со следующими параметрами настройки:

- шрифт — Times New Roman;
- стиль шрифта — нормальный (обычный);
- размер кегля шрифта — 12;
- межстрочный интервал — 1;
- параметры страницы: формат А4, поля верхнее — 20 мм, нижнее — 20 мм, левое — 20 мм, правое — 20 мм;
- формулы набирать, пользуясь Microsoft Equation (настройка символов в редакторе формул пропорциональна основному тексту; по возможности, использовать запись формулы в строчку);
- если для понимания сути работы необходим рисунок, он выполняется в виде единой картинки (формата типа bmp) в пределах поля для текста (**не допускаются рисунки, составленные из отдельных элементов**);

Текст тезисов в текстовом поле располагается следующим образом:

- на первой строчке (выровнять влево — левом верхнем углу): УДК (вместе с цифрами печатать прописными буквами);
- на следующей строчке (выровнять вправо): инициалы, фамилия студента строчными буквами, в скобках курс и название кафедры и ВУЗа); инициалы, фамилия, ученая степень, должность руководителя (использовать принятые сокращения); если не умещается в строчку, то можно в две строчки;
- через пробел (выровнять по центру): НАЗВАНИЕ ТЕЗИСОВ ДОКЛАДА (прописными буквами);
- через пробел (с красной строки, равной 1 см): текст тезисов.

ВНИМАНИЕ: Тезисы доклада должны быть написаны ясным языком, без орфографических ошибок.

Образец оформления работы:

УДК 621.319

Андреев Д.В., Литвинов А.П. (4 курс, каф. ИУС, СПбГПУ),
Гаригин А.В., Никитин М.А. (асп., каф. ИУС, СПбГПУ),
Котляров В.П., к.т.н., проф.

УДАЛЁННОЕ УПРАВЛЕНИЕ PC С ИСПОЛЬЗОВАНИЕМ МОБИЛЬНЫХ УСТРОЙСТВ POCKETPC И SMARTPHONE

Целью работы является исследование возможностей новой среды разработки Microsoft Visual Studio 2005 и проектирование системы универсального удалённого управления ПК с использованием мобильных устройств на базе Windows CE/Windows Smartphone.
Эта система должна обеспечивать взаимодействие между устройствами путём обмена командами, результатами их выполнения и информационных сообщений.

…

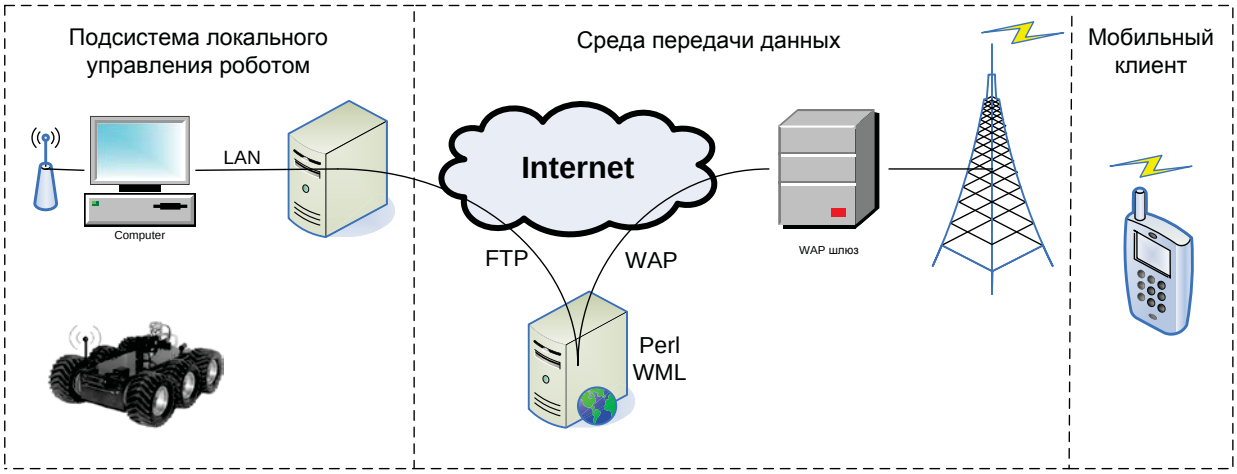


Рис.1 Система дистанционного управления роботом с помощью мобильного телефона.

…

Особенности реализации системы управления:

Сетевое взаимодействие реализовано на базе технологии Wi-Fi, как наиболее удобной в программировании. Однако далеко не все мобильные устройства оснащены необходимым оборудованием, поэтому система был разработан отдельный модуль для работы с Bluetooth. В перспективе предполагается полностью абстрагироваться от технологии связи и предоставить возможность пользователю выбирать её самостоятельно.

В качестве формата передачи команд и данных используется XML. Для работы с этой формой представления данных используется система чтения XML DOM, доступная в Compact Framework. Важно обеспечить относительно простую схему XML для передаваемых данных, чтобы её можно было разбирать на мобильных устройствах в режиме он-лайн без видимых задержек.

IV. Порядок проведения презентаций и выступлений.

В процессе сбора материалов на конкурс-конференцию конкурсная комиссия отбирает лучшие работы для представления презентаций и выступлений, о чем участник конкурса ставится в известность.

1. Порядок проведения выступления на конкурсе-конференции.
   1.1. В ходе конференции используется проектор Multi Media.
   1.2. Для доклада при помощи Microsoft PowerPoint готовится презентация.
   1.3. Презентация должна содержать не менее 10 слайдов. Обязательными являются следующие слайды:
   - Титульный, на котором должна быть представлена следующая информация:
     - Название проекта.
     - Фамилия и имя докладчика;
     - Учебное заведение, которое он представляет;
     - Фамилия И.О. научного руководителя.
   - Область применения проекта.
   - Какие задачи решаются проектом.
   - 2 слайда на описание концепции проекта.
   - 2 слайда на особенности реализации проекта.
   - Представление полученных результатов проекта.
   - Перечень использованных для реализации продуктов Microsoft (указать, какие технологии и для чего использовались в проекте).
   - Заключение.

   1.4. Длительность доклада не должна превышать 12 минут.

V. Положение о системе награждения победителей конкурса – конференции.

Победителей конкурса определяет конкурсное жюри, в состав которого включены ведущие преподаватели вузов. Жюри оценивает представленные материалы и отбирает лучшие из них для участия в работе конференции. Работы будут премироваться следующим образом:

1. Тезисы докладов по всем отобранным комиссией проектам будут опубликованы в сборнике тезисов докладов конкурса-конференции.
2. Победители получат именные дипломы.

Тезисы докладов конкурса-конференции

Секция «Программная инженерия: приложения, продукты и системы»

УДК 004.4+004.7

А. А. Андреев (4 курс, каф. ИМО, ПетрГУ),
А. С. Колосов (ст. преподаватель, каф. ИМО, ПетрГУ),
Ю. А. Богоявленский, к.т.н., доцент

МОДЕЛЬ ИКТ-ИНФРАСТРУКТУРЫ ПОСТАВЩИКА СЕТЕВЫХ УСЛУГ ДЛЯ АВТОМАТИЗИРОВАННОГО ПОСТРОЕНИЯ ГРАФА СЕТИ

Экспериментальная платформа (ЭП) Nest [1], разрабатываемая на кафедре ИМО ПетрГУ, предназначена для исследования моделей и методов управления ИКТ-инфраструктурами поставщиков сетевых услуг (далее Сеть) и эффективности их использования в бизнес-процессах. Основным инструментом ЭП является граф Сети, представляющий как физические так и логические взаимосвязи между ее узлами. Его построением занимается подсистема Nestopo [2,3]. Такой граф может быть использован для документирования Сети, моделирования нагрузок на ее сегменты, оптимизации ее структуры и т. п.

В основе построения графа Сети лежит процесс обнаружения сетевых устройств и взаимосвязей между ними, заключающийся в обработке большого количества разнородных источников данных напрямую или косвенно описывающих структуру Сети. Существующие работы в данном направлении ограничиваются обработкой только одного источника данных о связях канального уровня [4], накладывая на них информацию об организации VLAN [6] или уровня IP [7].

Целью данной работы является разработка математической модели для описания физической структуры Сети с учетом ее логической организации на уровне виртуальных сетей (VLAN стандарта IEEE 802.1Q) и уровне межсетевого взаимодействия IP, а также разработка на основе этой модели унифицированного и независимого от источников данных алгоритма построения графа Сети.

Представим Сеть в виде неориентированного графа, вершины которого соответствуют устройствам и их сетевым интерфейсам, а ребра отражают физические связи между интерфейсами и отношения принадлежности между интерфейсами и устройствами. Вершины графа разметим идентификаторами VLAN, которым принадлежат соответствующие элементы Сети. Данная модель позволяет описать основные способы организации сетевых устройств — широковещательные домены (в том числе VLAN) и IP-подсети.

Разработанный на основе данной модели новый комплексный алгоритм построения графа Сети, использует множество источников прямых и косвенных данных о физической и логической структуре Сети (кэши CDP, LLDP, ARP, деревья STP, RSTP, MSTP, таблицы

коммутации и маршрутизации, конфигурацию VLAN) и состоит из четырех основных этапов: 1) сбор данных из MIB устройств; 2) наполнение множеств вершин графа и соединение устройств с их интерфейсами, назначение меток VLAN; 3) построение связей между интерфейсами; 4) выделение независимых друг от друга широковещательных доменов и IP-подсетей.

Новая программная реализация подсистемы на языке Java предоставляет возможность выбора источников данных для построения, последовательного и параллельного исполнения каждого из этапов алгоритма, кеширования данных с опрошенных устройств, возможность реализации сторонних алгоритмов. Подсистема насчитывает 53 класса и интерфейсов общим объемом 4651 строк кода (включая комментарии). Тестирование проводилось в вычислительной сети Петрозаводского государственного университета, а также в виртуальных сетевых окружениях GNS3 [9] и показало повышение точности построения графов Сетей за счет использования множества доступных источников данных.

ЛИТЕРАТУРА:

1. Богоявленский Ю. А. Прототип экспериментальной платформы Nest для исследования моделей и методов управления ИКТ-инфраструктурами локальных поставщиков услуг Интернет // Программная инженерия. — 2013. — No 2. — С. 11–20.

2. Колосов А. С., Богоявленский Ю. А. Параллельный алгоритм построения графа ИКТ-инфраструктуры интернет-провайдера // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. — 2013. — № 3. — С. 105–110.

3. Андреев А. А., Колосов А. С., Богоявленский Ю. А., Автоматизация построения графа канального уровня ИКТ-инфраструктуры локального поставщика услуг интернета // Ученые записки Петрозаводского государственного университета. Сер. «Естественные и технические науки». — 2015. — № 2 (147). — С. 97–102.

4. Gobjuka H., Breitbart Y.J. Ethernet Topology Discovery for Networks With Incomplete Information // Networking, IEEE/ACM Transactions on. — 2010. — Vol. 18, no. 4. — P. 1220–1233.

5. Ethernet topology discovery for virtual local area networks with incomplete information / Li Zichao, Hu Ziwei, Zhang Geng, Ma Yan // Network Infrastructure and Digital Content (IC-NIDC), 2014 4th IEEE International Conference on. — 2014. — P. 252–256.

6. Bejerano Y. Taking the Skeletons Out of the Closets: A Simple and Efficient Topology Discovery Scheme for Large Ethernet LANs // Networking, IEEE/ACM Transactions on. — 2009. — Oct. — Vol. 17, no. 5. — P. 1385–1398.

7. GNS3 [Электронный ресурс]. URL: http://www.gns3.com (дата обращения: 21.03.2015).

УДК 004.4'24

Н. Д. Борисов (5 курс, ИУС, СПбПУ)
И.В. Никифоров (ассистент, к.т.н., ИУС, СПбПУ)

ИСПОЛЬЗОВАНИЕ БИБЛИОТЕКИ APACHE OLINGO ДЛЯ ДОСТУПА К ДАННЫМ ПО ПРОТОКОЛУ ODATA

В настоящее время все более актуальной становиться задача удаленного доступа и манипулирования данными в рамках разработки web-приложений. В рамках данной работы будет рассмотрен протокол OData[1], а так же библиотека Apache Olingo[2] для доступа к OData сервисам с целью получения запрашиваемых данных.

Для доступа и манипулирования данными в современных веб-приложениях целесообразно использовать протокол OData. OData — это веб-протокол на основе ресурсов для запроса и обновления данных. OData определяет операции с ресурсами, используя HTTP-команды (PUT, POST, UPDATE и DELETE), и идентифицирует эти ресурсы по стандартному синтаксису URI[3]. Данные передаются поверх http[4] с применением стандартов XML или JSON. Любой клиент, способный работать с протоколом OData, может выполнять операции с любым источником данных. OData позволяет конструировать URI-идентификаторы для именования набора сущностей, осуществлять фильтрацию содержащихся в этом наборе сущностей, а также прослеживать отношения со связанными сущностями и коллекциями сущностей.

Библиотека Apache Olingo является реализацией спецификации OData для платформы на языке Java. Процедура получения данных от OData-сервиса с помощью библиотеки Olingo состоит из этапов перечисленных ниже.

1. **Создание объект класса ODataClient**
   Экземпляр данного объекта используется для управления механизмами отправки запроса к OData-сервису и получение ответа соответствующего запросу.
2. **Получение метаданных**
   Получение метаданных осуществляется путем создания объекта ODataServiceDocumentRequest с указанием адреса OData сервиса и вызова метода execute(), с последующим присвоением результата его выполнения объекту ODataServiceDocument. Метаданные необходимы для того, чтобы определить какие-именно данные будет предоставлять OData-сервис.
3. **Получение данных**
   Получение данных состоит из 3 этапов:
   - Создание URI запроса
   - Выполнения запроса
   - Просмотр полученных данных

Библиотека Olingo предоставляет класс URIBuilder, который позволяет создавать OData-запросы для получения запрашиваемых данных. Данный класс поддерживает все особенности протокола OData. После формирования URI-запроса и последующей его отправкой, необходимо обработать полученный ответ от OData-сервиса с помощью класса ODataRetrieveResponse. В результате обработки ответа от сервиса образуется множество объектов ODataEntity, содержащее запрашиваемые данные.

Описанный выше метод получения данных, применен при разработке программного компонента, реализующего поиск и отображение информации из системы хранения пользовательских контактов. На рис.1 показана схема работы программного компонента.

Рис. 1 Схема работы программного компонента

Система хранения контактов представлена OData-сервисом. Клиентом является web-браузер, отправляющим запросы сервлету[5]. Существует возможность отправлять запросы к OData-сервису из клиента, минуя java-сервлет, путем формирования особого http-запроса. В данном случае технология сервлетов необходима для создания программного компонента, реализующего запросы и обработку данных полученных от OData-сервиса, и последующего встраивания программного компонента в уже существующее программное окружение, с целью преобразования информации в более удобный тип для обработки клиентом.

В результате выполнения данной работы был описан метод применения библиотеки Apache Olingo для получения данных по протоколу OData. По описанному методу был реализован программный компонент для отправки запроса и последующей обработки данных от внешней системы хранения пользовательских контактов протоколу OData.

ЛИТЕРАТУРА:

[1]OData протокол [Электронный ресурс]. Режим доступа: // http://www.odata.org/

[2] Apache Olingo [Электронный ресурс]. Режим доступа: // http://olingo.apache.org/

[3] URI [Электронный ресурс]. Режим доступа: // https://tools.ietf.org/html/rfc3986

[4] HTTP-протокол [Электронный ресурс]. Режим доступа: // http://www.w3.org/Protocols/rfc2616/rfc2616.html

[5] Сервлет [Электронный ресурс Режим доступа: // http://www.oracle.com/technetwork/java/index-jsp-135475.html

УДК 004.4

Н.М. Гаврилова (3 курс, каф. ИУС, СПбПУ),
С.А. Молодяков, д.т.н., проф.

ПЕРВЫЕ ШАГИ РАЗРАБОТКИ СИСТЕМЫ ПРОЕКТИРОВАНИЯ АНАЛОГОВЫХ IP-БЛОКОВ

Проектирование аналоговых устройств, входящих в системы на кристалле (SoC, SIC), требует временных затрат в несколько раз больше, чем проектирование цифровых устройств, хотя последние занимают на кристалле примерно во столько же раз большую площадь, чем аналоговые электронные устройства. Поэтому разработчики интеллектуальных (Intellectual Property - IP) блоков заинтересованы в использовании автоматизированных систем

проектирования, которые бы не только сокращали время проектирования, но и позволяли получать аналоговые IP-блоки высокого качества. Номенклатура IP-блоков, которые требуется разрабатывать, очень широка, и может включать аудио и видео кодеки, LVDS приемники и передатчики, фотоприемники, фильтры и др.

Известна методология проектирования и набор методов решения задачи синтеза IP-блоков [1]. Методология построена на базе спиралевидной модели проектирования, которая содержит четыре аспекта: технологический, функциональный, структурный и конструкторский, а также семь уровней, или этапов. Первый весьма непростой этап – этап формирования требований S_{TET}, которым должен удовлетворять проектируемый объект. Затем идут этапы: синтез принципов построения S_{Pr}; аппроксимация S_A желаемого облика объекта проектирования или его характеристик; синтез способов построения S_{Res}; синтез структуры S_{App}, параметров S_{Par} её элементов и допусков на них S_{Tol}. Композиция отображений этапов проектирования структур имеет вид

$$\Pi = S_{Tol} \circ S_{Par} \circ S_{App} \circ S_{Res} \circ S_A \circ S_{Pr} \circ S_{TET} \qquad (1)$$

Композиция начинается с формализации требований, а заканчивается проектированием допусков на параметры элементов.

Цель работы состоит в разработке программного обеспечения системы автоматизированного проектирования аналоговых электронных IP-блоков, которая функционировала бы в соответствии со спиралевидной моделью.

Выполнен первый этап работы, связанный с разработкой системы проектирования RC фильтров. В качестве прототипа использована система проектирования фильтров Filter Wiz [2]. Разработано программное обеспечение (первая версия) системы проектирования IP-блоков (названо Design IP). Система Design IP написана на языке программирования C# в среде Visual Studio. Построение сценария разработки позволяет провести разработчика по этапам проектирования IP-блока. В основной форме задаются проектируемый объект и его параметры, отображается схема проектирования/продвижения. Если в базах данных имеются соответствующие элементы, то на выходе показывается итоговая схема объекта. Если нет выходного объекта, то продвигаемся по этапам, начиная с этапа, на котором отсутствует элемент в базе. Система Design IP синтезирует фильтр высокого порядка из имеющихся фильтров первого и второго порядка. На этапе аппроксимация S_A используются соотношения, которые используются при расчете фильтра Баттерворта.

Дальнейшее развитие проекта связано с разработкой программного обеспечения для синтеза других устройств, использования методов синтеза на основе графов, формирования выходного файла для дальнейшего использования (в системе Cadence).

ЛИТЕРАТУРА:

1. Лыпарь Ю.И. Системное проектирование. Функциональный и структурный аспекты. // Кибернетика и информатика. Сб. стат. Изд-во: Политехника.- 2006.- С.217-238.

2. Filter Wiz Pro – программа для расчета активных фильтров http://www.schematica.com/active_filters/fwpro.html

УДК 004.4, 004.8

И.В. Галов (асп., каф. ИМО, ПетрГУ)
Д.Ж. Корзун, к.ф.-м.н., доцент

РАЗРАБОТКА БРОКЕРА СЕМАНТИЧЕСКОЙ ИНФОРМАЦИИ ДЛЯ ЛОКАЛИЗОВАННЫХ ВЫЧИСЛИТЕЛЬНЫХ СРЕД ИНТЕРНЕТА ВЕЩЕЙ

В работе рассматривается разработка брокера семантической информации (от англ. Semantic Information Broker, SIB) [1] в локализованных средах Интернета вещей (от англ. Internet Of Things, IoT). В таких вычислительных средах технологии IoT обеспечивают взаимосвязь между устройствами, находящимися в пространственно-ограниченной области, а также предоставляют доступ к внешним ресурсам сети Интернет. Такие возможности позволяют создавать так называемые «интеллектуальные окружения», накапливающие знания о самой среде и адаптирующиеся к ее участникам. Частным случаем таких окружений являются интеллектуальные пространства (ИП, от англ. Smart Spaces). Брокер SIB обеспечивает взаимодействие между программными агентами ИП на основе коммуникационных моделей «классная доска» и «публикация/подписка». Взаимодействие между агентами — косвенное, через общее разделяемое информационное содержимое, представленное в виде хранилища RDF-троек в брокере SIB. Существует несколько реализаций брокера SIB [2]: Smart-M3 SIB, RIBS, OSGi SIB, RedSIB. Они создавались как исследовательские прототипы для экспериментальной апробации подхода ИП для условий IoT-сред. В то же время, для локализованных IoT-сред существенными практическими требованиями выступают простота, расширяемость, надежность и переносимость брокера SIB, значимость которых, в частности, рассматривается в [3].

Предлагается архитектура новой версии брокера SIB для локализованных IoT-сред, поддерживающая перечисленные характеристики. Особенностью новой архитектуры является модульный подход, позволяющий выборочно подключать и отключать модули. Тем самым, предоставляется возможность настроить функциональность SIB под конкретную IoT-среду. Как отдельные модули – плагины с динамическим подключением реализуются протоколы доступа к информационному содержимому ИП, такие как SSAP и KSP [2]. Возможно расширение набора операций для информационного содержимого.

Брокер SIB реализуется на языке программирования C/C++ с использованием программного каркаса Qt. В результате достигается высокая степень переносимости, позволяющая запускать брокер SIB на различных платформах: как на стандартных компьютерах (напр., с ОС Linux или Windows), так и на специализированных устройствах (напр., маршрутизаторы с ОС OpenWrt). Для организации информационного содержимого ИП используется открытая библиотека Redland, предоставляющая общий интерфейс к различным реализациям хранилищ RDF-троек. Так, становится возможной конфигурация брокера для хранилищ, ориентированных на работу в оперативной памяти, файловой системе или семантической базе данных Virtuoso.

Разработка выполняется при финансовой поддержке Минобрнауки России по заданию № 2014/154 на выполнение государственных работ в сфере научной деятельности в рамках базовой части государственного задания (НИР № 1481). Работа поддержана РФФИ (проект № 14-07-00252).

ЛИТЕРАТУРА:

1. Honkola J. et al. Smart-M3 information sharing platform // Proceedings of the IEEE Symposium on Computers and Communications. 2010. P. 1041–1046.

2. Kiljander J. et al. Semantic interoperability architecture for pervasive computing and Internet of Things // IEEE Access. 2014. Vol. 2. P. 856–873.
3. Галов И.В., Корзун Д.Ж. Обеспечение устойчивости к сбоям Smart-M3 приложения на уровне программной инфраструктуры // Труды СПИИРАН. 2014. Вып. 37. С. 188–207.

УДК 004.4'24

Н. С. Голубев (5 курс, ИУС, СПбПУ), И. В. Никифоров (ассистент, к.т.н., ИУС, СПбПУ)
П. К. Полубенцев (3 курс, ИУС, СПбПУ), В. П. Котляров (проф, к.т.н., ИУС, СПбПУ)

ИССЛЕДОВАНИЕ ВОЗМОЖНОСТЕЙ ПРИМЕНЕНИЯ APACHE HADOOP В УЧЕБНОМ ПРОЦЕССЕ НА КАФЕДРЕ ИУС

На сегодняшний день одним из самых перспективных и бурно развивающихся направлений в информационной индустрии является область "Больших данных" (BigData) [1]. Эта область напрямую связана с тем, что современные цифровые средства: различного сорта датчики, видеокамеры, фотокамеры, социальные сети, алгоритмы моделирования поведений элементарных частиц в физике, информация поступающая от клиентов какого-либо сервиса, каждую секунду генерируют гигабайты и даже терабайты данных, которые нужно уметь как хранить, так и обрабатывать.

Крупные компании, сталкивающиеся с огромными объемами данных, уже наработали существенное множество различных фреймворков, методов и подходов по обработке терабайтов информации. К таким компаниям относятся ведущие флагманы IT-индустрии: EMC, IBM, Oracle, Microsoft и HP.

Стремительный рост интереса крупных компаний влечет за собой потребность в специалистах по "Большим Данным", что заставляет высшие учебные заведения реагировать на запрос и готовить выпускников в данной области. Для этого на базе Санкт-Петербургского государственного политехнического университета (СПбПУ) на кафедре Информационные и управляющие системы (ИУС) создают новые учебные курсы, направленные на формирование у обучающихся общих знаний предметной области, формирование кругозора и получение умений и навыков применения технологий и фреймворков, используемых в области "Больших данных".

В области "Больших данных" выделяют три главные характеристики, которые еще называют «принцип трех V» в соответствии английским значениям: объем (volume), скорость (velocity), многообразие (variety). Наиболее широко используемой технологией в области "Больших данных" является технология компании Apache Hadoop с моделью распределенных вычислений MapReduce [2]. По данным Yahoo 50% всех больших данных будет обрабатываться с помощью Apache Hadoop [3]. Проект Hadoop состоит из нескольких частей. Выделим несколько основных: HDFS, YARN и MapReduce.

Одной из особенностей Hadoop является то, что он использует свою собственную распределенную файловую систему – Hadoop Distributed File System (HDFS) [4]. Это файловая система, предназначенная для хранения файлов больших размеров, которые поблочно распределены между частями большого кластера. YARN – модуль, отвечающий за управление ресурсами кластеров и планирование заданий. YARN может быть рассмотрен как кластерная операционная система. Он управляет интерфейсом между аппаратными ресурсами кластера и

широким классом приложений, использующих его мощности для выполнения вычислительной обработки. Hadoop MapReduce – программный каркас для программирования распределённых вычислений в рамках парадигмы MapReduce. Разработчику приложения для Hadoop MapReduce необходимо реализовать базовый обработчик, который на каждом вычислительном узле кластера обеспечит преобразование исходных пар «ключ – значение» в промежуточный набор пар.

У Hadoop есть три режима работы [5]: автономный режим работы (Standalone Operation Mode), псевдо-распределенный режим (Pseudo Distributed Mode), полностью распределенный режим (Fully Distributed Mode). Автономный режим используется для того, чтобы на локальной машине использовать Hadoop как кластер, состоящий из одного узла. Псевдо-распределенный режим похож на автономный режим, за исключением того, что каждый демон Hadoop запускается на отдельном процессе. Полностью распределенный режим используется, когда нужно установить систему в параллельном режиме на два и более компьютеров. При этом стоит отметить, что использование полностью распределенного режима требует значительных аппаратно-вычислительных мощностей.

В работе были исследованы возможности автономного режима Apache Hadoop для проведения лабораторных работ по курсу "Наука о данных и аналитика больших объемов данных". Были проанализированы существующие примеры для обучения основам технологии, выбраны наиболее пригодные из них для изучения студентами и список был расширен новыми типовыми задачами.

Каждая лабораторная работа состоит из двух частей. Первая часть общая для всех слушателей курса - это установка и развертывание Apache Hadoop на локальном компьютере, без которой невозможно приступить к выполнению второй части.

Вторая часть - уникальная для каждого из студентов и требует реализации смысловой задачи на основе парадигмы MapReduce. Ниже представлен список уникальных вариантов лабораторных работ для студентов.

1. Разработать проект, который вычисляет наиболее часто встречающееся слово в тексте и предоставляет информацию о числе вхождений каждого слова в заданном тексте.

2. Создать программу, которая вычисляет число вхождений последовательности слов в тексте. Число элементов в последовательности слов задается входным параметром [6].

3. Найти максимальную и минимальную температуру в городе на основе входных данных почасовой записи показаний термостата.

4. Реализовать сортировку данных методом слияния.

5. Разработать алгоритм архивации данных с использованием Hadoop.

6. Вычисление маршрута автотранспорта по камерам слежения. Расчет маршрута и его восстановление в заданный промежуток времени и даты.

7. Реализовать поиск по документам с определенным критерием.

8. Репликация данных на разных узлах файловой системы. Использование самых быстрых узлов для получения информации от HDFS.

9. Поиск не подходящих под паттерны макетов для повышения безопасности приложений.

Реализацию кода программы в рамках лабораторных работ следует делать в IDE Eclipse, т.к. для этой интегрированной среды разработки существует удачный модуль (plug-in), расширяющий возможности IDE по разработке приложений MapReduce [7].

Несмотря на то, что инструментальные средства Apache Hadoop требуют значительных вычислительных мощностей для работы с полностью распределенном режимом, программно-

аппаратное обеспечение кафедры ИУС позволяет развернуть технологию на многоядерном кластере. Для наиболее заинтересованных студентов предлагается возможность по использованию кластера ИУС с развернутыми средствами Apache Hadoop для выполнения лабораторных работ в полностью распределенном режиме.

В работе были изучены возможности по использованию Apache Hadoop в учебном процесс на кафедре ИУС СПбПУ в рамках курса "Наука о данных и аналитика больших объемов данных". Были предложены варианты лабораторных работ для студентов. В перспективе будущей учебно-методической работы планируется изучение языка R для проведения анализа и визуализации данных и подготовка соответствующей лекции и лабораторных работ.

ЛИТЕРАТУРА:

1. А.А.Барсегян, Анализ данных и процессов. – БХВ-Петербург, 2009. – с. 512.
2. White T., Hadoop: The Definitive Guide. - 2-е изд. - O'Reilly, 2011. - 625 с.
3. The Big Cost Of Big Data [Электронный ресурс]. Режим доступа: http://www.forbes.com/sites/ciocentral/2012/04/16/the-big-cost-of-big-data/
4. Apache Hadoop [Электронный ресурс]. Режим доступа: http://hadoop.apache.org/
5. Lam C., Hadoop in Action. - Manning Publications, 2010. - 336 с.
6. Bigram Counter Example [Электронный ресурс]. Режим доступа: http://www.dia.uniroma3.it/~torlone/bigdata/E1-Hadoop.pdf
7. Hadoop Eclipse Plugin [Электронный ресурс]. Режим доступа: https://github.com/winghc/hadoop2x-eclipse-plugin

УДК 621.396

Н.В. Ермаков (4 курс, каф. ИУС, СПбПУ),
Б.М. Медведев, доцент

ВСТРАИВАЕМЫЕ СРЕДСТВА ОТЛАДКИ АППАРАТНО-ПРОГРАММНЫХ СИСТЕМ ОБРАБОТКИ СИГНАЛОВ НА БАЗЕ ПРОЦЕССОРНОГО МОДУЛЯ STM32F4DISCOVERY

ARM - самая популярная платформа для встраиваемых систем. Модули STM32F4DISCOVERY обладают следующими характеристиками: 1 Мбайт флеш памяти, 192 Кбайт ОЗУ, а также многоканальный АЦП.

Целью работы является создание программных средств, позволяющих обработать аналоговый сигнал и отобразить его на экране компьютера в удобном виде.

Программное обеспечение состоит из встраиваемого ПО для модуля STM32F4DISCOVERY и ПО визуализации для компьютера. Для встраиваемого ПО требуется записывать отсчеты сигнала с АЦП в оперативную память, передавать накопленные данные по USB, обрабатывать команды, переданные с компьютера. Для ПО визуализации требуется принимать данные с платы, строить графики изменения сигнала во времени и спектра сигнала, передавать на плату управляющие команды.

Возможно 3 способа отображение сигнала:

1. Сначала записываем в память, затем отображаем
2. Отображаем постоянно по 255 отсчетов с заданной задержкой или по требованию

3. Отображаем 255 отсчетов после пересечения сигналом некоторого заданного порога с заданной задержкой или по требованию

Программа для микроконтроллера написана на языке C в среде разработке Embedded Workbench for ARM (IAR EWARM). Программа для компьютера написана на языке C++ в среде разработке Qt с использование библиотек qwt (для графика) и QtSerialPort (для USB).

Особенности реализации системы и технические характеристики:

1. АЦП:

   Диапазон напряжения от 0 до 3.3 В, разрядность 12 бит, максимальная частота дискретизации 36МГц. Сигнал подается на ножку PC2 на плате, частота дискретизации настраивается с помощью таймера, для которого надо указать делитель и период. Частота АЦП будет равна 168 МГц/(делитель*период). Данные с АЦП записываются в оперативную память с помощью контроллера прямого доступа к памяти, а затем передаются на компьютер по USB.
2. USB:

   Используется режим эмуляции COM-порта. Достигнутая скорость передачи составляет примерно 5 Мбит/с. В этом случае режиме постоянной передачи данных можно работать с частотой АЦП до 300 КГц. По USB от компьютера передаются команды для задания частоты дискретизации АЦП, режима работы, порога для третьего режима отображения и команда для получения данных.
3. Приложение для компьютера:

   Вначале на плату передаются все необходимые настройки. Для второго и третьего режимов отображения можно задать задержку между отображениями. Полученные данные отображаются в виде графика. Можно отобразить спектр сигнала, полученный с помощью быстрого преобразования Фурье.

УДК 681.3, 004.031.4

М. М. Заславский (аспирант первого года, каф. ИПМ, НИУ ИТМО),
Т.А. Берленко (2 курс магистратуры, каф МО ЭВМ, СПбГЭТУ),
К.В. Кринкин, к.т.н. (доц. каф. МОЭВМ, СПбГЭТУ)

РЕАЛИЗАЦИЯ МЕХАНИЗМА ПОДБОРА РЕКОМЕНДАЦИЙ В ИНФОРМАЦИОННОЙ СИСТЕМЕ “ОТКРЫТАЯ КАРЕЛИЯ”

Целью данной работы является разработка универсального интерфейса построения списка рекомендаций по заданному объекту в ИС с анонимным доступом “Открытая Карелия”[1]. Интерфейс должен удовлетворять следующим требованиям:

1. Гибкость с точки зрения замены критерия близости объектов.
2. Вычисление близости объектов должно учитывать индивидуальные множества тегов и полнотекстовые поля такие, как “Название”, “Полное описание”, “Аннотация”.

Для решения поставленной задачи был предложен подход, основанный на вычислении баллов близости между всеми объектами системы для различных критериев близости. Балл близости характеризует сходство двух объектов по определенному критерию. Величина

баллов близости объектов A и B в смысле критерия близости C прямо пропорциональна сходству объекта B с объектом A в рамках критерия близости C. C является совокупностью критерия отбора (предварительного ограничения на поля объекта B) и упорядоченного и взвешенного списка полей D. По критерию близости C производится сравнение объектов – сначала отбраковываются объекты, не соответствующие критерию отбора (для них значение баллов близости равно нулю), затем для каждого из элементов D производиться сравнение значений полей A и B. Сумма взвешенных количественных результатов сравнения отдельных полей представляет собой значение баллов близости для A и B.

Механизм построения рекомендаций был реализован в виде программного модуля для ИС “Открытая Карелия”. Работа с рекомендациями в рамках модуля разделена на два этапа:

1. Построение кэша рекомендаций. При каждом запуске бэкэнда “Открытой Карелии” происходит последовательное вычисление значений баллов близости для всех пар объектов системы и для всех критериев близости. Вычисленные значения индексируются совокупностью упорядоченной пары идентификаторов объектов и идентификатора критерия близости.
2. Предоставление программного интерфейса получения рекомендаций по идентификатору объекта и критерия близости. Выборка подборки рекомендованных объектов величины N осуществляется путем выбора N объектов, которым соответствуют N наибольших значений баллов близости при заданном критерии близости.

Выводы. Разработанная система построения рекомендаций для ИС “Открытая Карелия” представляет собой решение, основанное на понятии баллов близости. Данное решение является гибким - для построения нового типа рекомендаций необходим только критерий отбора, набор полей и весов, по которым будет вестись вычисление баллов близости. При этом система учитывает поля различной структуры – имеющие фиксированное количество значений, полнотекстовые и теги, что позволяет находить взаимосвязи между различными по своей природе объектами.

ЛИТЕРАТУРА:

1. Еврорегион Карелия: Музейный Гипертекст [Электронный ресурс] / Режим доступа: http://openkarelia.org/about, свободный

УДК 681.3, 004

В. А. Касилов (1 курс, каф. ИУС, СПбПУ)
И. А.Веренинов, к.т.н., доц.

ЭКОНОМИЯ ДИНАМИЧЕСКОЙ ПАМЯТИ В ПРИЛОЖЕНИЯХ СО МНОЖЕСТВОМ СТРУКТУР ДАННЫХ (НА ПРИМЕРЕ PASCAL)

Для работы с большим объёмом данных чаще используется динамическая память, в которой исходная информация представлена, как правило, в виде структур данных (Data Structure): списков, очередей, деревьев и других.

Случается, что одни и те же данные содержатся в нескольких структурах, поэтому при стандартной их организации исходная информация дублируется в динамической памяти, что может быть излишним.

Решение этой проблемы рассмотрим на примере однонаправленного списка. Два варианта организации такой структуры данных представлены в статье (Рис. 1).

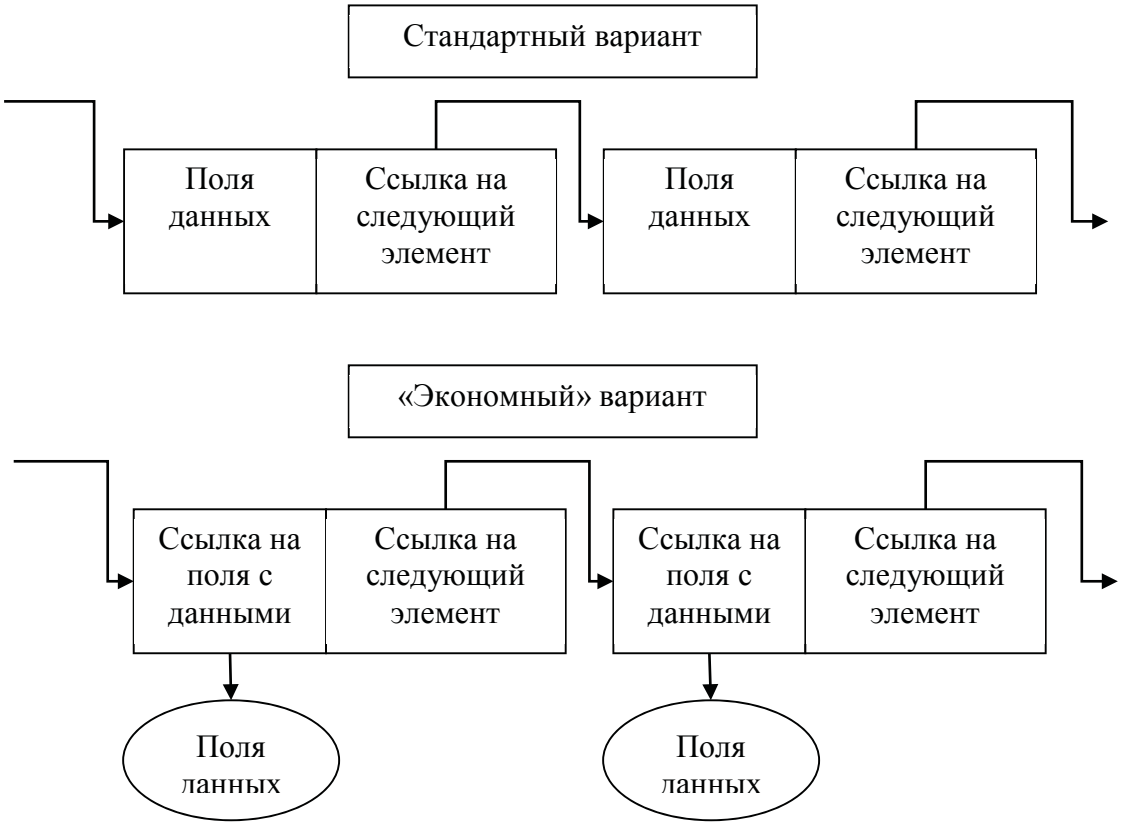


Рис. 1. Структура данных «Список» в стандартной и «экономной» формах

На первый взгляд кажется, что второй вариант никак не решает нашей проблемы экономии памяти, а только наоборот – усугубляет её. Это действительно так, если имеется только одна структура данных. Но в том случае, когда их несколько и одни и те же данных задействованы в двух и более структурах, преимущество становится очевидным: нам не требуется дублировать информацию в памяти – мы ссылаемся на уже имеющуюся. Таким образом, любая структура данных будет представлять собой набор ссылок: на исходные данные и на другие элементы структуры.

Большие объёмы информации чаще всего требуется упорядочить по некоторому признаку (например, отсортировать по одному из полей). Иногда, таких признаков может быть несколько. Тогда «экономная» организация позволяет создать нам несколько списков из ссылок, каждый из которых упорядочен в соответствии с заданным параметром и выделение дополнительной памяти не потребуется.

Из плюсов также стоит отметить более удобную работу со структурой в тех случаях, когда необходимо переставлять её элементы местами: при «экономной» организации подобное действие будет осуществляться в три шага.

Очевидно, что в такой структуре путь непосредственно к данным станет более «длинным» - это стоит отнести к её минусам.

Опираясь на опыт работы с подобной структурой, следует отметить, что имеет смысл выполнить её в объектном виде, где каждый её элемент будет представлять объект с необходимыми свойствами (например, выводить или не выводить элемент), а сама структура данных будет состоять из этих элементов и обладать своими свойствами (например, отсортирован список или нет). Если рассматривать пример со списком, то его «новый вид» можно представить, как показано на рисунке (Рис. 2).

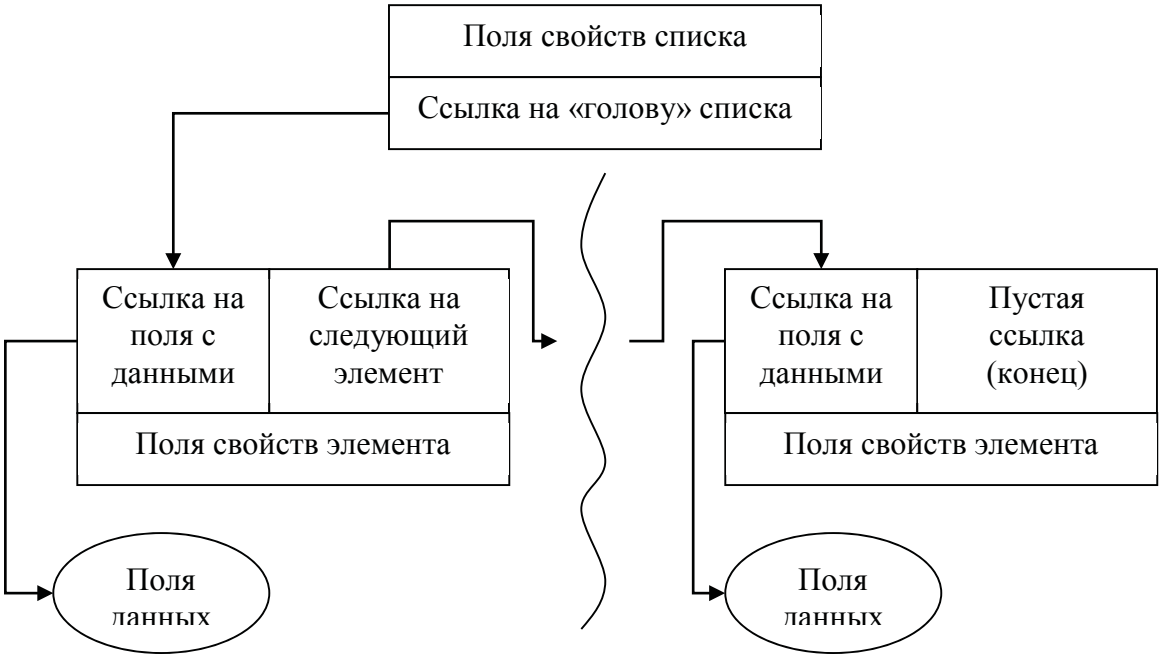


Рис. 2. Схема «экономной» структуры данных «Список» в объектной форме

Кроме того, эффективность «экономной» организации возрастает с количеством повторений, так как ссылочный тип занимает меньший объём памяти по сравнению с большинством других типов данных. Другими словами, качественная оценка такова: по памяти мы выигрываем во столько раз, сколько бы исходные данные повторялись в различных структурах данных.

В общем, основная идея заключается в том, чтобы исключить дублирования исходной информации в динамической памяти. Единожды выделив место в памяти под информацию, используем только ссылки на неё. Поэтому стандартные структуры данных заменяются доработанными – «ссылочными», что и обеспечивает выигрыш по памяти.

Эта идея непременно найдёт своё отражение не только в языке Pascal, но и в других языках программирования высокого уровня.

Документирование работы осуществлено с применением средств MS Office.

УДК 621.37

С.В.Стрельников (4 курс, каф. ИУС, СПбПУ)

СОЗДАНИЕ ВЕБ-ИНТЕРФЕЙСА ДЛЯ АДМИНИСТРИРОВАНИЯ СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ MYSQL

Актуальность работы: база данных (БД) может облегчить работу с информацией, необходимой для организации выдачи книг на абонементе библиотеки.

Цель исследования: получение навыков создания веб-ориентированных программных средств для прикладных задач (управления базами данных). Для реализации поставленной цели были выполнены следующие задачи:

1. Осуществлен анализ существующих технологий, видов и структур баз данных;
2. Обеспечена возможность получения данных по всем необходимым запросам;
3. Обеспечена целостность данных, правильность их содержания, исключены потери и противоречия в их содержании;
4. Описаны результаты исследования и сформулированы перспективные решения.

Для реализации данных задач были выбраны следующие программные средства:

- Denwer. Пакет программных средств для организации веб-сервера на локальном компьютере;
- PHP. Язык серверных сценариев для выборки данных из БД и формирования веб-интерфейса системы;
- PHPmyAdmin. Веб-интерфейс для администрирования системы управления базами данных (СУБД);
- Dreamweaver и Notepad++. Среды разработки сценариев.

В рамках проведенного исследования были реализованы следующие функциональные возможности веб-интерфейса для администрирования СУБД:

- создание новой базы данных;
- создание таблиц в базе;
- заполнение полей в таблице;
- просмотр результата;

Таким образом, в результате исследования было спроектировано и реализовано средство для создания баз данных, получены навыки и умения работы со следующими программными средствами:

- системой управления базами данных MySQL, создан веб-интерфейс для администрирования, произведено тестирование и эксплуатация;
- интегрированными средами разработки Notepad++ и AdobeDreamweaver.

ЛИТЕРАТУРА:

1. Теория СУБД – учебно образовательный портал лекции-онлайн [Электронный ресурс] — Режим доступа: http://www.mylect.ru/informatic/bd/300-subd.html?showall=1, свободный — Загл. с экрана.

2. Данные, базы данных [Электронный ресурс] — Режим доступа: http://www.sergeeva-i.narod.ru/inform/page6.htm, свободный — Загл. с экрана.
3. Zend Studio - the PHP IDE for professionals [Электронный ресурс] — Режим доступа: http://www.zend.com/en/products/studio/, свободный — Загл. с экрана.
4. PHP Expert Editor - Ankord Development Group. [Электронный ресурс] — Режим доступа: http://www.ankord.com/ru/phpxedit.php, свободный — Загл. с экрана.
5. Программное обеспечение для редактирования HTML, программное обеспечение для веб-дизайна | Adobe Dreamweaver CS6 [Электронный ресурс] — Режим доступа: http://www.adobe.com/ru/products/dreamweaver.html, свободный — Загл. с экрана.
6. Notepad++ Home [Электронный ресурс] — Режим доступа: http://notepad-plus-plus.org/, свободный — Загл. с экрана.
7. Как базы данных используются в повседневной жизни Cardomat.ru [Электронный ресурс] — Режим доступа: http://cardomat.ru/article/kak-bazy-dannyh-ispolzuyutsya-v-povsednevnoj-zhizni, свободный — Загл. с экрана.
8. Методология IDEF0 [Электронный ресурс] — Режим доступа: http://itteach.ru/bpwin/metodologiya-idef0/all-pages, свободный — Загл. с экрана.

УДК 681.3, 004

К.С. Кобышев (1 курс, каф. ИУС, СПбПУ)
И.А.Веренинов, к.т.н., доц.

МЕТОДИКА ОБЪЕКТНО-ОРИЕНТРОВАННОГО ПРОГРАММИРОВАНИЯ НА ПРИМЕРЕ ИГРЫ ("ТАНЧИКИ"), РАЗРАБОТАННОЙ НА TURBO PASCAL 7.0

Еще в середине 70-х годов XX века объектно-ориентированное программирование вытеснило процедурное ввиду того, что обладает рядом преимуществ над ним. Существует множество методик объектно-ориентированного программирования, позволяющих эффективно реализовать сложные алгоритмы. Цель данного исследования - рассказать об одной из методик, использованной при разработке приложения, написанном на объектно-ориентированном языке TURBO PASCAL 7.0.

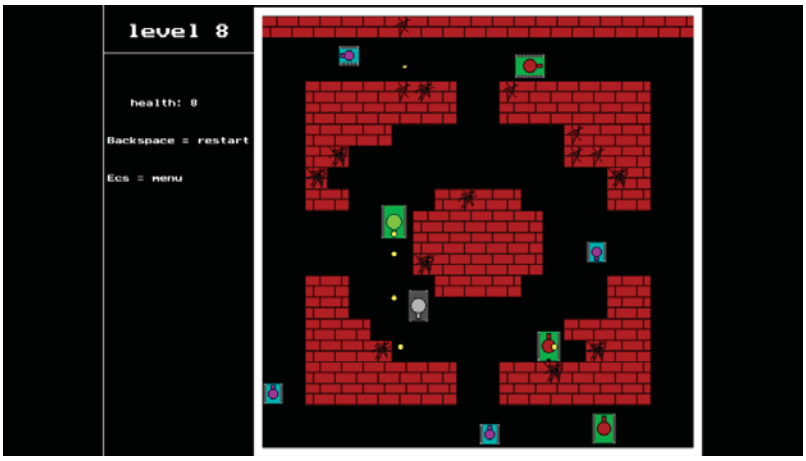


Рис. 1. Скриншот игры.

В данной игре активные компоненты представлены в виде объектов, а именно: игрок, противники, текстуры. Каждый из объектов содержит поля, методы, которые осуществляют его динамику. Поля представлены как в виде элементарных переменных, так и в виде простых структур данных (односвязных списков). Более подробная структура объектов представлена ниже:

Состав	Игрок	Противник	Текстуры
Поля	Длина, ширина, координаты x и y, скорость, направление движения, цвет 1 и цвет 2, количество жизней, список снарядов (у каждой элемента списка - координаты x и y, направление движения снаряда), список вражеских снарядов.	Наследованные поля игрока + указатель на игрока-объекта.	Координаты x и y, количество жизней, параметр для прорисовки, список снарядов игрока и противников.
Методы	Прорисовка, стирание игрока, поиск случайных координат при инициализации, проверка корректности положения (чтобы избежать прорисовки над текстурами, границей игрового поля), смещение координат (движение игрока в определенном направлении), инициализация (формирование первоначальных параметров).	Наследованные методы игрока + переопределенный метод смещения координат (движение противника в зависимости от положения игрока, случайного выбора направления, корректности положения), дополнительная инициализация (получение доступа к данным игрока-объекта через указатель).	Инициализация (формирование первоначальных параметров), перерисовка в случае попадания снаряда.

Таким образом, применение процедурного программирования для реализации взаимосвязи между активными компонентами игры было бы крайне неэффективным - без четкой структуризации в виде объектов усложнились бы следующие взаимодействия между компонентами игры: попадание снаряда противника в игрока, попадание снаряда игрока в противника, попадание снаряда противника или игрока в текстуру, реагирование противника на приближение игрока с некоторой вероятностью (движение в сторону игрока и стрельба в его сторону).

УДК 004.932.(72)

В.К. Костебелова (5 курс, каф. ИУС, СПбГПУ),
Т.А. Вишневская, доц.

РАЗРАБОТКА ЭФФЕКТИВНОГО МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ СОЗДАНИЯ БЛОК-СХЕМ НА ПЛАТФОРМЕ ANDROID

Целью данной работы является разработка эффективного мобильного приложения на платформе Android для создания и редактирования блок-схем, нарисованных от руки.

В работе были поставлены и решены следующие задачи:

1. Проанализировать существующие мобильные приложения для рисования блок-схем
2. Проанализировать существующие методы распознавания простых геометрических фигур
3. Разработать эффективный алгоритм распознавания фигур, нарисованных от руки
4. Разработать алгоритм распознавания действий пользователя
5. Создать прототип мобильного приложения для создания блок-схем, нарисованных от руки
6. Провести тестирование и оценить эффективность разработанного продукта

В рамках теоретического исследования было изучено 8 методов распознавания геометрических фигур: преобразование Хафа, преобразование Радона, метод стохастического поиска, BFOA, преобразование Фурье над контуром (контурный анализ), нейронные сети, метод статистических оценок, метод классификаций.

Для решения поставленной задачи был выбран метод статистических оценок и на его основе разработан собственный эффективный алгоритм распознавания фигур, нарисованных от руки. В качестве статистических параметров для определения класса фигуры использовалось два основных параметра E (определяет линия это или объект) и H (определяет эллипс это или прямоугольник). Сложность разработанного алгоритма распознавания – линейная и составляет около O(3n), где n – это количество точек в контуре.

Разработанный прототип мобильного приложения написан на языке Java в среде разработки Eclipse и совместим со всеми мобильными устройствами под управлением операционной системы Android 2.2 и выше. Прототип состоит из 15 классов, сгруппированных в 4 модуля: Activity, Model, Draw, ReadWrite.

Оценка эффективности разработанного приложения проведена согласно требованиям международного стандарта ISO 9126, описывающего качество программного продукта. В данной работе основные параметры эффективности – это время работы, занимаемая оперативная память и точность распознавания.

В процессе тестирования было создано 10 тестовых функций, все они завершились успешно. Фигуры, состоящие из 100 000 точек, распознаются приложением менее чем за секунду, а фигуры, состоящие из 100 точек, за 0,001 с (для устройства Samsung Galaxy S3). Разработанное мобильное приложение потребляет 9,02 Мб оперативной памяти, для сравнения, бесплатная версия альтернативной программы требует 20-30 Мб.

Таким образом, создано эффективное мобильное приложение, которое обладает высокой производительностью и требует небольшого количества ресурсов.

Практическая значимость данной работы заключается в том, что разработанное мобильное приложение позволяет быстро и легко создавать, редактировать и сохранять блок-схемы, нарисованные от руки. Данное приложение могут использовать не только разработчики программного обеспечения, но и управляющий персонал. Ученики, студенты и преподаватели могут использовать приложение в обучающих целях в любом месте с помощью смартфона или планшета.

УДК 681.3.06

А. О. Латыпова (5 курс, каф. ВТиИТ, СПбГМТУ),
В.А. Семенова-Тян-Шанская, к.т.н., доцент

РАЗРАБОТКА WEB-ОРИЕНТИРОВАННОГО OLAP-КЛИЕНТА

Целью работы является разработка клиентского приложения, позволяющего эффективно использовать OLAP-технологию для интерактивного анализа данных отдела снабжения строительной фирмы. Данная технология дает возможность пользователю приложения проанализировать графически представленные данные, упрощающие контроль над отделом и стратегическое планирование фирмы.

Для реализации проекта был выбран реляционный способ хранения данных, то есть данные для многомерного представления извлекаются из реляционных структур. Такой подход в рамках заданной предметной области имеет ряд преимуществ, основное из которых позволяет приложению функционировать на менее мощных клиентских станциях. В качестве схемы реализации ROLAP была выбрана схема «снежинка».

Для работы используются методы и инструменты Business Intelligence. Технология BI позволяет обрабатывать большие объемы неструктурированных данных, чтобы найти стратегические возможности для бизнеса.

Данные для анализа, хранимые в Microsoft SQL Server Analysis Services, могут быть получены выполнением запросов на языке MDX. В случае ROLAP процессор многомерных запросов транслирует многомерные запросы в SQL-запросы, которые выполняются реляционной СУБД. Результат MDX-запроса представляет собой многомерное подпространство – подкуб, который может содержать множество измерений.

Для разработки Web-клиента для OLAP-сервера выбрана трехуровневая архитектура, в которой Web-сервер выступает в роли промежуточного звена между клиентом и сервером баз данных. Такая схема позволяет эффективно использовать возможности Интернета и упрощает работу с приложением для пользователя, так как он использует уже хорошо знакомый ему интерфейс Web-страниц.

В качестве Web-сервера используется IIS, который в работе используется как сложный сервер приложений, предоставляющий широкое множество функциональных средств, наиболее важным из которых является поддержка хостинга приложений ASP.NET.

В качестве OLAP-сервера используется Microsoft SQL Server 2008 R2.

УДК 004.5

С.А. Марченков (6 курс, каф. ИМО, ПетрГУ)
А.С. Вдовенко (6 курс, каф. ИМО, ПетрГУ)
Д.Ж. Корзун, к.ф.-м.н., доцент

СЕРВИС ПОСТРОЕНИЯ КУЛЬТУРНОЙ ПРОГРАММЫ ДЛЯ УЧАСТНИКОВ КОНФЕРЕНЦИИ В ИНТЕЛЛЕКТУАЛЬНОМ ЗАЛЕ

Многоагентная система ИЗ [1] развертывается на оборудовании в пределах аудитории с подключением к сети Интернет. Предоставляется набор сервисов для проведения таких мероприятий коллаборативной деятельности, как конференция. Взаимодействие агентов основано на общем «интеллектуальном пространстве». Участники получают доступ к сервисам через мобильного клиента [2]. Целью данной работы является разработка сервиса ИЗ для построения культурной программы во время проведения конференции, а участники приехали из разных регионов. Сервис выполняет поиск достопримечательностей места проведения конференции в открытых источниках. Участники голосуют за эти места, наблюдая и решения других участников. Архитектура показана на рис. 1.

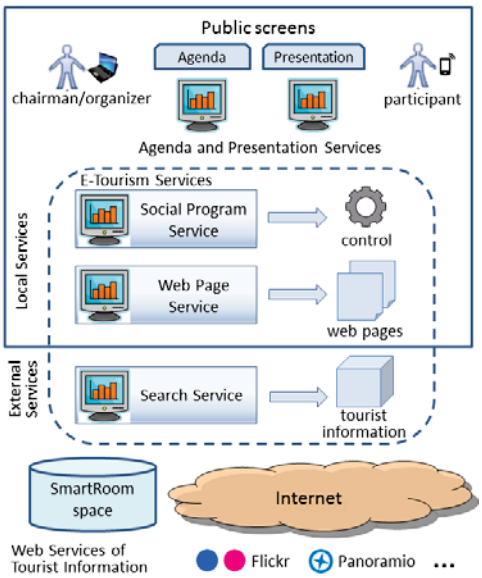


Рис. 1 Архитектура системы ИЗ с сервисом построения культурной программы

Основным управляющим сервисом является Social Program service. Он предоставляет организаторам управление параметрами и ходом построения. Сервисы Search service и Web Page service выполняют вспомогательные задачи (и могут использоваться и другими сервисами). Первый выполняет поиск туристической информации в сети Интернет. Второй используется для управления web-страницами, которые генерируются по содержимому, накапливаемому сервисом Social Program service, и передаются мобильными клиентам участников. Взаимодействие сервисов использует механизмы подписки и REST-запросов.

Разработка выполняется при финансовой поддержке Минобрнауки России в рамках проектной части государственного задания в сфере научной деятельности (НИР № 2.2336.2014/К). Работа поддержана федеральной целевой программой «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2014-2020 годы», соглашение № 14.574.21.0060 (RFMEFI57414X0060).

ЛИТЕРАТУРА:

1. D. Korzun, I. Galov, A. Kashevnik, S. Balandin. Virtual shared workspace for smart spaces and M3-based case study. Proc. 15th Conf. of Open Innovations Association FRUCT. SPb.: ITMO Univeristy, 2014. P. 60-68.
2. A. Vdovenko, S. Marchenkov, D. Korzun. Mobile Multi-Service Smart Room Client: Initial Study for Multi-Platform Development. Proc. 13th Conf. Open Innovations Framework Program FRUCT and 2nd Regional Seminar on e-Tourism. SPb. SUAI, 2013. P. 143-152.

УДК 681.3

А. Д. Носов (4 курс, каф МОЭВМ, СПбГЭТУ).

РАЗРАБОТКА СИСТЕМЫ СБОРА, ОБРАБОТКИ И АНАЛИЗА ДАННЫХ ОТ ДАТЧИКА ВИДЕО

Цель данной работы состояла в разработки инструментов для сбора, обработки и анализа данных, полученных от датчика видео. В качестве датчика использовалась камера мобильного телефона.

Основные реализуемые функции и требования:

1. Возможность съемки и покадровой обработки видео
2. Возможность ассоциации каждого кадра с гео-тегом(совокупность параметров: время, географическое положение и ориентация камеры в пространстве)
3. Возможность загрузки полученных изображений на сервис хранения и обработки(Image-сервис)
4. Возможность получения изображений от Image-сервиса по заданным параметрам(время, местоположение)
5. Возможность построения сцены/панорамы.

В ходе выполнения работы были поставлены и решены следующие задачи:

1. Разработка графического интерфейса пользователя.
2. Разработка клиентского приложения, осуществляющее съемку видео, его покадровую обработку и загрузку его на сервер, а также считывание данных с GPS и акселерометра.
3. Реализация сервиса хранения и обработки кадров/видео.
4. Реализация возможности поиска фрагментов видео по заданным параметрам.
5. Обеспечение взаимодействия всех компонентов
6. Реализация алгоритма построения сцены

Сервис хранения и обработки изображений представляет собой RESTful веб сервис, разработанный на базе фреймворка Ruby on Rails с использованием MongoDB. Данные инструменты были выбраны исходя из соображений удобства хранения графической информации, а также скорости и удобства разработки.

Процесс построения сцены был разбит на следующие шаги:

1. Получение изображений
2. Размещение изображений относительно друг друга

3. Нахождение общих точек и их дескрипторов методом SIFT(Scale Invariant Feature Transform)
4. Отсеивание ложных совпадений точек. Алгоритм RANSAC(RANdom Sample Consensus)
5. Построение матрицы гомографии
6. Проецирование с помощью полученной матрицы гомографии одного изображения на другое

Выводы. В результате работы был получен набор инструментов, совместное использование которых позволяет производить обработку изображений, полученных с одной или нескольких камер, а также получать цельную картину – панораму, собранную из этих изображений.

ЛИТЕРАТУРА:

1. Image Mosaics. Jonathan Brumberg. URL: http://cns.bu.edu/~brumberg/CS580/mosaic/MAIN/mosaic.html
2. Image Stitching. URL: https://courses.engr.illinois.edu/cs498dwh/fa2010/lectures/Lecture%2017%20-%20Photo%20Stitching.pdf
3. Automated Panorama Stitching. URL: http://cs.brown.edu/courses/csci1950-g/results/proj6/edwallac/
4. Построение SIFT дескрипторов и задача сопоставления изображений. URL: http://habrahabr.ru/post/106302/

УДК 621.37+621.391.8

Д.И. Сергеев (4 курс, каф. ИУС, СПбГПУ),
В.С.Тутыгин, к.т.н., доц.

НЕДОРОГАЯ СИСТЕМА ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ ДВИЖЕНИЯ ТРАНСПОРТА В ПРОИЗВОДСТВЕННЫХ ПОМЕЩЕНИЯХ

Безопасность движения малоскоростного (2-10 км/ч) транспорта в цехах и складских помещениях обеспечивается с помощью ультразвуковых датчиков и лазерных дальномеров. Недостатком первых является отсутствие какого-либо угла обзора, а вторых, использующих лазерные сканеры Sick, высокая стоимость. Использование сенсора Microsoft Kinect в сочетании с ПО Microsoft Kinect SDK [1] и библиотеки OpenCV [2] фирмы Intel позволило создать недорогую систему обеспечения безопасности движения малоскоростного транспорта в производственных помещениях.

Предлагаемая система обеспечения безопасности движения (СОБД) включает сенсор Microsoft Kinect, малогабаритный (14x10x3 см) компьютер Barebone PC N3520 QuadCore 2.166 Ghz и модуль светодиодной индикации. Сенсор Microsoft Kinect позволяет получить метрическую карту глубины на расстоянии до 8 м., что позволяет своевременно обнаруживать объекты в секторе движения погрузчика.

В качестве объекта, для которого проектировалась предлагаемая система обеспечения безопасности движения, мы рассматривали автопогрузчик, параметры движения которого в производственных помещениях регламентированы ГОСТ.

Алгоритм работы СОБД заключается в обработке карты глубины, возвращаемой сенсором, и определении текущего уровня опасности движения. Предложено выделить 3 уровня опасности:

1. Зеленый - дальность до ближайшего препятствия более 3-х метров, скорость не ограничена.
2. Желтый - обнаружено препятствие на расстоянии от 2-х до 3-х метров, ограничение скорости до 4-х км/час.
3. Красный - препятствие ближе 2-х метров, ограничение скорости до 2-х км/час.

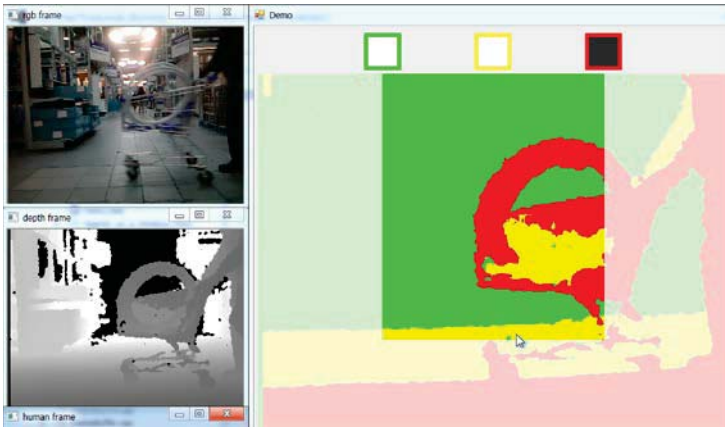


Рис. 1 Пример обработанной карты глубин

При приближении погрузчика к месту погрузочно-разгрузочных работ, например к стеллажу, скорость будет ограничена, и водитель автопогрузчика будет выполнять операции погрузки и отгрузки товара со скоростью не более 2 км/час. В случае, если в зону движения погрузчика вышел человек, либо движущийся объект (например, грузовая тележка), скорость также будет ограничена.

Для потоковой обработки кадров глубины и взаимодействия с сенсором “Kinect” система использует драйвер Microsoft Kinect и пакет Microsoft Kinect SDK.

Уровень опасности определяется независимо для каждого кадра потока, что позволяет достичь высокой скорости обнаружения объектов - 70 мс.

Карта глубины строится по пороговым значениям уровней опасности в метрах (рис. 1). Каждый цвет обрабатывается отдельно. Вначале обрабатываются области красного цвета, затем желтого, если нет ни красного, ни желтого, детектируется зеленый.

Чтобы определить, присутствует ли уровень опасности заданного цвета, был разработан алгоритм обработки карты глубины (рис. 2).

Бинаризация исходного изображения производится с порогом, соответствующим уровню опасности.

Метод поиска границ базируется на морфологических преобразованиях, в частности на эрозии изображения с окном 3x3: $I_f = I_b - I_b \oplus P$, где I_b и I_f – бинарное и граничное изображения соответственно, $I \oplus P$ – эрозия изображения I по примитиву P. При этом яркость пикселя на изображении границ равна разности между текущим и минимальным значением в данном окне:

Рис. 2. Схема алгоритма обработки карты глубин

$I_f(x,y) = I_b(x,y) - \min_{s,t} I_b(x+s, y+t)$, для всех $(x,y) \in I_b$, $s, t = -1..1$.

Для поиска контуров использовалась модификация обнаружения внешних контуров алгоритма [3]. Преимущества данного алгоритма в его высокой скорости и возможности выделять только внешние контуры.

При реализации алгоритма мы использовали функции пакета OpenCV: cv::threshold для бинаризации и cv::findContours для поиска контуров. Из найденных контуров, имеющих общую часть с целевой областью, выбирается контур с наибольшей площадью. Для этого вызывается функция cv::contourArea. Уровень считается обнаруженным, если относительная площадь наибольшего контура превышает пороговую.

Испытание предлагаемой системы в условиях складского помещения – в магазине «Максидом» – с использованием тележки в качестве макета погрузчика показало, что система обладает необходимыми характеристиками быстродействия, а также удобна в применении. Система своевременно обнаружила все статические и динамические объекты, угрожающие столкновением с макетом и не ограничивала скорость при их отсутствии. В качестве компьютера в макете системы был использован ноутбук.

Таким образом, использование технологий Microsoft и Intel при создании программного обеспечения позволяет создать компактные недорогие промышленные системы обеспечения безопасности движения транспорта в производственных помещениях.

ЛИТЕРАТУРА:

1. Kinect for Windows SDK from Microsoft Research [Электронный ресурс] – Режим доступа: http://research.microsoft.com/en-us/um/redmond/projects/ kinectsdk/.
2. OpenCV library [Электронный ресурс] – Режим доступа: http://sourceforge.net/projects/opencvlibrary/.
3. Suzuki, S. and Abe, K., Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30 1, pp 32-46 (1985).

УДК 681.3

М. И. Шехтман (6 курс, каф. МО ЭВМ, СПБГЭТУ)
К.В. Кринкин, к.т.н., доцент.

СОЗДАНИЕ УНИВЕРСАЛЬНОГО ВИРТУАЛЬНОГО УСТРОЙСТВА ДЛЯ ХРАНЕНИЯ И СЖАТИЯ ДАННЫХ В РЕЖИМЕ ОНЛАЙН

Главная тенденция развития в окружающем нас компьютерном мире – переход от экстенсивной модели развития к интенсивной. Развитие твёрдотелых накопителей не поспевает за огромным объемом информации. Отсюда возникают следующие проблемы: большое количество повторяющихся копий данных, неэффективное использование пространства, низкая скорость передачи данных. В случае вопросов хранения и обработки информации так просто от этих проблем не избавиться.

На сегодняшний день существуют следующие варианты решения: дедупликация данных, массовые системы и виртуальные устройства для хранения и передачи информации. Но все эти решения не доступны для обычных пользователей ввиду своей дорогой стоимости и сложности в использовании.

Целью данной работы является исследование возможностей современных технологий хранения и алгоритмов компрессии данных, а также проектирование универсального устройства для хранения и сжатия информации в онлайн-режиме, то есть при непосредственной работе с ней.

Устройство работает без участия пользователя, который сам может выбрать алгоритм сжатия данных. Оно хранится виртуально в системе и при операциях с данными сжимает их в начале и распаковывает по завершению. Сжатие информации позволяет избежать вышеперечисленных проблем хранения данных при работе с ними. Для сжатия и распаковки используются алгоритмы компрессии данных показавшие наилучшие результаты при тестировании:

Алгоритм	Коэффициент Сжатия	Скорость Сжатия,мб/с	Скорость распаковки,мб/с
LZ4 (r41)	2.08	319	1070
lzw	2.02	257	1150
lzo	2.07	321	950
zlib	2.74	46	179

Таблица 1. Тестирование алгоритмов компрессии данных

Разработанное устройство является готовым продуктом и для установки не требует усилий и особых умений.

В перспективе предполагается увеличить количество алгоритмов компрессии для выбора их пользователем.

УДК 004.75

К. Г. Юденок (асп., каф. МОЭВМ, СПбГЭТУ)
К. В. Кринкин, к.т.н., доцент

ПРИМЕНЕНИЕ И ИСПОЛЬЗОВАНИЕ ГЕОГРАФИЧЕСКОГО КОНТЕКСТА В ИНТЕЛЛЕКТУАЛЬНОМ ПРОСТРАНСТВЕ (SMART SPACES)

В последние несколько лет существенно повысилось внимание научного сообщества к технологиям, предоставляемым так называемым стеком технологий Web 2.0, который ставит во главу угла вовлеченность пользователя в генерацию контента, распределенную сервисную архитектуру построения приложений, контекстно-зависимую обработку информации и человеко-машинный интерфейс. Прежде всего, это обусловлено распространением мобильных устройств и беспроводных коммуникационных возможностей, позволяющих пользователям постоянно находится в режиме «онлайн» и быть интегрированными с глобальными информационными ресурсами и сервисами.

Ключевым научным направлением в этой области является концепция «Повсеместных вычислений» и его основная парадигма «Интеллектуальные пространства» (от англ. Smart Spaces, далее ИП), которое должно обеспечивать динамическое множество участников вычислительной среды контекстно-зависимой информацией, сервисами и персонифицированными рекомендациям. Также стоит отметить вид сервисов, основанных на местоположении пользователей (от англ. Location-Based Services, далее LBS-сервисы), как один из популярных контекстно-зависимых сервисов.

Городецкий В.И. выделил следующие проблемы концепции ИП – 1) проблема интеграции в единую систему разнородных моделей, аппаратных средств, коммуникационных протоколов, пользовательских интерфейсов и программного обеспечения, 2) необходимость совместной обработки гетерогенной информации, получаемой из распределенных источников для формирования текущего контекста ИП, поиска веб-сервисов, прогноза контекста и решения задачи принятия решений и другие.

В ИП особое место занимают контекстные данные, которые необходимы для использования наиболее актуальных на текущий момент знаний о текущей ситуации объектов пространства. Одной из главных частей контекста являются данные о местоположении. Эти данные могут быть использованы в двух случаях 1) для уточнения семантики запросов, когда сервис получает данные из ИП; 2) для ограничения пространства поиска данных (нет необходимости делать заного глобальный поиск).

LBS-сервисы используют метод геотеггинга для разметки реальных или виртуальных объектов. Это основа для определения географического контекста, который может быть использован в различных типах приложений, например, как семантический информационный поиск или межмашинное взаимодействие (M2M).

Целью работы является разработка архитектуры, модели и методов представления и использования географической информации в ИП, обеспечивающих своевременную обработку геоконтекстной информации в ИП. Другими словами, основная цель – интеграция платформы ИП и LBS-системы для моделирования и обработки геоконтекстной информации в ИП. Для этого решаются проблемы интеграции разнородных моделей исследуемых платформ и их совместная обработка геоконтекстной семантической информации в ИП.

С практической точки зрения используется платформа ИП Smart-M3 и LBS-система Geo2Tag. Интеграция проходит через специальный компонент – медиатор или агент интеграции (процессор знаний). В качестве технологии решения проблемы интеграции платформ используется стандарт концепции «Повсеместных вычислений» – ***Ubiquitous ID.***

В результате разработаны основные механизмы моделирования и использования географического контекста в ИП на основе платформ Smart-M3 и Geo2Tag. Следующим этапов проекта является оптимизация платформы ИП Smart-M3 и её миниатюризация для использования во встраиваемых устройствах (концепция «Интернет вещей»).

Секция «Программная инженерия: инструментальные средства и технологии проектирования и разработки»

УДК 004.492.3

К.И.Алдонин (6 курс, каф. МОЭВМ, СПбГЭТУ),
Н.Н.Варлинский, к.т.н., доцент
А.К.Большев, к.т.н.

РАЗРАБОТКА СЕТЕВЫХ СИСТЕМ ДЛЯ УПРАВЛЕНИЯ СОБЫТИЯМИ

Целью работы является разработка программного обеспечения, выполняющего функции сетевой системы для управления событиями. Эта система должна обеспечивать автоматизацию процесса обнаружения инцидентов, консолидацию и хранение журнала событий от различных источников, а также своевременно информировать о произошедшем событии.

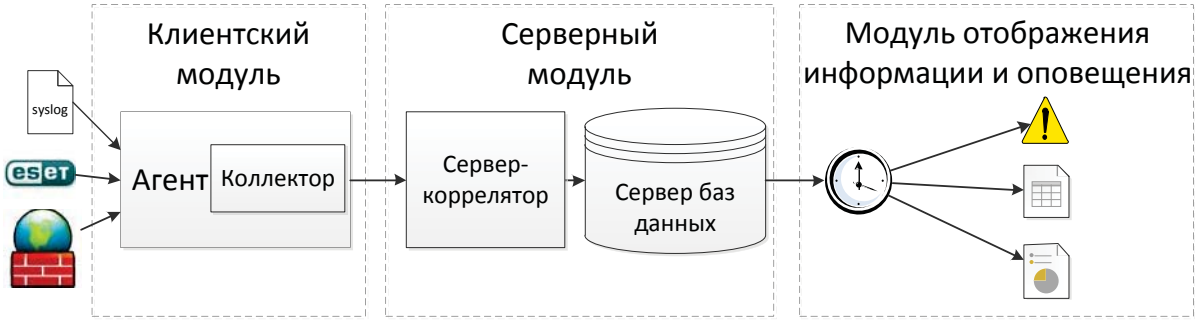


Рис.1 Состав системы управления событиями

В качестве клиентского модуля разрабатываемой системы выступает Агент, демон-программа, включающая в себя коллектор. Данный модуль предназначен для сбора информации от различных источников, среди которых могут выступать журналы событий серверов и рабочих станций, счетчики сетевого трафика, сетевые системы обнаружения вторжений, антивирусные программы, сканеры уязвимостей и др.

Далее собранная информация поступает на сервер-коррелятор, основная задача которого состоит в унификации полученных данных и обработка/фильтрация событий. В качестве формата передачи файлов по сети выбран XML формат, поскольку, с одной стороны, это язык с простым формальным синтаксисом, удобным для создания и обработки документов, а с другой – существует множество библиотек, поддерживающих быструю работу с данным форматом данных (в данной работе был использован модуль QtXml фреймворка Qt).

В качестве обработки/фильтрации событий был выбран метод опорных векторов (SVM), ввиду того, что для классификации достаточно небольшого количества данных, а также можно строить линейные классификаторы для работы с линейно неразделяемыми данными, сочетая при этом простоту и эффективность, не теряя при этом точность оценки. Однако метод относиться к категории алгоритмов обучения с учителем, поэтому перед его использованием

происходит обучение распознающей машины, и от качества и точности обучения будет зависеть точность классификатора.

Собранная и унифицированная информация с результатом классификатора передаются в сервер без данных, в качестве которого выступает SQLite база данных. SQLite является встраиваемой базой данных, ввиду чего отпадает необходимость дополнительной установки баз данных, основанных на клиент-серверной парадигме. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. Ввиду того, что запись в базу данных ведется только посредством сервера-коррелятора, не возникает ограничений на одновременную запись в базу.

На выделенном узле обслуживаемой сетевой инфраструктуры посредством настраиваемого демона предоставляется информация и статистика о хранимых инцидентах базы данных. Через всплывающие сообщения по таймеру или при изменении в базе данных происходит оповещение о возможных угрозах сети.

УДК 519.685.3

Г.Г. Александрия (3 курс, каф. СП, факультет мат-мех, СПбГУ)
С.В. Григорьев, магистр информационных технологий, аспирант, СПбГУ

СРАВНЕНИЕ ИНСТРУМЕНТОВ АНАЛИЗА ДИНАМИЧЕСКИ ФОРМИРУЕМЫХ СТРОКОВЫХ ВЫРАЖЕНИЙ

Программное обеспечение может содержать строковые выражения в виде исходного кода на другом, встроенном, языке. Такие выражения формируются динамически из строковых литералов во время выполнения программы и передаются во внешнюю среду на исполнение. Соответственно, выражения встроенного языка — это тоже код на некотором языке программирования, и с ними возникают те же проблемы, что и при разработке на обычных языках. Одна из главных проблем — отсутствие возможности статической проверки динамически формируемых строковых выражений, и, как следствие, любая незначительная ошибка в них обнаруживается лишь на этапе исполнения. Это значительно увеличивает затраты на разработку, отладку и сопровождение продуктов и приложений, в которых используются встроенные языки. Другой проблемой является то, что при разработке приложений, содержащих выражения на встроенном языке, нет возможности использовать функции для работы с этими выражениями, которые могут упростить процесс разработки и облегчить поддержку кода. Например, простые функции интегрированных сред разработки: подсветка синтаксиса, авто-дополнение, рефакторинг, некоторые подсказки. Анализ динамически формируемых выражений также необходим и при реинжиниринге программного обеспечения. Например, при наличии динамически формируемых строковых SQL запросов, не проанализировав строковые выражения, нельзя точно сказать, с какими объектами в базе данных взаимодействует система. А значит, нет возможности гарантировать безопасность изменений в структуре базы данных.

На сегодняшний день существует ряд инструментов для анализа строковых выражений: Alvor [1], Java String Analyzer (JSA) [2], PHP String Analyzer (PHPSA) [3], IntelliLang [4] и YaccConstructor [5] и другие. Несмотря на то, что они решают схожие задачи, все они имеют достоинства и недостатки, отличающие их друг от друга. Целью работы является сравнение

инструментов анализа динамически формируемых строковых выражений и выявление направления дальнейшего пути развития разработки инструментов анализа.

В результате работы выявлено, что существующие инструменты либо решают узкую задачу, как правило, проверка корректности выражений, и расширение их функциональности затруднено (к таким анализатором относятся JSA и PHPSA), либо предоставляют широкий набор функций, но не поддерживают сложные и частые случаи формирования выражений, как, например, IntelliLang. При этом существуют многофункциональные инструменты, такие как Alvor, реализующие мощный алгоритм анализа, но не обладающие архитектурой, позволяющей создавать новые инструменты. Таким образом, необходима разработка набора готовых унифицированных компонент, упрощающих создание инструментов для решения различных задач анализа динамически формируемых строковых выражений. Разработка такого инструментария ведётся на основе проекта YaccConstructor, так как он изначально разрабатывался как модульный инструмент и предоставляет необходимую инфраструктуру для расширяемости.

ЛИТЕРАТУРА:

[1] Aivar Annamaa, Andrey Breslav, Jevgeni Kabanov, and Varmo Vene. 2010. An interactive tool for analyzing embedded SQL queries. In *Proceedings of the 8th Asian conference on Programming languages and systems* (APLAS'10), Kazunori Ueda (Ed.). Springer-Verlag, Berlin, Heidelberg, 131-138.

[2] Aske Simon Christensen, Anders Møller, and Michael I. Schwartzbach Precise Analysis of String Expressions BRICS, Department of Computer Science University of Aarhus, Denmark

[3] Yasuhiko Minamide. 2005. Static approximation of dynamically generated Web pages. In *Proceedings of the 14th international conference on World Wide Web* (WWW '05). ACM, New York, NY, USA, 432-441.

[4] https://www.jetbrains.com/idea/help/intellilang.html — сайт проекта IntelliLang

[5] Semen Grigorev, Ekaterina Verbitskaia, Andrei Ivanov, Marina Polubelova, and Ekaterina Mavchun. 2014. String-embedded language support in integrated development environment. In *Proceedings of the 10th Central and Eastern European Software Engineering Conference in Russia* (CEE-SECR '14). ACM, New York, NY, USA, , Article 21 , 11 pages.

УДК 004.42

Т.В. Коваленко (3 курс, каф. ИУС, СПбПУ),
Т.В. Коликова, доц.

СОЗДАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ПРИЛОЖЕНИЯ ДЛЯ УПРАВЛЕНИЯ СЕРТИФИКАТАМИ НА ПЛАТФОРМЕ ANDROID С УРОВНЕМ API НЕ НИЖЕ 17

Для осуществления конфиденциальности при передаче информации в веб-среде используется протокол Secure Sockets Layer (SSL). Для того, чтобы SSL стал доступен, требуется сертификат. Сертификат - это электронный документ, который связывает данные для проверки подписи с определенным лицом, подтверждает идентичность этого лица и заверяется электронной цифровой подписью Удостоверяющего центра [1]. Для получения сертификата необходимо создать CSR (Certificate Signing Request) - запрос на вашем сервере; найти необходимый SSL сертификат и поставщика; передать CSR запрос и другую,

необходимую для проверки вашего домена и организации, информацию; установить выданный сертификат [2].

Целью данной работы является создание приложения для управления сертификатами. Приложение должно обладать удобным графическим пользовательским интерфейсом и должно быть реализовано на платформе Android с уровнем API не ниже 17.

При генерации CSR необходимо заполнить латинскими символами следующие поля:

1. Имя сервера — полностью определенное доменное имя, например, www.nic.ru»
2. Название страны — двухбуквенный код страны, для РФ — «RU»
3. Область — регион, например, «Moscow»
4. Город или населенный пункт
5. Название организации или ФИО физического лица

Проект состоит всего из двух классов:

1. Класс MainActivity.java (главный экран графического интерфейса)
2. Класс Files.java (реализация в приложении выбора файла пользователем)

На форме, соответствующей классу MainActivity, располагаются 6 текстовых полей с указанием того, как необходимо заполнить поля для ввода и две кнопки: «Генерация» и «Импорт».

При нажатии на кнопку «Генерация», при правильном заполнении всех полей, происходит генерация сертификата с именем, указанным в поле «Название» и расширением *.csr. При этом указывается папка, в которой будет расположен сертификат. После подписания сертификата, располагаем его на SD карте мобильного телефона.

Далее пользователь может импортировать этот сертификат из карты памяти, нажав на кнопку «Импорт». После этого происходит запуск диалогового окна, которое отображает все файлы, находящиеся на SD-карте телефона. Пользователь выбирает необходимый файл, указывает название сертификата и используемый аккаунт. После этого происходит установление сертификата.

Приложение локализовано на русский язык.

ЛИТЕРАТУРА:

1. http://samaratender.ru/text/22
2. http://www.ssl.ua/news/ssl-for-newbs/

УДК 004.588

Д.А. Когутич, М.А.Смирнов (2 курс, фак. Мат-мех, СПбГУ)
Ю.В. Литвинов, ст. преподаватель

ПОДДЕРЖКА КОНСТРУКТОРА EV3 В TRIK STUDIO

Целью данной работы была поддержка конструктора EV3, пришедшего на смену конструктору предыдущего поколения - NXT. EV3 является роботом третьего поколения в серии роботов Lego. С помощью него можно создавать различные конструкции: грузовики, краны, автоматические сортировщики кубиков по цвету, машины и прочее. Фактически это

игрушка для детей от 10 до 17 лет, но отчасти они создавались и для образования, EV3 используют в кружках по робототехнике и некоторых школах.

TRIK Studio — среда обучения основам программирования и кибернетики. В TRIK Studio можно создавать графические программы для Lego® Mindstorms® NXT, TRIK и исполнять эти программы на компьютере, отправляя роботу команды по Bluetooth или USB, которые, в свою очередь, он будет выполнять, а по построенным схемам генерировать код на различных языках программирования и загружать его для исполнения в робота, также можно работать с двумерной моделью роботов.

Целью же данной работы является поддержка работы со звуком, кнопками на роботе, моторами и датчиками, рисованием на экране, реализация возможности дистанционно управлять EV3 и всеми его компонентами (различными датчиками, моторами), подключенными к нему, т.е. взаимодействие с помощью робота с окружающим миром, и реализация 2D модели, которая позволит тестировать поведение EV3 при его отсутствии или невозможности обращения к нему.

В нашем решении соединение с EV3 реализовано двумя способами: через Bluetooth и через USB. Соединения по Bluetooth и USB реализованы с помощью кроссплатформенных библиотек QextSerialPort и libusb, соответственно. В обоих случаях роботу посылается команда, которая может остаться без ответа, например, включить моторы, или команда, которая подразумевает ответ от робота, например, измерить расстояние до объекта. Команда, посылаемая роботу, является массивом бит и его формирование зависит от многих факторов, но, в основном зависит от того к какому объекту мы хотим обратиться (мотор, датчик касания, экран робота и т.п.). По сути, есть два типа устройств, подключаемых к EV3: те, от которых ожидается ответ (подключаются к input портам) и те, которым просто посылается команда (подключаются к output портам), а они выполняют. Моторы относятся ко второму типу, а датчики к первому. Всего существует 6 датчиков. Мы вели работу с датчиками касания, освещённости, цвета и расстояния (датчики температуры и звука применяются достаточно редко). Двумерная модель робота практически идентична реальному, поддерживается работа с датчиками, моторами и звуком, можно рисовать на экране, есть возможность создавать препятствия и следить за поведением робота.

В итоге реализовано соединение с EV3, в визуальный язык было добавлено множество новых блоков для работы с роботом, реализована 2D модель и сейчас идёт работа над режимом генерации, для автономной работы робота.

УДК 004.422

А.Ю. Коровянский (4 курс, каф. СП, СПбГУ),
И.С. Озерных (3 курс, каф. СП, СПбГУ),
А.В. Подкопаев (асп., каф. СП, СПбГУ)

ЯЗЫКОНЕЗАВИСИМОЕ ФОРМАТИРОВАНИЕ ТЕКСТА ПРОГРАММ НА ОСНОВЕ СОПОСТАВЛЕНИЯ С ОБРАЗЦОМ И СИНТАКСИЧЕСКИХ ШАБЛОНОВ

При корпоративной разработке программного обеспечения достаточно часто возникает проблема приведения к единому стилю программного кода, написанного группой людей. Типичная задача, решение которой позволяет устранить эту проблему, называется pretty

printing, а соответствующая реализация - принтером. Обычно принтеры пишутся вручную для каждого рассматриваемого языка. При добавлении нового языка приходится писать новые принтеры с нуля.

Целью данной работы является исследование метода генерации принтеров для различных языков.

В начале разработки у нас была реализация принтера для языка Java на основе комбинаторов, использующий сопоставление с идеальным образцом, а также синтаксические шаблоны. В принтере используется синтаксический анализатор среды разработки IntelliJ IDEA. В качестве входящих данных используются образец «идеального» текста программы и код, который необходимо отформатировать согласно стилю «идеального» образца.

Предлагается разработать генератор принтеров, подобных представленному.

Особенности реализации генератора:

Для генерации языкозависимой части представленного принтера необходимо предоставить описание структур целевого языка. В качестве формата описания структур используется XML. Для работы с этой формой представления данных используется система чтения StAX. Важно обеспечить простую схему XML, чтобы пользователь мог самостоятельно задавать с ее помощью структуры языка, но при этом достаточно гибкую для возможности описания как можно большего множества структур.

```
<component name="For" psiComponentClass="PsiForStatement" createFromText="Statement">
    <subtree name="condition"      psiGetMethod="Condition"
             isCodeBlock="false"    isRequired="false" />
    <subtree name="initialization" psiGetMethod="Initialization"
             isCodeBlock="false"    isRequired="false" />
    <subtree name="update"         psiGetMethod="Update"
             isCodeBlock="false"    isRequired="false" />
    <subtree name="body"           psiGetMethod="Body"
             isCodeBlock="true"     isRequired="false" />
</component>
```

Рис.1 Пример описания конструкции For языка Java.

Генератор минимизирует время, затрачиваемое на формализацию большинства структур, но некоторые специфические структуры, например, оператор Try-Catch языка Java, все также придется описывать вручную. В перспективе предполагается совершенствование генератора для увеличения множества генерируемых им структур.

УДК 681.3, 004.031.4

М.В. Крень (асп., каф. инф., СПбГУ),
В.О. Сафонов, д.т.н., проф.

ВЫБОР АРХИТЕКТУРНОГО РЕШЕНИЯ ПРИ СОЗДАНИИ SAAS-ПРИЛОЖЕНИЯ ДЛЯ ОБЛАКА

Создание облачных *SaaS*-приложений для облака становится с каждым годом популярнее [1]. В первую очередь это происходит потому, что SaaS-приложения обладают рядом преимуществ над обычными приложениями:

1) не требуют покупки и поддержки собственной компьютерной инфраструктуры, для работы программного обеспечения
2) предоставляют повсеместный доступ пользователю, по сетевым каналам, как правило, через веб -- интерфейс
3) за пользование программным обеспечением взимается периодическая плата.
4) решения любого масштаба легко и быстро разворачиваются

Создание таких приложений - непростая задача. Рассмотрим список требований и целей, которые необходимо продумать в ходе разработки.

Существует два принципиально разных архитектурных решения для SaaS-приложений: *многоклиентское* (multi-tenant) и *выделенное* (single-tenant). Выделенная архитектура предоставляет каждому пользователю собственную инфраструктуру -- логические или физические серверы. Такой подход может быть не всегда оправдан, так как подписчику выделяется фиксированное количество ресурсов, независимо от того, сколько их требуется, что может увеличить конечную стоимость подписки для пользователя и сделать ее существенно выше, чем у конкурентов.

Многоклиентская архитектура позволяет приложению изолированно обслуживать нескольких пользователей из разных организаций в рамках одного сервиса. Это свойство разделяемой инфраструктуры, которое позволяет получить максимальную загрузку приложения, что снижает стоимость вычислительных ресурсов и хранилища.

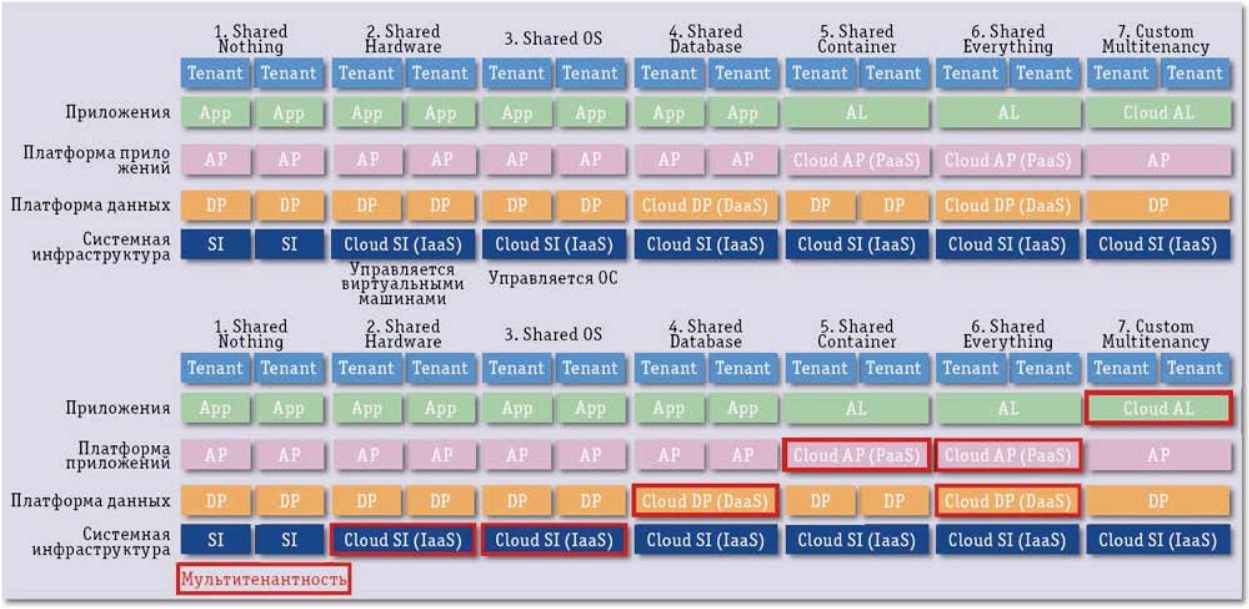


Рис. 1 -- Уровни многоклиентской архитектуры.

Существует огромное разнообразие уровней многоклиентских моделей (рис. 1). Уровень инфраструктуры (System Infrastructure), данных (Data Platform), прикладной платформы (Application Platform) и логики приложения (Application Logic). Стоит изучить множество факторов, прежде чем остановиться на выборе конкретной модели. Рассмотрим некоторые из факторов, оказавших влияние на мой выбор на примере облачной платформы *Microsoft Azure*.

1) Эмулировать выделенную архитектуру на многоклиентской достаточно просто -- это режим “один подписчик”. С другой стороны, обеспечить возможность расширения выделенного приложения до некоторого числа независимых подписок затруднительно.

2) Многоклиентское приложение менее безопасно, чем выделенное. Сбой выделенного приложения влияет только на клиента, который использует данное приложение, в то время как сбой многоклиентского приложения затронет всех клиентов. Можно создать несколько экземпляров такого приложения или его части, что позволит минимизировать негативные последствия при его сбое.

В Microsoft Azure существует возможность развертывания нескольких идентичных копий приложения за счет использования нескольких экземпляров ролей. Microsoft Azure распределяет нагрузку по обработке запросов между текущими экземплярами ролей автоматически, а при возникновении сбоя автоматически перезапускает экземпляры.

3) Использование многоклиентской архитектуры приложения подразумевает возможность различных проверок подлинности и авторизации. Клиенты могут предпочесть использовать как существующую у них систему, так и общую для всех. В Microsoft Azure для настройки проверки подлинности и авторизации используются Access Control Services, с помощью которых можно реализовать аутентификацию формами через различные OpenID-провайдеры.

4) От специфики приложений зависят различные функциональные возможности. Например, обеспечение отдельных баз данных для каждого конкретного подписчика в определенном географическом регионе.

Существуют и другие факторы, оказывающие влияние как на управление жизненным циклом, так и на настройку приложения.

При выборе модели стоит всерьез задуматься о том, как добавлять, изменять и хранить данные, другими словами продумать архитектуру слоя данных. От количества клиентов приложения зависит выбор хранилища и способ его использования, защиты. Подробный анализ скорости обработки данных, методы хранения и защиты информации, и советы по использованию и выбору хранилища в облаке хорошо освещены в статьях [2,3].

Рассмотрим вопрос с финансовой точки зрения. Для выделенного приложения оптимальным решением является создание отдельного аккаунта Windows Azure. Это позволит определять итоговую сумму отдельно для каждого клиента. Некоторых ресурсов, выделяемых в фиксированном размере, на начальном этапе может оказаться более чем достаточно, что сделает приложение неоправдано дорогим. С многоклиентской архитектурой все обстоит иначе. Затраты можно поделить между всеми пользователями, но стоит учитывать, что количество потребляемых ресурсов варьируется от клиента к клиенту. Более того, стоит позаботиться о том, чтобы пользователи могли самостоятельно контролировать свои расходы при использовании приложения. Должны быть разработаны механизмы, позволяющие получать любому пользовотелю доступ к таким данным. Модуль Cloud Ninja Metering Block для Windows Azure позволяет снимать метрики использования ресурсов каждым пользователем, имеет встроенный портал отображения отчетов для клиентов.

При проектировании облачных SaaS приложений перед разработчиком возникает ряд сложных вопросов, ответив на которые, можно выбрать наиболее подходящую архитектуру для конкретной задачи.

ЛИТЕРАТУРА:

1. David Chappel. How Saas changes an ISV's business. A guide for ISV leaders. 2012.
2. Brad Calder, Ju Wang. Windows Azure Storage: A Highly Available Cloud Storage Service with Strong Consistency. Microsoft.
3. Zach Hill, Jie Li, Ming Mao. Early Observations on the Performance of Windows Azure. University of Virginia Charlottesville. 2010.
4. Наталья Ефимцева. Тонкости разработки в облаках. "Открытые системы" . 2012.
5. Мультитенантная архитектура для Saas. Блог Microsoft. 2012.
6. Доминик Беттс, Алекс Хомер. Разработка мультитенантных приложений для облака. 3-е издание.

УДК 004.4'2

М.Е. Стрельцова (студ. 4к., каф. инф., СПбГУ),
А.В. Иванова (студ. 4к., каф. инф., СПбГУ),
О.А. Ямурзина (студ. 4к., каф. инф., СПбГУ).
Д.А. Григорьев (к.ф.-м.н., доц., каф. инф., ММФ, СПБГУ)

АСПЕКТНО-ОРИЕНТИРОВАННАЯ РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ПОМОЩЬЮ СИСТЕМЫ ASPECT.NET

Не секрет, что при разработке или сопровождении различных приложений, необходимо реализовывать ту или иную сквозную функциональность. Это не всегда простые задачи, которые необходимо выполнить в нескольких модулях. Так, например, решение задач ведения журнала системы или обработки исключений в различных точках различных модулей может осложнить восприятие целевого кода и затруднить дальнейшую разработку приложения.

Аспектно-ориентированный подход может решить проблему "запутанной функциональности" путем вынесения этой сквозной функциональности в один модуль, аспект. А система Aspect.NET [1] обеспечит дальнейшую бесшовную интеграцию этих аспектов в целевой проект. Таким образом, наш целевой код останется неизменным, а нужная функциональность будет реализована [2].

В апреле 2013 компания Microsoft выпустила продукт MS Enterprise Library 6 (EL) [3], предназначенный для реализации сквозной функциональности. Библиотека EL представляет собой набор функциональных блоков.

Таким образом, было бы целесообразно внедрить вызовы методов классов, которые предоставляет нам библиотека EL, в наше целевое приложение с помощью Aspect.NET.

Для иллюстрации данного подхода авторами были созданы аспекты, которые обеспечивают журналирование и обработку исключений [4].

Помимо этого, Aspect.NET можно использовать для построения безопасного программного обеспечения. Зачастую архитекторы и разработчики не вспоминают об

обеспечении безопасности проекта на ранних стадиях его разработки. Это приводит к большим трудностям и некорректной работе продукта. Чтобы избежать подобной ситуации, можно использовать систему Aspect.NET и вынести методы, обеспечивающие безопасность, в отдельные аспекты. Как было сказано выше, целевой код останется прежним. Таким образом, нужная функциональность будет добавлена в проект без особых затрат, необходимых для изменения основного кода, и ошибок. Кроме того, использование системы Aspect.NET позволит изменять ее в любое время, например, при изменении прав доступа или способа аутентификации.

В итоге, авторами было предложено несколько аспектов, которые повышают безопасность приложений, а именно: аутентификация с помощью форм, авторизация на основе ролей, шифрование строки запроса.

Ни один продукт на разных этапах разработки не обходится без тестирования. Когда необходимо отследить взаимодействие классов, часто прибегают к использованию таких сущностей, как заглушки, обёртки и мок-объекты. Такая фиктивная реализация и подмена класса очень удобна и помогает избежать «влияния» на используемые данные при тестировании [7]. Например, если тестируемый класс взаимодействует с классом, который в свою очередь выполняет какие-то манипуляции с базой данных - в таком случае, для сохранности реальных данных можно использовать “фейковые” объекты. Чтобы оставить код практически неизменным, для создания такого рода заглушек можно использовать аспекты [8], которые будут эффективно манипулировать методами класса – подменять их, реализовывать некоторые пост- и предусловия для конкретного метода – и средства Aspect.NET отлично для этого походят. В рамках данной работы был реализован класс аспектов, эмулирующий некоторые возможности мок-объектов.

Таким образом, Aspect.NET является довольно удобным и мощным инструментом для добавления необходимой функциональности в целевой код.

ЛИТЕРАТУРА:

1. Safonov V.O. Using aspect-oriented programming for trustworthy software development. – Wiley Interscience. John Wiley & Sons, 2008.
2. Григорьев Д. А., Григорьева А. В., Сафонов В. О. Бесшовная интеграция аспектов в облачные приложения на примере библиотеки Enterprise Library Integration Pack for Windows Azure и Aspect.NET // КОМПЬЮТЕРНЫЕ ИНСТРУМЕНТЫ В ОБРАЗОВАНИИ, 2012. — № 4. — P. 3-15
3. https://msdn.microsoft.com/en-us/library/ff648951.aspx // Доступ 19.03.2015
4. Стрельцова М.Е. Примеры бесшовной интеграции функциональных блоков MS Enterprise Library с использованием Aspect.NET // СПИСОК-2014. Материалы Всероссийской научной конференции по проблемам информатики. Санкт-Петербург, — 2014. — P. 151-156
5. М. МакДональд, А. Фриман, М. Шпушта Microsoft ASP.NET 4 с примерами на C# 2010 для профессионалов // Издательство “Вильямс”, 2011.
6. А. Фримен, С. Сандерсон “ASP.NET MVC3 Framework с примерами на C# для профессионалов” // Издательство “Вильямс”, 2012.
7. A. Hunt, D. Thomas, M. Hargett Pragmatic Unit Testing in C# with Nunit, 2nd Edition // The Pragmatic Bookshelf, 2004.

8. G. Xu, Z. Yang. A Novel Approach to Unit Testing: The Aspect-Oriented Way // http://www.researchgate.net/publication/237602338_A_Novel_Approach_to_Unit_Testing_The_Aspect-Oriented_Way // Доступ 19.03.2015

УДК 004.052.42

И. А. Селин (5 курс, каф. ИУС, СПбПУ), И. В. Никифоров, к. т. н., асс. каф. ИУС,
В.П. Котляров, к.т.н., проф.

ПРЕОБРАЗОВАНИЕ ТЕСТОВЫХ СЦЕНАРИЕВ УРОВНЯ UCM-МОДЕЛИ В ТЕСТОВЫЕ СЦЕНАРИИ УРОВНЯ РЕАЛЬНОЙ СИСТЕМЫ

Одним из наиболее важных этапов разработки программного продукта является проверка корректности его функционирования, т.е. этап функционального тестирования. Практически всегда этот этап стараются автоматизировать, т.к. ручной труд при создании тестов приводит к существенному увеличению затрат временных и материальных ресурсов. Для проектов большого объёма особенно критично минимизировать ручной труд при тестировании.

Рис. 1 Обновлённая технологическая цепочка VRS/TAT

Одним из наиболее мощных решений по автоматизации тестирования является технология VRS/TAT [1], которая позволяет создать формальную модель по исходным требованиям (шаг 1 на рис. 1), провести верификацию системы и сгенерировать тестовые сценарии по формальной модели (шаг 2 на рис. 1). Технология VRS/TAT представляет собой комплекс специализированных инструментов, соединённых вместе в единую цепочку, работающую с требованиями к системе, формализованными на языке UCM [2].

Модель системы на языке UCM находится на более высоком уровне абстракции, чем реальная система, а значит информация о разрабатываемой системе в UCM-модели не полная, т.е. абстрагирована до нужной степени детальности. Соответственно, и тестовые сценарии, генерируемые по ней, недостаточно детализированы для проведения тестирования реальной системы, так как в них используются сигналы уровня UCM.

UCM-сигнал состоит из имени сигнала и списка его параметров. Это линейная структура, которая не содержит в себе вложенностей. Сигнал же уровня реальной системы состоит из имени и параметров, заданных в грамматике сложных структур [3]. То есть каждый параметр может раскрыться как ещё одна сложная структура со своим списком параметров (рис. 2).

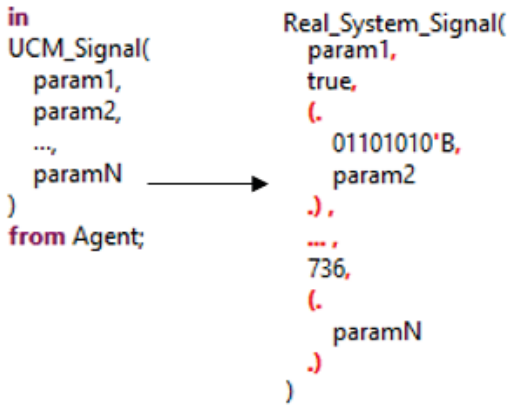


Рис. 2 UCM-сигнал и сигнал уровня реальной системы

Чтобы использовать в тестировании реальной системы сценарии уровня UCM-модели, необходимо уточнить сигналы из них до уровня реальной системы. На текущий момент, эта задача выполняется вручную, что требует большого количества ресурсов и времени. В связи с этим возникает задача автоматизации преобразования тестовых сценариев уровня UCM-модели в тестовые сценарии уровня реальной системы (шаг 3 на рис. 1).

Для решения задачи был разработан инструмент LoweringTool, интегрированный в технологическую цепочку VRS/TAT, который позволяет автоматизировать процесс преобразования UCM-сигналов в сигналы уровня реальной системы.

Первым шагом по преобразованию тестовых сценариев является сбор всех сигналов, использующихся в UCM-модели. Сигналы заданы в метаданных UCM-элементов [4]. После сбора информации пользователю выдаётся сводная статистика о сигналах в UCM-модели с указанием мест их использования.

Следующим шагом является задание соответствия между UCM-сигналом и сигналом уровня реальной системы (нижнего уровня). Так как UCM-сигналы имеют параметры, то в зависимости от их значений должны изменяться и сигналы нижнего уровня. Более того, один UCM-сигнал может быть заменён на несколько сигналов/действий уровня реальной системы. Для обеспечения данной функциональности был создан редактор, позволяющий назначать правила преобразования для UCM-сигналов. Каждое правило состоит из двух частей: условие его срабатывания и список сигналов/действий нижнего уровня, на которые будет заменён исходный сигнал. Данный механизм позволяет гибким образом настраивать подстановку в зависимости от условия срабатывания.

Для упрощения задания правил преобразования сигналов в инструменте реализована поддержка переменных. Используя переменные можно значительно сократить время разработки сигналов, в которых встречаются повторяющиеся части, путём вынесения одинаковых групп значений в переменные. Например, однотипных заголовков сигналов.

После задания правил преобразования, они вместе с тестовыми сценариями, сгенерированными VRS, поступают на вход преобразователя тестовых сценариев (рис. 3).

Преобразователь осуществляет обход тестовых сценариев, заменяя используемые в них сигналы уровня UCM на сигналы уровня реальной системы по указанным правилам, и осуществляя подстановку значений переменных. Результирующие тестовые сценарии являются пригодными для тестирования реальной системы, так как в них уже используются детализированные сигналы.

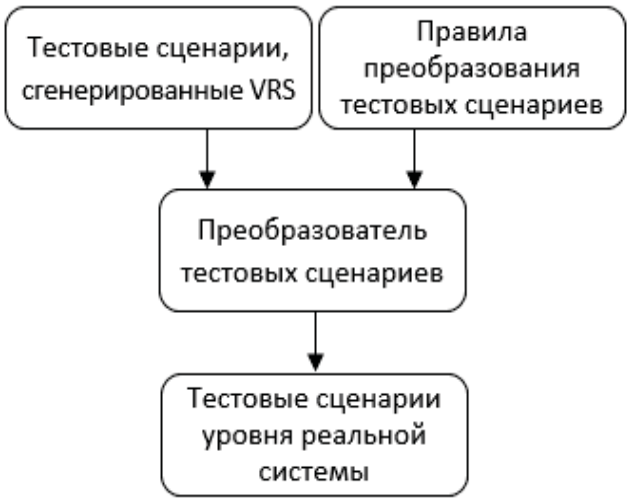


Рис. 3 Схема преобразования тестовых сценариев

Выводы. Существующая технологическая цепочка VRS/TAT была дополнена инструментом LoweringTool, который позволяет преобразовывать тестовые сценарии уровня UCM в тестовые сценарии уровня реальной системы на основе задаваемых пользователем правил. Инструмент позволяет тестировать реальную систему на основе сценариев, созданных по формальной модели, а также снижает трудоёмкость разработки тестов для реальной системы.

ЛИТЕРАТУРА:

1. Baranov S., Kotlyarov V., Letichevsky A., Drobintsev P., Yusupov Yu. Implementation of an Integrated Verification and Testing Technology in Telecommunication Project // IEEE. Russia Northwest Section. International conference “Radio – that connects time. 110 anniversary of radio invention”. Proceedings. Volume II. SPb., 2005. P. 87-92.
2. Buhr R. J. A., Casselman R. S., “Use Case Maps for Object-Oriented Systems.” — Prentice Hall, 1995.
3. ITU-T. Recommendation Z.100. Specification and Description Language (SDL). Technical Report Z-100, International Telecommunication Union, Standardization Sector, Genève, November 1999.
4. Никифоров И. В., Петров А. В., Юсупов Ю. В., Котляров В. П., Применение различных методик формализации для построения верификационных моделей систем по UCM-спецификациям // Научно-технические ведомости СПбГПУ. № 3(126). СПб.: Изд-во Политехнического ун-та, -2011. -С. 180-184

УДК 004.052.42

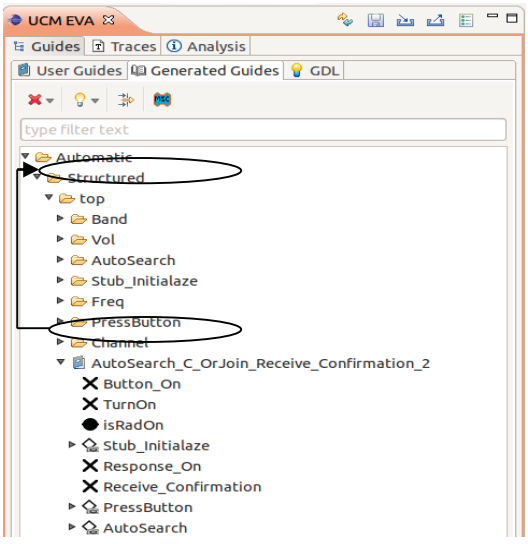
А. П. Маслаков (6 курс, ИУС, СПбПУ), И. В. Никифоров (асст., к.т.н., ИУС, СПбПУ),
В. П. Котляров (проф., ИУС, СПбПУ)

АДАПТАЦИЯ МЕТОДОВ ИНКРЕМЕНТАЛЬНОЙ РАЗРАБОТКИ ГИДОВ ДЛЯ ПРОЕКТОВ ВЫСОКОЙ СЛОЖНОСТИ

На сегодняшний день существует технологическая цепочка VRS/TAT[1] для тестирования и верификации систем на основе моделей. По формальным спецификациям создаётся набор диаграмм на языке UCM[2], описывающих поведение системы. По данной модели в нотации UCM автоматически генерируется модель на языке базовых протоколов (БП)[3] и гиды[4], используемые в трассовом генераторе VRS для получения тестовых сценариев.

Гиды являются набором правил, обеспечивающим заданный критерий покрытия поведения системы и позволяют существенно сократить время генерации сценариев. Но, несмотря на ограничение на генерируемые гиды, в виде конкретного критерия покрытия (критерия ветвей) их количество может достигать нескольких тысяч для систем большого размера. Это приводит к увеличению времени, необходимого на каждую итерацию тестирования, т.к. генерация гидов и, соответствующих им, тестовых сценариев занимает слишком много времени. Для устранения этого недостатка предлагается использовать структурные гиды. Они также автоматически генерируются инструментом UCM2MSC[5]. Структурные гиды могут содержать в себе ссылки - элементы Reference (Рисунок 1), они используются, если на пути обхода диаграммы встречается элемент Stub. Все Stub-элементы привязаны к собственным UCM-диаграммам, каждая из которых описывается собственным набором гидов. Данная структура позволяет анализировать поведение системы на разных уровнях абстракции в соответствии со структурой UCM-проекта.

Рис. 1 - Reference-элемент



Затем по структурным гидам генерируются детальные гиды. Генерация осуществляется в автоматическом режиме, но конкретные пути для неё задаёт пользователь, используя редактор конфигурации экспорта (Рисунок 2). При генерации детальных гидов элемент Reference заменяется одним или несколькими гидами из соответствующего набора. Таким образом, пользователь получает детальные сценарии поведения, учитывающие поведение на нижних уровнях абстракции. Также они позволяют пользователю выбирать конкретные пути

обхода UCM-диаграммы. Такой подход позволил существенно сократить общее число тестовых сценариев.

На сегодняшний день в технологии VRS/TAT присутствует ряд недостатков. Во-первых,

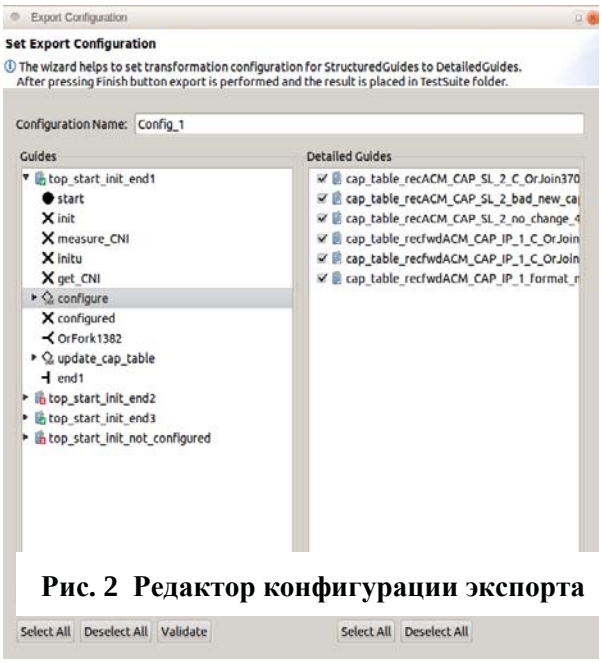


Рис. 2 Редактор конфигурации экспорта

сейчас генерация структурных гидов осуществляется в нотации БП, а затем полученные гиды транслируются в термины UCM-элементов. Во-вторых, на UCM-моделях с числом элементов больше 500, наличием вложенностей диаграмм более чем 10 уровней, и присутствием параллельных конструкций, появляется проблема с недостатком памяти для генерации гидов. В-третьих, в текущей версии пользователь имеет возможность выбирать пути для генерации детальных гидов по структурным, но не может редактировать структурные гиды, полученные в результате автоматической генерации. В-четвертых, конфигурации, созданные для генерации детальных гидов, удаляются из памяти сразу по завершению генерации и пользователь вынужден каждый раз создавать их с нуля. Последним, пятым недостатком является отсутствие отображения сгенерированных гидов на UCM-диаграмму.

Для устранения первого недостатка в работе предлагается избавиться от зависимости от других инструментов, в частности генератора БП UCM2BP[6]. Генерация структурных гидов должна осуществляться сразу в терминах UCM-элементов, чтобы избежать проблем с разными форматами данных и ошибок трансляции модели на языке БП в модель на языке UCM.

Для решения проблемы недостатка памяти при обработке проектов большой сложности (второй недостаток), в работе реализован метод оптимизации затрат памяти в алгоритмах генерации структурных и детальных гидов.

В качестве решения для третьего недостатка, реализован редактор, который будет предоставлять пользователю широкие возможности по изменению автоматически сгенерированных гидов.

Четвёртый недостаток устраняется за счёт сохранения конфигурации экспорта в отдельный *.xml файл. Структура *.xml файла изображена на рисунке 3. Это позволяет выполнять сохранение/загрузку различных конфигураций и реализовать поддержку одновременной работы с несколькими конфигурациями.

Для решения последней проблемы (пятый недостаток) предлагается использовать механизм подсветки UCM-элементов (Рисунок 4), благодаря чему пользователь получил

```
<configuration id="1" name="Config_1">
   <sequence> id="3" name="Secuence"</sequence>
        <elements ucmElemId="26"/>
        <elements ucmElemId="73"/>
        <elements elementId="2" ucmElemId="77"/>
        <referense stubId="51" path="//@Structured/@UCMmap/@Stub51"/>
        <elements ucmElemId="75"/>
        <elements ucmElemId="24"/>
 </configuration>
```

возможность наглядного представления путей, выбранных для генерации.

Рис. 4 - Подсветка

Исправление всех недостатков позволило увеличить размер поддерживаемых технологической цепочкой VRS/TAT проектов, а также упростить и сделать более удобной работу пользователя по созданию эффективного набора тестовых сценариев на основе структурных гидов.

ЛИТЕРАТУРА:

1. Веселов А.О., Котляров В.П. Автоматизация тестирования в области телекоммуникаций // Научно-технические ведомости СПбГПУ. №4(103). СПб.: Изд-во Политехнического ун-та, 2010. С. 180-185.
2. ITU Recommendation Z.151. User Requirement Notation (URN), 2008.
3. Баранов С.Н., Дробинцев П.Д., Летичевский А.А., Котляров В.П. Верификационная технология базовых протоколов для разработки и проектирования программного обеспечения // Программные продукты и системы. 2005. № 1(69). С. 25-28.
4. Воинов Н.В., Котляров В.П. Применение метода эвристик для создания оптимального набора тестовых сценариев // Научно-технические ведомости СПбГПУ. № 4 (103). СПб.: Изд-во Политехнического ун-та. – 2010. – С. 169-174.
5. Ким Р.И. Методика генерации MSC сценариев и эвристик на основе UCM диаграмм // Дисс. на соискание ученой степени магистра. СПб.: СПбГПУ, 2012

6. Дробинцев П.Д., Никифоров И.В., Котляров В.П., Методика проектирования тестов сложных программных комплексов на основе структурированных UCM моделей // Научно-технические ведомости СПбГПУ. № 3(174). СПб.: Изд-во Политехнического ун-та, -2013. -С. 99-105.

УДК 004.4'22

М.В. Тихонова (4 курс, Мат-мех ф-т, каф. системного прогр., СПбГУ)
Ю.В. Литвинов, ст.преп. каф. системного прогр.

ГЕНЕРАЦИЯ КОДА В РЕЖИМЕ МЕТАМОДЕЛИРОВАНИЯ НА ЛЕТУ В СИСТЕМЕ QREAL

Одним из способов ускорения процесса разработки является визуальное моделирование --- способ задавать программу, используя для этого графические объекты. Использование языков общего назначения для визуального моделирования приводит к излишней сложности диаграмм, поэтому часто используются языки, специально созданные для решения какой-либо группы задач. Такие языки называются предметно-ориентированными. Для создания предметно-ориентированных языков используются DSM-платформы, одной из которых является система QReal.

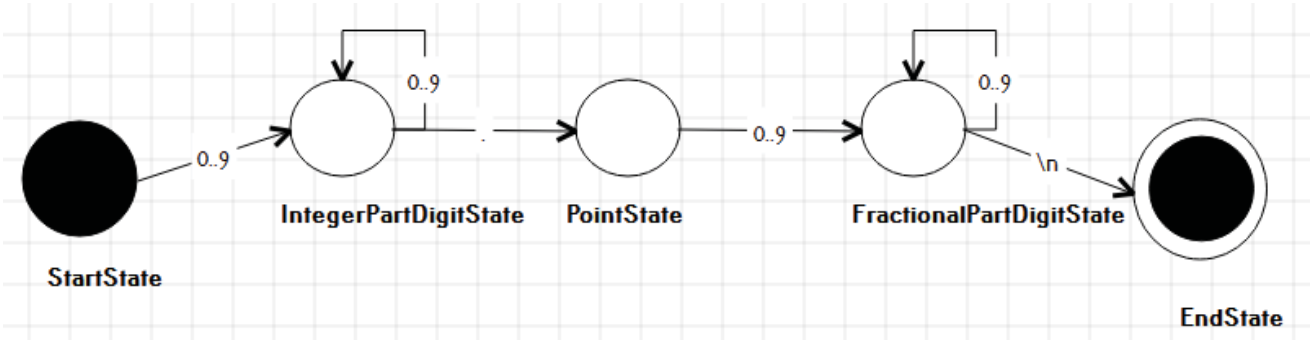
Для создания предметно-ориентированных языков в DSM-платформах используется специальный метаязык. Модель, описанная с его помощью, называется метамоделью. Для быстрого прототипирования языка в системе QReal было реализовано средство метамоделирования на лету, позволяющее непосредственно во время работы над визуальным языком вносить изменения в его метамодель. Пользователи могут не знать про метаязык и метамодель, работая с этим средством, поскольку уровень метамодели скрыт.

Одной из важнейших задач DSM-платформы является получение по диаграмме на визуальном языке его отображения в код на некотором текстовом языке программирования. В системе QReal было реализовано визуальное средство задания правил генерации, но оно работало только для языков, описанных на метаязыке. Для режима метамоделирования на лету тоже бы хотелось иметь возможность получать текстовое представление, чтобы в случае несоответствия ожидаемому результату иметь возможность заметить это сразу и исправить язык. Также, поскольку режим метамоделирования на лету позволяет быстро вносить изменения в язык, добавление возможности получать итоговый код в этом режиме позволяет ускорить процесс разработки. Получается законченное решение, когда прямо на глазах у пользователя создаётся и язык, и генератор, позволяющий получать из диаграмм на этом языке реальную пользу. Поэтому целью данной работы стала реализация текстового средства задания правил генерации для режима метамоделирования на лету.

В ходе работы были проанализированы средства задания правил генерации в различных системах (MetaEdit+, Microsoft Visualization and Modeling SDK, Eclipse, Razor), а также уже реализованные в системе QReal средства задания правил генерации для метаязыка (QReal:Geny и визуальное средство). На основе анализа был разработан язык для задания правил генерации в режиме метамоделирования на лету и реализован синтаксический анализатор и генератор для него. Код, который непосредственно должен попасть в итоговый файл, заключается в кавычки; также присутствуют следующие управляющие конструкции:

конструкция foreach позволяет совершить обход по всем элементам модели или по элементам некоторого типа; конструкции incomingLinks, outcomingLinks и links позволяют получить список всех входящих в элемент связей, исходящих из него связей и связей, имеющих хотя бы один общий конец с данным элементом соответственно; конструкция newline позволяет перейти на новую строку.

Ниже приведена модель конечного автомата и два фрагмента кода: правило для получения списка всех состояний конечного автомата и сгенерированный по нему код.




```
‘enum State {‘
        foreach (currentState in State) {
                currentState.name
                ‘,’
                newline
        }
        StartState.name’,’newline
        EndState.name newline
‘}’
```

```
enum State {
        IntegerPartDigitState
        , PointState
        , FractionalPartDigitState
        , StartState
        , EndState
};
```

УДК 004.4

А.С. Фалько (3 курс, каф ИУС, СПбПУ)
А.О. Веселов (научный сотрудник каф. ИУС ИИТУ, СПбПУ)

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ ПРОЕКТОВ В ОБЛАСТИ ТЕЛЕКОММУНИКАЦИЙ

Данная статья в первую очередь адресована тем, кто уже имеет некий опыт автоматизации тестирования небольших и средних проектов. Все нижеизложенное основано на личном опыте разработки и сопровождении программного продукта, который сочетает в себе методики формальной верификации, тестирования и генерации целевого кода с целью улучшения качества программного обеспечения, а также снижения трудозатрат и себестоимости.

Технология предназначена для автоматизации ручного труда в процессе разработки программного обеспечения. В соответствии с технологической цепочкой, показанной на Рис. 1, разработка программного обеспечения, при использовании данного подхода, состоит из следующих восьми этапов:

1. Формализация требований на основе модели базовых протоколов[2].
2. Проверка модели системы при помощи инструмента VRS[2].
3. Преобразование модели базовых протоколов в SDL [3] модель.
4. Генерация целевого кода из SDL модели.
5. Создание критериев покрытия требований.
6. Генерация тестовых сценариев на основе созданных критериев[1].
7. Генерация исполняемых тестов на целевом языке с использованием инструмента TAT[5].
8. Прогон тестовых наборов и анализ результатов.

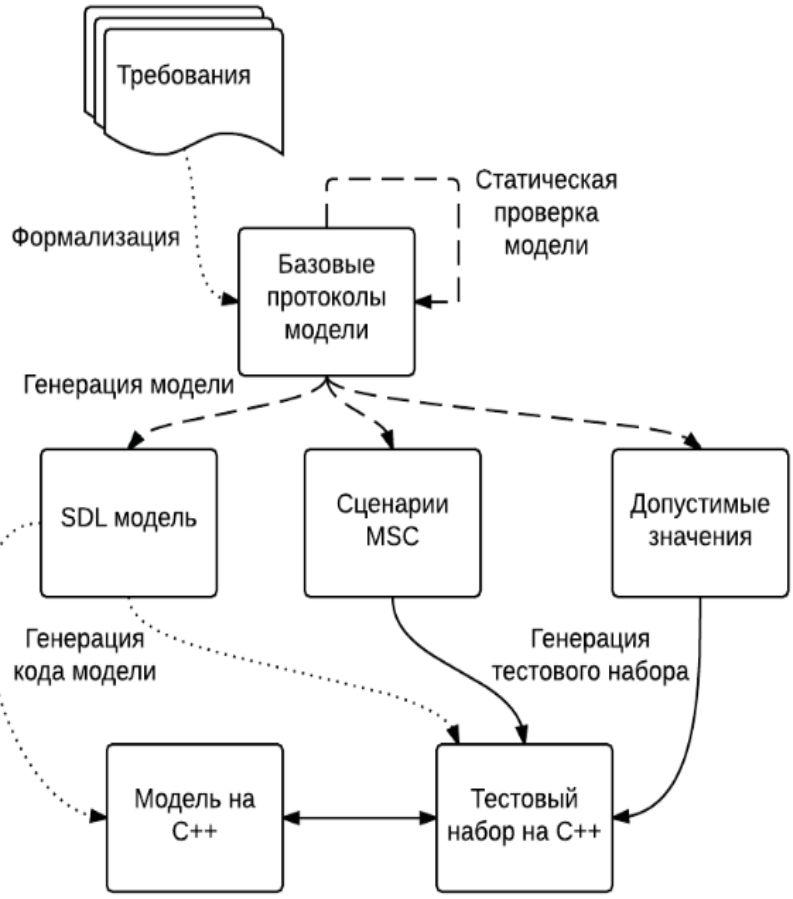


Рис. 1. Технологическая цепочка

Как правило из неограниченного множемтва поведенй системы, инженеры по тестированию выбирают разумное количество сценариев, которые удовлетворяли бы критериям покрытия требований.

Как только тестовый набор выбран в соответствии с критериями, необходимо произвести подстановку символических параметров конкретными значениями из допустимых диапазонов при помощи инструмента VRS[2].

Возможен выбор любых значений, входящих в допустимый диапазон, на практике обычно применяется следующее[4]:

- "Левая граница" - все независимые параметры имеют минимальные возможные значения (в допустимом диапазоне)
- "Правая граница" - все независимые параметры имеют максимально возможные значения (в допустимом диапазоне)
- "Случайные" - все независимые параметры имеют произвольные значения (в допустимом диапазоне)
- "Ошибочные" - по крайней мере, один параметр имеет значение вне диапазона допуска.

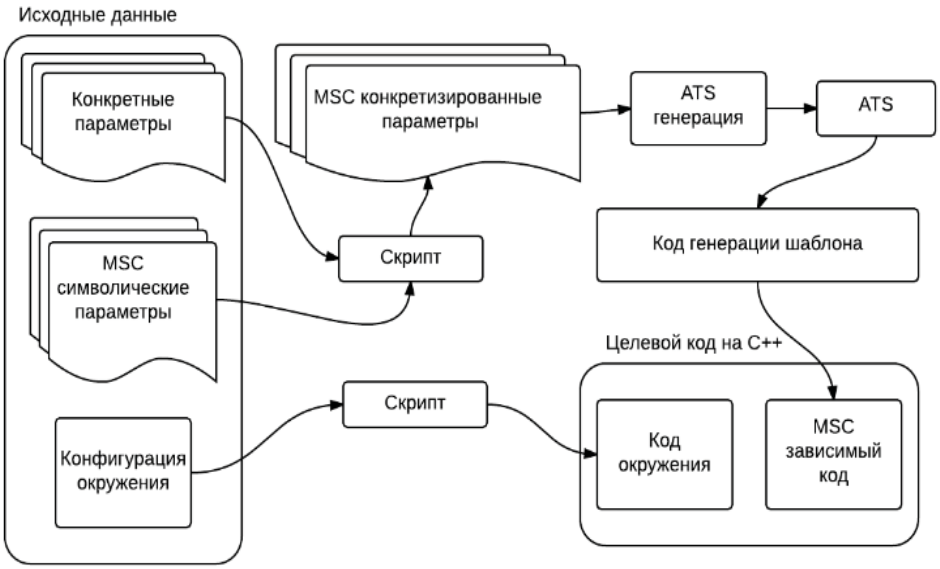


Рис. 2. Генерация теста

Для генерации тестов в целевой язык на базе сценариев, представленных в виде MSC диаграмм и XML конфигурации с описанием окружения, используется TAT[1]. Генерация тестового набора состоит из нескольких основных этапов:

- Анализ конфигурационных файлов и генерация окружения целевого языка
- Генерация абстрактных тестовых наборов (ATS) - представление тестов в виде набора состояний и переходов на языке TCL.
- Генерация тестов в целевой язык на основе ATS.

Эти шаги проиллюстрированы на Рис. 2.

TAT имеет очень гибкий механизм шаблонов генерации кода. С использованием шаблонов, TAT может быть сконфигурирован для генерации тестового кода на произвольном языке[3].

Выводы. Описанный подход автоматизированной генерации тестовых наборов на базе формальных спецификаций помогает значительно сократить время реализации и трудоёмкость создания тестов. Я изучил этот подход на примере: TAT, TAT2, SMscToMsc проектов и убедился в его эффективности.

ЛИТЕРАТУРА:

1. *Н.В.Крупный, Е.П.Назарьев, Б.В.Тютин.* Автоматизация регрессионного тестирования на примере системы TAT. Изд-во Политехн. ун-та, 2014. – С. 163-165

2. *Дробинцев П.Д.* Интегрированная технология обеспечения качества программных продуктов с помощью верификации и тестирования. Канд. дис.,СПбГПУ. 2006. 238 р.

3. *Тютин Б.В.,Котляров В.П.* Масштабирование выполнения тестового набора при автоматизированном тестировании // Научно-технические ведомости СПбГПУ. № 3(174). СПб.- 2013. -С. 118-122.

4. *Тютин Б.В.,Котляров В.П.* Тестирование на основе ключевых слов с использованием диаграмм последовательности событий // Научно-технические ведомости СПбГПУ. № 3(198). СПб.- 2014. -С. 78-84.

5. TAT User's Manual © 2001-2005 МОТОРОЛА

Секция «Программная инженерия: методы и алгоритмы теории программирования»

УДК 004.92

М.В. Братаев (1 курс, каф. ИУС, СПбПУ)

АЛГОРИТМ ОТСЕЧЕНИЯ ПРОСТРАНСТВА В КОМПЬЮТЕРНОЙ ГРАФИКЕ

В последние два десятилетия компьютерная графика нашла широкое применение в нашей жизни. Она используется при создании фильмов, для моделирования различных высокотехнологичных приборов, в военных симуляторах, не говоря уже о такой огромной отрасли, как видеоигры.

С каждым годом сложность 3D сцен растет. Для съемок фильмов оборудуются специальные рендер-фермы, состоящие из десятков и даже сотен рабочих станций. Домашние ПК не обладают достаточной мощностью, чтобы отрисовывать сложные 3D сцены. Поэтому возникает потребность в новых, более оптимизированных алгоритмах их разбиения и отрисовки.

На данный момент одним из самых популярных алгоритмов разбиения является *двоичное разбиение пространства* (**BSP** - *binary space partition*), после которого происходит отсечение невидимого пространства обходом BSP-дерева. Принцип работы алгоритма заключается в том, что все объекты пространства разбиваются на группы, лежащие в разных выпуклых подпространствах относительно некоторой гиперплоскости. Алгоритм рекурсивно повторяется до тех пор, пока каждая группа не будет состоять из одного объекта. Наглядно алгоритм представлен на рисунке:

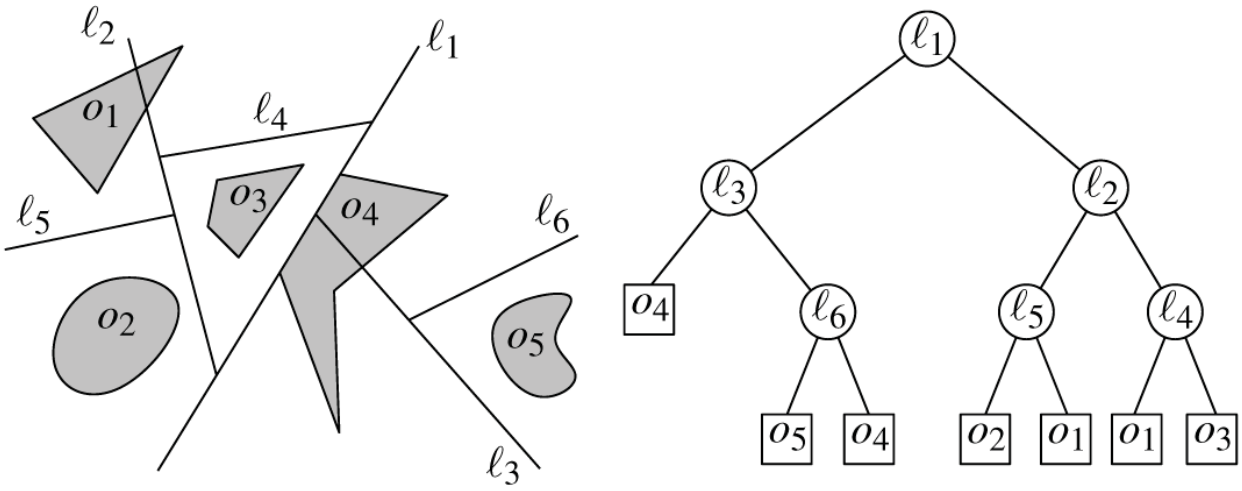


Рис.1 Слева набор объектов на плоскости, а справа соответствующее BSP-дерево

Недостаток данного алгоритма заключается в том, что при разбиении объектов существует вероятность потери части полигонов, которые могут быть не отсечены, либо, что хуже, отсечены, когда этого делать не следует. Также стоит добавить, что задача удаления невидимых объектов для статических сцен хорошо решается с помощью BSP-деревьев, тогда как динамические сцены требуют постоянного обновления BSP, что может занимать существенное время и неприемлемо в реальных приложениях без использования дополнительных алгоритмов.

Я предлагаю следующий метод отсечения пространства: в процессе компиляции исходное пространство разбивается на равные квадратные (или кубические, в трехмерном случае) подпространства до минимального условного размера. Далее от места нахождения камеры в реальном времени во все стороны пускаются лучи, которые следуют до тех пор, пока не пересекутся с объектами. После выполнения алгоритма отрисовываются лишь те участки, где произошли пересечения с объектами. Наглядный пример действия алгоритма можно увидеть на рисунках:

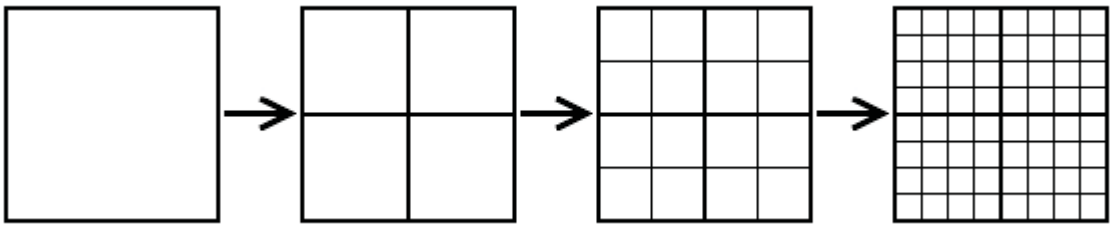


Рис. 2 Разбиение плоскости

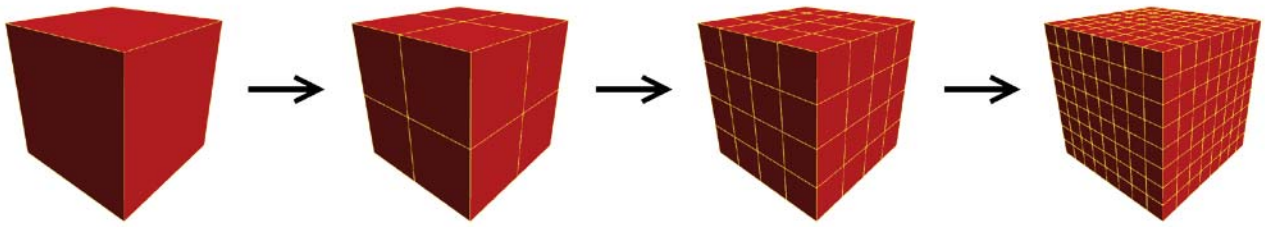


Рис. 3 Разбиение трехмерного пространства

Рассмотрим данный метод на примере рисунка 1. В процессе компиляции пространство разбилось на 80 подпространств. Расположим камеру в центре, например, 58-й секции. Исходящие от камеры лучи пересекутся с объектами в следующих секциях: 49, 59, 69, 66, 56, 46, 47, 37, 38. Всего 9 из 80, что составляет ~ 11% от всей площади пространства. Очевидно, что чем мельче разбиение, тем меньшую площадь поверхности необходимо будет отрисовать.

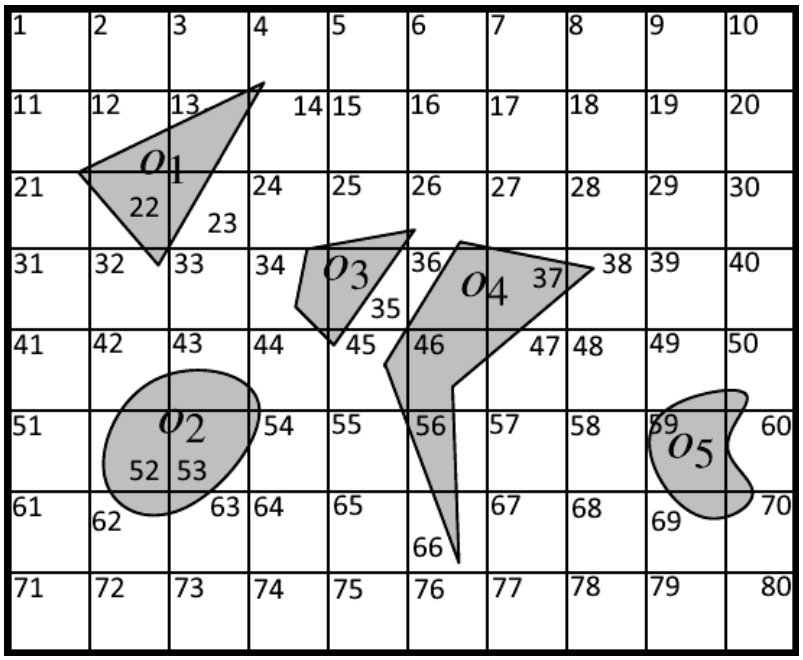


Рис. 4 Пример разбиения двумерного пространства

Достоинством этого алгоритма является тот факт, что с его помощью можно отсекать огромные пространства в реальном времени, затрачивая на это относительно мало оперативной и видео памяти. Таким образом, с помощью данного метода можно работать с большими 3D сценами на компьютерах малых мощностей, например, домашних ПК. Этот алгоритм будет полезен в военных симуляторах авиа и бронетехники, где размеры локаций составляют десятки километров, а так же в индустрии видеоигр, новых поколениях графических движков.

УДК 519.688

В.Г. Бусаров (2 курс, Мат. -мех. ф-т, мат. обеспечение и админ. инф. систем, СПбГУ)
С.Ю. Сартасов, старший преподаватель кафедры системного программирования СПбГУ

АВТОМАТИЧЕСКОЕ СОЗДАНИЕ БОЛЬШИХ ТРЕНИРОВОЧНЫХ БАЗ ДАННЫХ С РАЗНООБРАЗНЫМИ КЛАССАМИ ДЛЯ ОБУЧЕНИЯ БЕЗ УЧИТЕЛЯ НЕЙРОННЫХ СЕТЕЙ ПО РАСПОЗНАВАНИЮ РУКОПИСНЫХ СИМВОЛОВ

На сегодняшний день одной из наиболее актуальных прикладных задач является распознавание рукописных символов, применимое для расшифровки координат отправителя/получателя почтовых перевозок, в военном деле, а также финансовой сфере для оцифровки банковских чеков и выписок. Уже зарекомендовавший себя подход – нейронная сеть.

Выбор сети.

Наилучшим известным на сегодняшний день решением является свёрточная нейросеть с обратным распространением ошибки, предложенная в [1]. В [2] показано, что схожая задача решается нейросетью аналогичной архитектуры, с существенно меньшем коэффициентом

ошибки в сравнении с существующими подходами. Наиболее важным этапом создания нейронной сети является обучение, поскольку оно определяет качество работы системы.

Постановка задачи.

Основа обучения без учителя – тренировочная база. Требуется создать достаточное количество разнородных образцов рукописных символов с метками. Для описания основной идеи рассмотрим только цифры, в последующем обобщим до любых символов.

Обзор существующих решений.

Для наилучшего качества распознавания требуется как можно больший набор разнородных образцов с метками. С учётом объёма данных, эффективное обучение с учителем не представляется возможным. Существует несколько тренировочных баз с уже расставленными метками, такие как NIST, IAM, Cambridge HWI и наиболее популярная MNIST [3]. Однако в подобных решениях имеется множество ошибочных образцов, поскольку они в основном составлялись вручную; такие множества примеров не отличаются разнообразием. Всё это заметно ухудшает качество обучения. Доступно множество неразмеченных данных и некоторые базы (такие как MCYT, OCR) составляются путём распознавания этих изображений другой уже обученной сетью, но она имеет свой, зачастую высокий коэффициент ошибки, что приводит к тем же критичным недостаткам. Часто там собраны данные почерков небольшого (в сравнении с множеством образцов) количества людей, что ухудшает разнообразие.

Предлагаемое решение.

Предлагается автоматически составлять базу с метками, основываясь на множестве шрифтов Microsoft Office Word 2007 (Windows 7). Поскольку мы составляем образцы по уже существующим меткам, то ошибок не будет вовсе. Разнообразие и разнородность примеров строится за счёт 1100 – 1300 шрифтов и восьми настраиваемых многоуровневых параметров: начертание, ширина линий, размер, чёткость (резкость), контраст, различимость символов по отношению друг к другу, цвет. Помимо этого, можно искусственно добавлять некоторый сторонний шум на уже размеченные образцы, используя технологии сходные с CAPTCHA генераторами [4]. Все шрифты хранятся в форматах TTF и OTF, средства декодирования которых доступны.

Выводы.

Данное решение превосходит имеющиеся на сегодняшний день, поскольку исключает всякую возможность ошибки соответствия меток изображениям, гарантирует разнообразие и разнородность тестовых образцов, при достаточном их количестве, генерация проводится автоматически без непосредственного участия человека, решение даёт возможность обучения без учителя. Данный подход ещё не применялся к предложенной задаче, это первая публикация данного решения.

С небольшими изменениями в архитектуре можно обобщить нейросеть на любые символы, включая буквы различных алфавитов. База, построенная данным способом, может использоваться и для других нейронных сетей, вне зависимости от типа. Также, она применима к другим задачам, к примеру, к проблеме полностью автоматизированного символьного CAPTCHA breaker.

ЛИТЕРАТУРА:

[1] *LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel*: Handwritten digit recognition with a back-propagation network // Advances in Neural Information Processing Systems (NIPS 1989), Denver.

[2] *Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton* : ImageNet Classification with Deep Convolutional Neural Networks // NIPS: Neural Information Processing Systems, Lake Tahoe, Nevada, 2012

[3] *Qiao, Yu* : The MNIST database of handwritten digits // ISDEA '13 Proceedings of the 2013 Third International Conference on Intelligent System Design and Engineering Applications, Pg. 1498-1501

[4] *Ved Prakash Singh* : Survey of Different Types of CAPTCHA // (IJCSIT) International Journal of Computer Science and Information Technologies. Vol. 5 (2). 2014. Pg. 2242-2245

УДК 614.461

К.А. Власова (1 курс маг., каф. СП, СПбГУ),
С.Ю.Сартасов, ст. преп.

МОДЕЛИРОВАНИЕ И ПРОГНОЗИРОВАНИЕ РАСПРОСТРАНЕНИЯ ЭПИДЕМИЙ С ИСПОЛЬЗОВАНИЕМ ОБЛАЧНОЙ АРХИТЕКТУРЫ

В данной работе рассматривается построение имитационной модели для противодействия развития эпидемий на примере гриппа в городе Санкт-Петербург, позволяющей выполнять прогноз изменения уровня заболеваемости на основе текущих статистических данных.

Эпидемии — сложные социальные явления, и выбор соответствующей политики для сведения к минимуму их распространения является сложной задачей. Вычислительная эпидемиология использует математические и вычислительные методы для моделирования распространения эпидемий и воздействия медико-санитарных мероприятий для борьбы с распространением.

Особенности реализации модели:

Для построения имитационной модели распространения гриппа выбран агентный подход, как наиболее перспективный и подходящий для дальнейшей разработки системы принятия решений.

Каждый агент обладает своими собственными переменными состояния и поведением, что позволяет почти неограниченно детализировать модель. Важнейшим параметром агента, определяющим его поведение, является его возраст. В соответствии с рекомендациями специалистов НИИ гриппа, было выбрано семь возрастных групп агентов: люди до 2 лет, 3–6 лет, 7–14 лет, 15–24 лет, 25–39 лет, 40–64 лет, 65 лет и старше. Возрастная группа человека характеризует такие важные факторы, определяющие вероятность заболевания, как

интенсивность контактов с другими людьми, его возможные места пребывания (школа, работа и т. д.).

Модель города описывается набором вложенных контейнеров, представляющих собой места скопления людей: метро, наземный транспорт, развлекательные комплексы, магазины, больницы, учебные заведения, предприятия. Контейнеры могут отличаться размером, ёмкостью и плотностью взаимодействия. Люди могут переходить из одного контейнера в другой.

В рамках такого подхода становится возможным оценивать влияние погодных условий, а также эффективность таких мер как иммунизация (изменение характеристик иммунитета у агентов), карантин (исключение контейнеров из всей модели и из модели поведения агентов), вакцинация.

Точность моделирования напрямую зависит от точности описания всего множества рассматриваемых домов и контейнеров. Вычислительная ёмкость многомилионной мультиагентной системы очень существенна, поэтому предлагается эффективная схема распараллеливания с разделением отдельных районов, связанных главными транспортными магистралями, на отдельные машины в облаке.

Программный комплекс позволит построить краткосрочный прогноз развития эпидемической обстановки в городе, а также количественно оценить эффективность тех или иных мер и выбрать наиболее подходящие, исходя из текущего уровня заболеваемости.

Моделирование эпидемиологии позволит внести свой вклад в разработку и анализ эпидемиологических исследований, определить тенденции, сделать общие прогнозы и оценить их неопределенность.

УДК 614.461

Н.В.Волков, студ. кафедры информатики СПбГУ
В.О.Сафонов, д.т.н., профессор кафедры информатики СПбГУ

РАЗРАБОТКА И СОЗДАНИЕ ЭКСПЕРТНОЙ СИСТЕМЫ ПО ВЫБОРУ СМАРТФОНА НА БАЗЕ ТЕХНОЛОГИИ KNOWLEDGE.NET

Экспертные системы являются эффективным решением для задачи поиска необходимого элемента в широкой и сложной предметной области, в которой неподготовленному человеку будет сложно разобраться без помощи эксперта. Для одной из таких проблем как Выбор смартфона предложен прототип системы, созданной на базе технологий Knowledge.NET. Описаны основания для выбора среды разработки, структура системы и актуальность задачи.

В данной работе описывается создание экспертной системы по выбору смартфона на базе технологии Knowledge.NET. В рамках статьи будут приведены определение понятия экспертной системы и краткое описание системы Knowledge.NET, обоснование выбранной темы, постановка задачи. Результатом работы будет описание реализации данной системы.

Экспертные системы. В современном мире среди обилия информации зачастую нелегко найти то, что необходимо, не разбираясь детально в предметной области. Для решения таких проблем принято использовать экспертные системы. Как известно, экспертной системой (ЭС) называют компьютерную программу, позволяющую моделировать рассуждения человека-эксперта в некоторой определенной области. Разработка систем, основанных на знаниях, является составной частью исследований по ИИ и имеет целью создание компьютерных методов решения проблем, обычно требующих привлечения экспертов-специалистов. Экспертные системы предназначены для так называемых неформализованных задач, обычно обладающих следующими особенностями: неоднозначностью, неполнотой и, возможно, противоречивостью данных задачи в пространстве решений с большой размерностью, непрерывно динамически изменяющемся.

Проблема выбора языка реализации экспертной системы. Проблема базовых языков многих экспертных систем – это либо ограниченность возможностей для программирования, т.е. отсутствие сложных структур данных и средств типизации, либо непривычный синтаксис и семантика базового языка (примером могут служить экспертные системы, основанные на

Прологе или языке Лисп). Таким образом, базовый язык экспертной системы должен сочетать в себе черты языка представления знаний и языка программирования. Поэтому для разработки экспертной системы было решено применить систему Knowledge.NET, базирующуюся на языке Knowledge.NET, который является расширением популярного языка C#. Система Knowledge.NET предназначена для разработки и использования библиотек (баз) знаний из разнообразных предметных областей, которые могут проектироваться и реализовываться непосредственно на языке Knowledge.NET, а также могут автоматически извлекаться из сети.

Постановка задачи. Была поставлена задача спроектировать систему для поиска смартфонов по задаваемой или импортируемой базе данных ,наиболее подходящих для пользователя по основным параметрам , и реализовать эту систему при помощи средств Knowledge.NET. В качестве входных данных система будет получать ценовой диапазон моделей, ключевые параметры (операционная система, тип памяти, тип корпуса), а также дополнительные параметры, вводимые по желанию пользователя(вес, процессор, габариты, тип экрана, радио и т.д.) . Результатом работы программы будет список наиболее подходящих по желаемым параметрам моделей.

Актуальность такой системы очень велика, т.к. неподготовленному пользователю (на которого ориентирована данная система) сложно выбрать подходящую модель среди огромного многообразия моделей в интернете и магазинах. Система поможет разобраться без обращения к эксперту даже новичку, при помощи простых и понятных вопросов.

Архитектура разработанной системы. В ходе разработки системы были разработаны база знаний и интерфейс системы. Структура базы знаний, используемой в данной системы, описана frame-классами в экспертном файле. База данных – экземпляр frame-класса base, представляет базу с моделями телефонов, содержит в себе информацию о названии, производителе и цене телефона. Информация о модели телефона разбита на несколько частей и хранится в списках свойств в системе. Элементы, представляющие информацию об одной и той же модели, в различных списках, естественным образом имеют один и тот же идентификатор. Каждая модель ассоциируется с данным идентификатором. Параметры модели описаны экземплярами frame класса Property. Каждый такой экземпляр представляет отдельный тип характеристик.

Интерфейс системы включает в себя набор вопросов и списка параметров (по выбору), на основе которых определяются требуемые характеристики телефона, формируется множество моделей, удовлетворяющих запросу. На завершающем шаге модели из данного множества выводятся в качестве результата.

Заключение. В ходе проделанной работы были достигнуты следующие результаты

1. Произведена классификация моделей телефонов по предложенному набору значимых характеристик и признаков
2. На основе полученной классификации создана база знаний моделей
3. Разработан прототип системы, предоставляющий пользователю возможность получить консультацию по выбору телефона из базы, с указанием предпочтительных характеристик; результатом данной консультации является набор моделей из базы, наиболее удовлетворяющий пользовательскому запросу

ЛИТЕРАТУРА:

1. Общее описание системы Knowledge.NET http://www.knowledge-net.ru/
2. Сафонов В.О и др., "Knowledge.NET ontology-based knowledge management toolkit for Microsoft.NET", .NET Technologies 2006, Плизань, Чешская Республика, ISBN 80-86943-12-7
3. Джозеф Джарртано, Гари Райли. Экспертные системы: принципы разработки и программирование 4-е издание. Пер.с англ. – М. : ООО «И.Д. Вильямс», 2007.

УДК 681.518.001.33

М.С. Гурьев (2 курс, каф. ВТиИТ, СПбГМТУ),
Н.Е. Летов (2 курс, каф. ВТиИТ, СПбГМТУ),
Т.С.Горавнева (к.т.н., доцент, СПбГМТУ)

РАЗРАБОТКА КОМПЛЕКСА ПРОГРАММ РЕШЕНИЯ РЯДА ЗАДАЧ ДЛЯ САПР

В данной работе представлены программы, имеющие практическое значение для конструкторов – пользователей системы автоматизированного проектирования КОМПАС 3D российской фирмы АСКОН. Полученные в процессе выполнения программ результаты расчета подставляются далее в качестве входных данных в команды и операции двумерного проектирования и трехмерного моделирования различных изделий САПР.

В процессе обучения и выполнения курсовой работы по дисциплине «Инженерная и компьютерная графика» перед нами были поставлены задачи исследования построения пространственных кривых и поверхностей в системе КОМПАС 3D.

Пространственные кривые (ломаные и сплайны) могут быть построены по значениям трехмерных координат, которые могут быть введены, помимо задания одиночных величин, также из таблицы, сохраненной в виде текстового файла. Для автоматизированного процесса построения кривых была разработана программа на языке C#, которая позволила вычислить значения координат точек различных аналитически заданных кривых, таких как эллиптический параболоид, однополосный гиперболоид, винтовая линия и т.п.

Для построения поверхностей в системе КОМПАС 3D предусмотрены следующие способы: по точкам (по пласту точек, по сети точек); по пространственным кривым (по сети кривых, линейчатая поверхность); по аналитическим зависимостям, представленным в параметрическом виде (кривая по закону).

Многие команды построения поверхностей могут в качестве исходных данных использовать информацию, представленную файлом трехмерных координат. Поэтому также на языке C# в инструментальной среде Visual Studio 2012 была создана программа, которая в зависимости от различных параметров позволила провести необходимые расчеты и затем построить в системе КОМПАС 3D эллипсоид, тор и другие параметрически заданные поверхности.

Несколько иная задача решалась при исследовании построения и представления двумерных моделей – контуров произвольной формы, состоящих из ломаных линий. Одним из способов внутрикомпьютерного хранения информации двумерного графического изображения является метод цепочного кодирования. Он заключается в том, что на основе кодов направлений и элементарных векторов описывается последовательность кодов заданного контура в виде цепочки чисел. Используя такую цепь в качестве исходной информации, был разработан алгоритм и составлена программа расчета координат узловых и промежуточных точек контура. На основе полученных в результате работы программы

данных, записанных в текстовых файл определенного формата, далее с помощью библиотеки FTDRAW системы КОМПАС 3D построены контуры двумерных графических изображений. Результаты построения подтвердили правильность рассмотренного подхода хранения информации, а также работоспособность программы.

Разработанные программные средства на языке C# в инструментальной среде Visual Studio 2012 функционируют под управлением любой из операционных систем Windows.

УДК 004.75

А. А. Дзюба (асп., каф. инф., СПбГУ),
В.О. Сафонов, д.т.н., проф.

ПОИСК ПОХОЖЕЙ МУЗЫКИ НА ОСНОВЕ ТЕМПОРАЛЬНЫХ КОРРЕЛЯЦИЙ АКТИВНОСТИ ПОЛЬЗОВАТЕЛЕЙ

Размеры фонотек крупнейших музыкальных интернет-сервисов, таких как Spotify и Last.fm, в данный момент превышают 10^7 композиций. Огромное разнообразие музыки не позволяет пользователям ознакомится со всеми потенциально интересными объектами, что вынуждает разработчиков этих сервисов прибегать к методам информационного поиска и *рекомендательным системам*. Эти системы, анализируя историю прослушанной музыки, позволяют найти в базе объектов именно те, которые наиболее вероятно заинтересуют их слушателя. Важным компонентом многих рекомендательных систем являются *похожие предметы*.

Есть два принципиально разных подхода к поиску похожих треков. Один из них базируется на методах информационного поиска в музыке, когда, анализируя аудиодорожку записи, стремятся найти похожие по звучанию. Другой метод использует историю прослушивания или оценок пользователей, предполагая, что похожие музыкальные композиции будут больше слушать друг вместе с другом, нежели слабо похожие. Как правило, трек представляется вектором или множеством из пользователей, которые его прослушали/оценили, а похожесть между такими векторами ищется на основе метрик подобия, таких как коэффициент корреляции Пирсона. Одной из проблем описанного подхода является кубическая сложность расчета относительно N - количества треков в базе.

Если рассматривать историю прослушивания не как множество объектов, а как последовательность, можно не только получить более достоверные корреляции между свойствами объектов, но и упростить их расчет. Предлагается для каждой пары треков i,j считать количество совместных прослушиваний $n_{i,j}$ внутри ограниченного по времени временного окна. В данном исследовании взято окно в 20 минут. Таким образом, каждая сессия прослушиваний порождает набор неупорядоченных пар треков (см. Рис.1), просуммировав количество которых по всем пользователям в системе, мы получим *темпоральные корреляции* музыкальных композиций.

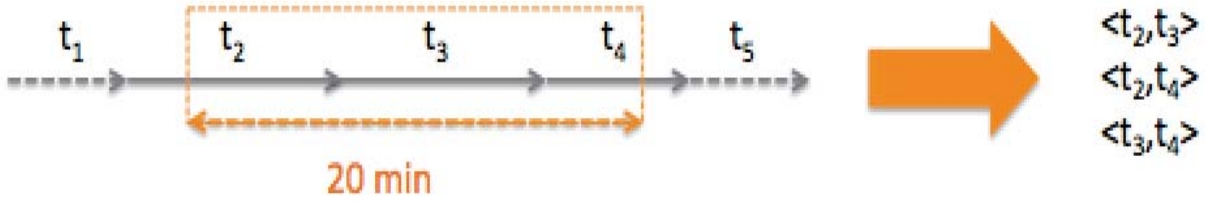


Рис.1 Пары треков, на основе которых вычисляется коэффициент темпоральной корреляции.

В чистом виде количество совместных прослушиваний будет сильно зависеть от общей популярности трека в системе. Для уточнения коэффициента подобия полезно вычесть из ni,j величину, равную нормализованному суммарному количеству прослушиваний трека. С точки зрения вычислений, из матрицы N x N необходимо вычесть матрицу такого же размера. Для того, чтобы величина элементов обеих матриц была взаимно сравнима, матрицы предлагается сделать частично стохастическими, т.е. привести к виду, когда сумма элементов строки равна 1. Все требуемые операции имеют квадратичную сложность относительно N. Помимо меньшей сложности, алгоритм легко распараллелить с помощью MapReduce фреймворка Apache Hadoop.

УДК 004

С.Л. Ключко (4 курс, каф. МОЭВМ, СПБГЭТУ «ЛЭТИ»)
К.В.Кринкин, к.т.н., доцент.

АЛГОРИТМЫ ОРИЕНТАЦИИ КВАДРОКОПТЕРА В ЗАМКНУТОМ ПРОСТАРНСТВЕ

Целью работы является исследование алгоритмов ориентации квадрокоптера в замкнутом пространстве и проектирование программной системы для мониторинга алгоритмов планирования.

Разрабатываемая система способна эмулировать квадрокоптер, а также предоставлять программные интерфейсы для подключения реального квадрокоптера и радиоаппаратуры с помощью технологии Wi-Fi или виртуального COM порта.

Квадрокоптер — это летательный аппарат с четырьмя несущими винтами, вращающимися диагонально в противоположных направлениях. Квадрокоптер не обладает возможностью восстанавливать (без вмешательства пилота) кинематические параметры невозмущенного движения и возвращаться к исходному режиму полёта после прекращения действия возмущений, отсюда вытекает необходимость стабилизации полета.

Стабилизация квадрокоптера происходит благодаря установленным на нем ряда сенсоров, таких как: акселерометр, позволяющий определять угол крена и тангажа; магнитометр, позволяющий определять угол рысканья; гироскоп, позволяющий корректировать показания акселерометра и магнитометра; барометр, позволяющий определять высоту; в некоторых случаях можно оснастить также GPS для полета по маршруту и возврату аппарата в точку взлета. Данные сенсоры позволяют определять положение квадрокоптера в пространстве, а для его стабилизации применяют ПИД – регулятор.

Назначение ПИД-регулятора - в поддержании заданного значения x_0, некоторой величины x с помощью изменения другой величины u. Для своей работы ПИД – регулятор требует настройки трех параметров. Для этого необходимо знать как квадрокоптер будет реагировать на разные воздействия, т.е. необходимо построить математическую модель квадрокоптера.

В самом простом виде формулу ПИД-регулятора можно записать в следующем виде:

$$U(t) = P * e(t) + I * \int_0^t e(t)dt + P * e(t)'$$

, где P, I, D – параметры ПИД – регулятора, t – время с начала работы ПИД - регулятора, $e(t) = (x_0-x)$ – невязка (ошибка регулирования).

Для ориентации в пространстве используют сенсоры, позволяющие определять расстояния до объекта, такие как: ультразвуковой дальномер, инфракрасный дальномер.

Далее необходимо спланировать траекторию перемещения. Алгоритмы планирования траектории можно условно разбить на два подхода, отличающиеся наличием информации об исследуемом пространстве. В первом случае необходимо знать все пространство. Обладая таким знанием можно наложить на пространство сетку и представить её в виде связного графа. Данный подход позволяет построить оптимальную траекторию за короткое время, но справедлив только в том случае, если после сканирования не будет происходить добавление новых препятствий или перемещение уже отсканированных. Во втором случае необходимо работать с пространством состояния, т.е. принимать решение о дальнейшем передвижении на каждом шаге алгоритма. Данный подход является более производительным и не требователен к памяти, чем первый, но пути, найденные таким образом, не являются оптимальными.

В начальный момент времени, когда нет информации об исследуемом пространстве, используем анализ быстрорастущих случайных деревьев. Перемещаясь по данному дереву строим граф пространства. Построение дальнейших путей происходит благодаря эффективному алгоритму поиска на графах A*.

УДК 004.67

О.Д. Елесина (6 курс, каф. ИУС, СПбПУ),
П.Д. Дробинцев, к.т.н., доцент, каф. ИУС СПбПУ
С.Ю. Когинов, ООО «DECOSP».

МОДУЛЬ «ПРОГНОЗИРОВАНИЕ» ДЛЯ ПОДСИСТЕМЫ ПЛАНИРОВАНИЯ ИНКАССАЦИИ БАНКОМАТОВ НА ОСНОВЕ АЛГОРИТМОВ DATA MINING

В настоящее время происходит увеличение количества людей, заинтересованных в надежном и удобном средстве расчёта, таком как платёжная карточка. В связи с этим постоянно происходит увеличение сети банкоматов, следовательно растут затраты на её обслуживание. Таким образом, для обеспечения эффективной работы сети банкоматов необходимо строить прогноз, отражающий день загрузки и сумму загрузки.

Процесс инкассации банкоматов предполагает ряд действий (рис. 1):

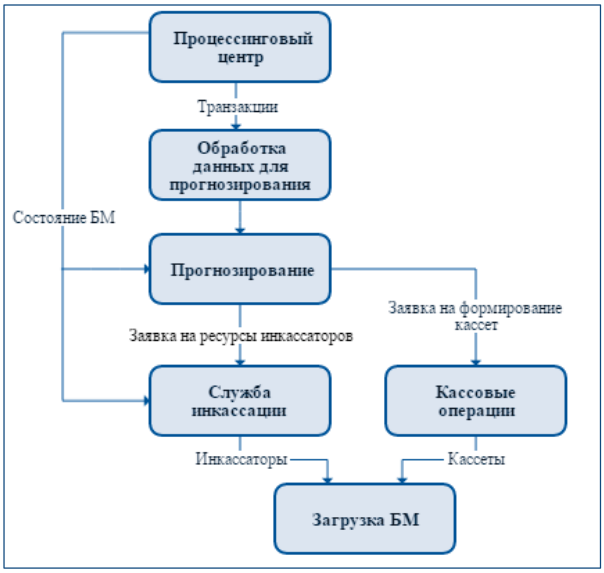


Рис. 1. Схема процесса инкассации банкомата [1]

Существующие на рынке решения для управления инкассации банкоматов имеют закрытые алгоритмы, поэтому сложны в использовании, кроме этого для внедрения необходима их модификация, чтобы обеспечить учёт всех особенностей работы конкретного банка. В связи с этим возникла необходимость создания модуля, как элемента большой системы. Была поставлена цель: разработать модуль построения прогноза для подсистемы планирования инкассации банкоматов.

Для достижения поставленной цели необходимо решить ряд задач:

- проанализировать существующие на рынке программного обеспечения продукты для решения задач планирования (прогнозирования) инкассации банкоматов;
- сформулировать требования к модулю;
- выбрать среду реализации модуля;
- построить модели;
- провести анализ получившихся моделей.

Требования к среде реализации:

- инструменты бизнес - аналитики;
- создание прогнозов;
- безопасность и развертывание;
- средства администрирования.

После анализа рынка был выбран продукт, отвечающий заявленным требованиям – Microsoft SQL Server 2008 R2 , а именно его компонент Analysis Services (службы анализа). В свою очередь Analysis Services позволяет строить многомерные (OLAP – Online Analytical Processing) хранилища данных и Data Mining (интеллектуальный анализ данных) модели для проведения анализа и построения прогнозов. Именно Data Mining будет применяться в данном проекте [2, 3, 4].

SQL Server Data Mining нашёл широкое применение в различных проектах, благодаря обширной функциональности и простоте использования. Кроме того, он обладает удобной средой разработки – Business Intelligence Development Studio, интегрированной в оболочку

Microsoft Visual Studio. Data Mining включает в себя алгоритмы, позволяющие осуществлять прогнозирование данных: Алгоритм дерева принятия решений (Microsoft Decision Trees) и Алгоритм временных рядов (Microsoft Time Series) [5].

В результате работы были созданы две модели на основе представленных алгоритмов, получены прогнозные значения и осуществлён анализ результатов (рис. 2).

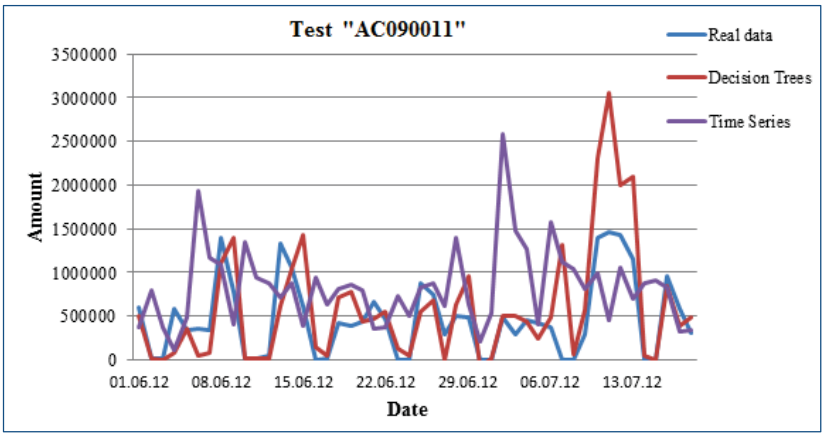


Рис. 2 Сравнение реальных данных с прогнозами для тестового банкомата «AC090011»

Для анализа результатов были выбраны несколько показателей ошибок: средняя абсолютная ошибка (MAE) и коэффициент несоответствия Тейла (вычислялся за тестовый период, показывает степень схожести данных).

Полученные на основе модели с алгоритмом Decision Trees прогнозные значения оказались ближе к реальным данным в большинстве временных отрезках. На данный момент проводится этап разработки пользовательского интерфейса для удобной работы с модулем.

ЛИТЕРАТУРА:

1. Страхарчук А.Я., Страхарчук В.П. Информационные системы и технологии в банках: Учебное пособие. – Украина.: Знание, 2010. 515 с. URL: http://uchebnikionline.ru/bankovskoe-delo/informatsiyni_sistemi_i_tehnologiyi_v_bankah_-_straharchuk_aya/inkasatsiya_bankomativ.htm (дата обращения (09.01.2015).
2. Макленнен Дж., Танг Чж., Криват Б. Microsoft SQL Server 2008: Data Mining – интеллектуальный анализ данных: Пер. с англ. – СПб.: БХВ – Петербург, 2009. 720 с.
3. Лобел Л., Браст Э.Дж., Форте С. Разработка приложений на основе Microsoft SQL Server 2008: Пер. с англ. – М.: Русская редакция; СПб.: БХВ – Петербург,2010. 1024 с.
4. Microsoft SQL Server 2008 R2 - Data Mining. URL: http://www.microsoft.com/sqlserver/ru/ru/solutions-technologies/business-intelligence/data-mining.aspx (дата обращения 15.01.2015).
5. MSDN Library. Алгоритмы интеллектуального анализа данных. URL: https://msdn.microsoft.com/ru-ru/library/ms175595.aspx (дата обращения: 20.02.2015).

УДК 621.321

Жигулин А.Ю. (асп., каф. Информатики, СПбГУ)
Сафонов В.О., д.т.н., проф.

АЛГОРИТМ ЛОЖНОГО ПУТИ ДЛЯ СИСТЕМ ПОИСКА КОНТЕНТА

Целью работы является представление накопленных системой поиска контента знаний в виде структурируемых, ранжируемых и быстро получаемых данных, дальнейшее их получение при нахождении совпадений в базе данных для увеличения шанса успешности поиска интересующего пользователей контента.

При внедрении системы поиска одной из главных проблем является большая вероятность ошибки в представлении добавляемого контента. Это может быть связано с человеческим фактором (непонимание контента при добавлении, опечатки, неправильная информация о контенте), а также с возможностями системы поиска (малое количество параметров контента при добавлении в систему, системные ошибки при добавлении в базу). В результате существует высокая вероятность невозможности нахождения контента в системе по введенному запросу, даже если контент есть в наличии.

Данный алгоритм не уменьшает вероятность ошибки в представлении контента, но позволяет на основании пользовательского взаимодействия с системой предлагать пользователю тот контент, который пользователь, возможно, хотел найти, но не смог из-за нетривиальности поиска.

Особенности реализации системы:

Система разделена на несколько частей:

- интерфейс, определяющий основные возможные действия пользователя, а также ранжированные сущности, определяющие все возможные действия (организация данного интерфейса и сущностей, ему удовлетворяющих, требует 90% всего времени на интегрирование алгоритма ложного пути в систему, и будет уникальна в каждой системе поиска)
- recorder, отвечающий за запись действий пользователя
- receiver, отвечающий за актуализацию путей при успешном завершении алгоритма и удовлетворенности пользователем алгоритмом

Суть работы системы:

Когда пользователь использует систему поиска, система записывает все его действия (выбор фильтров, переходы между страницами контента, комментирование контента, выставление рейтингов) и ищет совпадения с данными, имеющимися в базе данных. Самые актуальные совпадения выводятся в специально отведенной области на странице системы поиска, предлагая пользователю тот контент, который мог бы его заинтересовать, но не отобразился в системе поиска вследствие ошибки представления. В случае выбора такого контента, пользователю предлагается оценить алгоритм, и в зависимости от средней оценки и даты оценивания путь приобретает больший либо меньший вес при следующем выборе данных для предоставления пользователю алгоритмом.

УДК 004.42

А.Ю. Кирсанов (4 курс, каф. СП, СПбГУ),
Я.А. Кириленко ст. преп. Каф СП, СПбГУ.

ИСПОЛЬЗОВАНИЕ .NET/MONO ДЛЯ ПРОГРАММИРОВАНИЯ РОБОТОВ

Основной целью проекта является исследование возможности адаптировать и использовать высокоуровневые средства и существующие промышленные инструменты разработки ПО для программирования роботов и других встраиваемых устройств.

Развитие микроэлектроники и улучшение характеристик контроллеров позволяют использовать для разработки всё более и более современные технологии. Тенденция увеличения уровня абстракции используемых инструментов и языков наблюдается на протяжении десятков лет в промышленном программировании персональных и серверных компьютеров. Во многом повторяя эту эволюцию, мир встраиваемой техники способен не только перенять направление, но и использовать многие существующие продукты.

Среди доступных популярных технологий, созданных для ПК, важно подобрать наиболее подходящие для задач робототехники. В качестве программной платформы выбран .NET Фреймворк, сочетающий в себе:

1. Широкий выбор языков, компилируемых под эту платформу
2. Активное развитие как компонент, так и среды выполнения
3. Наличие свободных реализаций компиляторов и самого рантайма (Mono)
4. Кроссплатформенность на уровне ОС и различных архитектур
5. Технологические преимущества среды выполнения (Сборка мусора, JIT компиляция)

В качестве аппаратной платформы выбран Контроллер ТРИК. Один из процессоров ТРИК находится под управлением специального дистрибутива ARM Linux. Началом работы можно считать перенос Mono в архитектуру контроллера. Использование технологий в новых областях неразрывно связано с анализом производительности, особенно если технологи используются в среде с заметно урезанным количеством ресурсов. Поэтому каждая итерация разработки сопровождалась поиском возможных эвристик и менее ресурсоёмких альтернатив. К данному моменту выработан целый список оптимизаций, позволивший в несколько раз ускорить время запуска и снизить нагрузку на различные компоненты как контроллера, так и самой среды выполнения.

Главным результатом исследований является библиотека Trik-Sharp, через которую пользователь получает доступ к управлению роботом. В библиотеке поддержаны все представленные в ТРИК датчики и устройства такие как: моторы и серводвигатели, светодиодные ленты, датчики расстояния, освещенности, устройства для работы с видео, гироскопом и акселерометром. Функциональность библиотеки, совмещенная со стандартными средствами фреймворка .NET для работы с сетью, помогает решить задачи коммуникации, а средства для вычислений и математики служат хорошей базой для будущих алгоритмов.

Работоспособность библиотеки и самого подхода подтверждена многочисленными примерами, в том числе моделями с высокими требованиями по производительности и времени отклика, например роботом segway, балансирующим на двух колесах. Удобство использования проверено в нескольких студенческих школах, где библиотека использовалась участниками для создания итоговых проектов.

УДК 004.942:004.855.5

П. С. Лапеченков (4 курс, каф. ВТиИТ, СПбГМТУ),
О. Н. Петров, к.т.н., доцент

НЕЙРОСЕТЕВОЕ МОДЕЛИРОВАНИЕ ДИНАМИКИ СУДНА НА ЦИРКУЛЯЦИИ

Повышение надежности и качества принимаемых решений в бортовых интеллектуальных системах является актуальной проблемой при оценке и прогнозе динамики судна на волнении. Особенности оперативного контроля поведения судна в непрерывно изменяющейся динамической среде приводят к необходимости разработки вычислительной технологии, адекватно отображающей сложные процессы взаимодействия в системе «судно – внешняя среда» при различном уровне внешних возмущений. Анализ и интерпретация физических картин динамики взаимодействия судна с внешней средой осуществляются в условиях неопределенности и неполноты исходной информации, что обуславливает применение конкурирующих вычислительных технологий, использующих традиционные методы, нечеткую логику и искусственные нейронные сети (ИНС). Особый практический интерес представляют методы моделирования поведения судна на циркуляции при различном уровне внешних возмущений.

Математическую модель, описывающую пространственное движение судна, можно записать в виде нелинейного матричного уравнения. Общий вид этого уравнения представляется в форме Коши:

$$x' = \Phi(X, U, W, t)$$

где X, U, W – матрицы-столбцы (векторы) переменных состояния, управляющих и возмущающих воздействий; Ф – нелинейная векторная функция, представляющая собой матрицу-столбец скалярных нелинейных функций, общее число которых совпадает с числом переменных состояния. В качестве переменных состояния рассматриваются проекции угловой и линейных скоростей, углы рыскания, крена и дифферента, а также линейные координаты центра масс. Вектор состояния характеризует полное пространственное движение судна. Конкретизация математической модели при исследовании динамики взаимодействия судна с внешней средой связана с переходом от общей модели к системе нелинейных дифференциальных уравнений. Наибольший практический интерес с точки зрения безопасности судна на циркуляции представляет уравнение бортовой качки:

$$(J_x + \mu_{\theta\theta})\theta'' + M_R(\theta') + M(\theta, \varphi, t) = M_G + M_A$$

где $(J_x + \mu_{\theta\theta})\theta''$, $M_R(\theta')$, $M(\theta, \varphi, t)$ – инерционная, демпфирующая и восстанавливающая компоненты, а M_G и M_A – гидродинамический и ветровой кренящие моменты.

Функционирование ИНС в рамках решения задачи нейросетевого моделирования поведения судна в различных условиях эксплуатации обеспечивает возможности нелинейного преобразования информации и массовый параллелизм. При этом достигается высокий уровень параллельной обработки информации за счет использования быстродействующих алгоритмов при «настройке» синоптических связей (параметрический синтез) и организации общей системы параллельных вычислений.

Реализация нейросетевого моделирования осуществляется на языке программирования высокого уровня C# в соответствии с алгоритмом функционирования слоистой сети прямого распространения, в то время как обучение производится методом обратного распространения ошибки с использованием программного средства MATLAB.

Важным инструментом валидации математических и нейросетевых моделей является графическое представление выходных данных, а также анимация процесса моделирования. В качестве наиболее эффективных средств визуального анализа используются временные графики отдельных переменных за период моделирования и графики взаимозависимости.

Таким образом, вычислительная технология нейросетевого моделирования в рамках принципа конкуренции обеспечивает анализ и прогноз поведения судна на циркуляции.

УДК 004.4'2

С.А. Макаров (студ. 4 к., каф. инф., ММФ, СПбГУ),
М.А. Зотов (студ. 4 к., каф. инф., ММФ, СПбГУ),
Д.А. Григорьев (к.ф.-м.н., доц., каф. инф., ММФ, СПБГУ)

РАСШИРЕНИЕ MS VISUAL STUDIO ДЛЯ АСПЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

Аспектно-ориентированное программирование — парадигма программирования, в основе которой лежит идея выделения сквозной функциональности в отдельные модули, которые называются аспектами. Aspect.NET – это инструмент АОП для платформы .NET, разработанный под научным руководством профессора В. О. Сафонова [1].

С помощью Aspect.NET можно определять аспекты в отдельных библиотеках классов, а затем вплетать вызовы их статических методов (называемые действиями) в заданные места целевой сборки (joinpoints). Действия аспектов вплетаются в целевой проект на этапе компиляции. При таком внедрении, аспекты и основная программа сливаются вместе в сборки на уровне байт-кода MSIL-инструкций с минимальными накладными расходами на вызовы действий аспектов. Также поведение результирующего кода становится более предсказуемым, ведь внедренный код зафиксирован еще до момента исполнения программы [2].

Действия можно вставлять перед (%before), после (%after), либо вместо (%instead) вызова указанного в правиле внедрения целевого метода. Также внутри действия аспекта доступен контекст точки выполнения, через который аспект будет менять поведение целевой программы [3].

Для вплетения аспектов в целевой проект с бизнес-логикой, необходимо произвести следующую последовательность действий [4].

1. Выставить post-build events проекта аспектов – указать ссылки на проект бизнес-логики (целевая сборка) и директорию утилиты weaver.exe – подсистемы применения аспектов к целевому коду.
2. Установить зависимость проекта аспектов от проекта бизнес-логики.
3. Собрать проект бизнес-логики.
4. Собрать проект аспектов.
5. Запустить проект бизнес-логики. [5]

Основной целью данной работы является автоматизация этого рутинного процесса, которая достигается с помощью использования средств расширения среды Microsoft Visual

Studio SDK (VS SDK). Было создано расширение, которое добавляет в панель навигации меню, которое, в свою очередь, позволяет выполнять все эти действия с помощью нажатия одной кнопки. Таким образом, программист избавляется от обязанности самостоятельно формировать post-build events и другой однообразной работы. Стоит отметить, что автоматизация формирования post-build events не только ускоряет процесс разработки аспектно-ориентированного приложения, но и позволяет избавиться от ошибок, которые может допустить программист.

В данный момент компоновщик (weaver) написан с помощью инструмента Phoenix Compiler Framework от Microsoft. Этот инструмент используется для получения доступа и изменения MSIL-кода. Развитие MS Phoenix прекратилось в 2008 г. и существует вероятность, что новые версии платформы MS .NET окажутся с ним несовместимы. Поэтому перед авторами также была поставлена задача – переписать код вплетения действий аспекта в точки внедрения с использованием Mono.Cecil.

Mono.Cecil [6] – библиотека, предназначенная для создания, просмотра и изменения программ и библиотек в формате MSIL. Другими словами, используя Cecil, возможно загрузить существующие сборки, просмотреть все хранящиеся там типы, модифицировать их и сохранить на диск измененную сборку. Задачи, которые в Phoenix решаются десятками и сотнями строк кода, могут быть реализованы в Cecil лишь несколькими инструкциями. С использованием Mono авторами был реализован поиск точек внедрения в байт-коде целевой сборки. Получившийся код существенно более компактен, удобочитаем и понятен по сравнению с аналогичным методом, написанным с помощью Phoenix.

ЛИТЕРАТУРА:

1. Safonov V.O. Using aspect-oriented programming for trustworthy software development. – Wiley Interscience. John Wiley & Sons, 2008.
2. Григорьев Д.А. Реализация и практическое применение аспектно-ориентированной среды программирования для Microsoft .NET // Научно-технические ведомости // СПб. Изд-во СПбГПУ. 2009. № 3., - 225 с.
3. Григорьев Д.А., Григорьева А. В., Сафонов В. О. Реализация механизма доступа к динамическому контексту в точках применения аспектов для системы Aspect.NET // СПИСОК-2014. Материалы Всероссийской научной конференции по проблемам информатики. Санкт-Петербург, — 2014. — P. 142-147
4. Григорьев Д. А., Григорьева А. В., Сафонов В. О. Бесшовная интеграция аспектов в облачные приложения на примере библиотеки Enterprise Library Integration Pack for Windows Azure и Aspect.NET // КОМПЬЮТЕРНЫЕ ИНСТРУМЕНТЫ В ОБРАЗОВАНИИ, 2012. — № 4. — P. 3-15
5. Зотов М.А. Реализация надстройки MS VS 2012 для поддержки системы аспектно-ориентированного программирования Aspect.NET // СПИСОК-2014. Материалы Всероссийской научной конференции по проблемам информатики. Санкт-Петербург, — 2014. — С. 117
6. http://www.mono-project.com/docs/tools+libraries/libraries/Mono.Cecil/ // Доступ 18.03.2015

УДК 004.4’414

С. В.Приходько, Д. П.Малютин (2 курс, Мат-Мех СПбГУ),

Ю. В.Литвинов, ст. преп. каф. Системного программирования, СПбГУ

РЕАЛИЗАЦИЯ АЛГОРИТМА ВЫВОДА ТИПОВ ХИНДЛИ-МИЛНЕРА ДЛЯ ЯЗЫКА LUA В TRIK STUDIO

TRIK Studio — среда обучения основам программирования и кибернетики. Среда позволяет создавать графические программы для роботов Lego® Mindstorms® NXT 2.0, Lego® EV3, TRIK и исполнять эти программы прямо на компьютере, посылая команды роботу через Bluetooth или USB-интерфейс, а также генерировать по диаграммам код на различных языках программирования и закачивать его для исполнения в робота.

В частности, в диаграммах могут быть блоки-функции, написанные с использованием некого подмножества языка Lua. Он прост для написания небольших частей кода и позволяет не указывать явно типы переменных. Нужно это, например, потому, что школьникам младших классов, которые пользуются TRIK Studio, тяжело будет объяснить, что такое тип. Язык Lua является динамически типизированным, то есть проверка типов в нём происходит во время исполнения, но для робота может понадобиться преобразовать код на Lua в программу на языке со статической типизацией. Например, для конструктора NXT целевым языком является C. Поэтому возникает задача статической типизации данного языка.

Цель нашей работы состоит в исследовании возможности статической типизации используемого подмножества языка Lua и реализация оной с помощью алгоритма Хиндли-Милнера.

Алгоритм Хиндли-Милнера, построенный в середине 20-го столетия, служит для вывода типов любого заданного выражения некоторого строго типизированного языка, удовлетворяющего определённым критериям, основываясь на рассмотрении того, как это выражение используется. Алгоритм основан на том, что сначала каждому выражению сопоставляется типовая переменная, а затем для каждой такой переменной составляются уравнения на тип (ограничения) согласно определённым правилам, сформулированным Р. Хиндли и Р. Милнером, и затем, на основе этих уравнений с помощью процедуры унификации выводится наиболее общий тип для каждого выражения.

Для написания алгоритма был выбран кроссплатформенный фрэймворк Qt 5 и язык C++, так как они используются в TRIK Studio. Работа производится с абстрактным синтаксическим деревом с помощью паттерна визитор. Предполагается возможность легко расширять правила, добавлять поддержку новых конструкций и типов. В перспективе планируется абстрагировать алгоритм от конкретного языка с целью возможности его переиспользования при добавлении поддержки других языков программирования.

В результате реализован анализатор, позволяющий статически типизировать подмножество языка Lua с помощью алгоритма Хиндли-Милнера, а именно реализован визитор для частных случаев узлов (конструкций языка, таких как арифметические операторы, присваивание и др.), написаны и успешно выполняются тесты для проверки основных случаев программ, где нужна проверка типов.

УДК 004.89

А.М. Савин (5 курс, каф. ИУС, СПбПУ),
П.Д. Дробинцев, к.т.н., доцент,

В.П. Котляров, к.т.н., проф.

МЕТОДЫ И ИНСТРУМЕНТЫ РАБОТЫ С БИЗНЕС-ПРАВИЛАМИ

На протяжении последних десятилетий большинство компаний в процессе своей деятельности так или иначе сталкиваются с необходимостью автоматизации и управления бизнес-процессами, которые обычно включают в себе некоторые наборы правил, определяющие логику поведения в различных ситуациях [7]. Долгое время подобная логика непосредственно закладывалась в создаваемое программное обеспечение, а любые изменения приводили к необходимости переписывания различных участков кода и последующего развёртывания новых версий систем автоматизации.

Со временем стало понятно, что подобный подход является слишком дорогостоящим в силу необходимости привлечения технических специалистов, а также высокой длительности внесения изменений, что особенно актуально для компаний, работающих в банковской и финансовой сферах, для которых чрезвычайно важно максимально быстро реагировать на изменения рынка [3]. Это привело к возникновению нового подхода, основанного на использовании так называемых бизнес-правил (англ. business rules approach).

Основная идея данного подхода заключается в отказе от жёсткого программирования логики в пользу создания эквивалентного ей набора правил, преимущественно выраженного в форме утверждений вида “если <условие> - то <действие>”, которые, в свою очередь, должны быть записаны в максимально простой и понятной нетехническим специалистам форме, например, в графическом виде, в виде таблицы решений или на некотором языке близком к естественному [1]. На основе созданных правил формируется некоторое внешнее хранилище, в котором они могут быть легко изменены, и на которое в процессе своей работы должно опираться программное обеспечение для автоматизации. Таким образом, использование бизнес-правил позволяет максимально быстро вносить изменения в реализованную логику и как следствие добиться более высокой гибкости работы используемого программного обеспечения.

Процесс создания, управления и выполнения бизнес-правил осуществляется с помощью так называемых систем управления бизнес-правилами (англ. business rules management system, BRMS) [1]. На сегодняшний день существует достаточно большое количество подобных систем, однако выбор наиболее подходящей системы может быть весьма непростой задачей в связи с большим количеством различных вариантов. Все они предлагают собственные языки для описания правил и используют различные методы для их исполнения [4]. Более того, до сих пор не существует единого стандарта, описывающего бизнес-правила или системы управления ими [5]. Для того, чтобы облегчить выбор в подобной ситуация, в рамках данной работы предлагается обзор различных систем (IBM WebSphere ILOG JRules, Drools, Bosch Visual Rules, Corticon), методов и стандартов, применяемых в процессе использования данного подхода.

В качестве критериев для сравнения различных систем управления были выбраны такие параметры, как среда создания правил, способы создания правил, способы формирования моделей, отражающих бизнес-понятия, способы интеграции с программным обеспечением, а также методы выполнения правил. По результатам сравнения можно выделить ряд общих рекомендаций по выбору наиболее подходящей системы для различных целей:

- Основным преимуществом JRules является глубокая интеграция с другими продуктами IBM для автоматизации бизнес-процессов, таких как IBM Industry Models и IBM Business Process Manager [1,2]. В результате, если в организации уже используются другие продукты этой компаний, JRules будет наиболее подходящим выбором.
- Система Drools является бесплатной, полностью интегрируется в среду разработки Eclipse и, кроме того, является достаточно простой в использовании, не требуя сложных настроек и подготовительной работы [3], что делает её идеальным выбором для того, чтобы опробовать в действии подход использования бизнес-правил и оценить его целесообразность для организации.
- Visual Rules делает упор на графическое представление данных и правил, а также позволяет выводить последние на основе диаграммы состояний [3], в результате чего является наилучшим выбором в случае, когда требуется обеспечить максимальную модифицируемость бизнес-логики без необходимости привлечения технических специалистов.
- Основным преимуществом Corticon является непрерывный анализ правил в процессе их написания [3], что позволяет добиться высокой скорости выполнения, в результате чего система является наиболее подходящим выбором в случае необходимости работы с действительно большими наборами правил.

В общем случае, использование бизнес правил позволяет повысить контролируемость бизнес-логики, переиспользовать отдельные её компоненты, а также уменьшить зависимости от технических специалистов, что, в свою очередь, позволяет говорить о перспективах развития данного подхода. В частности, в рамках работы рассматриваются несколько подобных сценариев, связанных со слиянием с обработкой бизнес-событий (англ. complex event processing), а также интеграцией в управление решениями (англ. decision management) [6].

ЛИТЕРАТУРА:

1. Бирн, Б., & Аймирзян, Г. (2012). Системы управления бизнес-правилами IBM Industry Models и ILOG: Часть 1. Определение бизнес-правил при помощи моделей IBM Industry Models. Retrieved October 26, 2014, from http://www.ibm.com/developerworks/ru/library/dm-0809byrne/index.html
2. Бирн, Б., Аймирзян, Г., & Уолл, Д. (2012). Системы управления бизнес-правилами Industry Models и ILOG: Часть 2. Добавление системы управления бизнес-правилами. Retrieved October 26, 2014, from http://www.ibm.com/developerworks/ru/library/dm-0810byrne/index.html
3. Вейнберг, Р. Р. (2013). Мировой рынок и критерии оценки выбора и использования систем управления бизнес-правилами. Наука и практика, 2(10), 61-66.
4. Chaware, S., & Karandikar, A. (2014). A review of methods for developing business rules engine. International Journal of Advanced Research in Computer and Communication Engineering, 3(3). Retrieved October 26, 2014, from http://www.ijarcce.com/upload/2014/march/IJARCCE2H%20%20%20S%20sachin%20%20A%20review.pdf
5. Hall, J. (2010, June 22). Business rules standards landscape. Paper presented at the OMG Business Rules Symposium, Minneapolis. Retrieved October 26, 2014, from http://www.omg.org/news/meetings/tc/mn/special-events/br/Hall-BR.pdf

6. Rymer, J. R., & Gualtieri, M. (2011, July 5). The future of business rules platforms: customers are moving to event and decision management. Retrieved October 26, 2014, from ftp://ftp.software.ibm.com/software/websphere/dm/pdf/future_of_business_rules_platforms.pdf
7. Bosch Software Innovations. (2010b). The past, present, and future of business rules [White paper]. Retrieved October 26, 2014, from https://www.bosch-si.com/media/bosch_software_innovations/documents/white_paper/brm/2012-03_Business_Rules_History_EN.pdf

УДК 004.4’22

П.М. Тарасова, Ю.С. Храмышкина (2 курс, СПбГУ),
Ю.В. Литвинов ст.преп.

СРАВНЕНИЕ СПОСОБОВ ПОЛУЧЕНИЯ РЕДАКТОРОВ ПО МЕТАМОДЕЛИ И СКОРОСТИ ИХ РАБОТЫ

Существуют различные подходы к разработке программ и приложений, одним из которых является визуальное программирование. Оно позволяет сделать разработку более простой и наглядной, а также помогает избежать ошибок, возникающих в процессе написания кода. В том числе при командной работе способствует наиболее быстрому достижению взаимопонимания между участниками. Для реализации этого подхода применяются специализированные средства, называемые CASE-системами. Также существуют системы, основанные на технологии предметно-ориентированного моделирования (DSM), в которых можно быстро создавать редакторы своих собственных визуальных языков. Примерами таких систем являются: MetaEdit+, MetaLanguage, QReal.

Данная работа может быть полезна тем, кто занимается визуальным метамоделированием и кого интересуют различные способы и методы разработки редакторов визуальных языков. Здесь визуальное программирования рассматривается на основе среды QReal, в которой существует три различных способа получения редакторов по метамодели: генеративные (с использованием промежуточного xml-представления и без него) и интерпретативный (позволяющий вносить изменения в получившийся редактор «на лету»). Сложно сказать, какой из способов является наиболее предпочтительным. Можно лишь отметить, что если требуется возможность изменять метамодель прямо в процессе создания визуального языка, то наиболее предпочтительным окажется интерпретативный способ, а если требуется наиболее высокая скорость работы, то генеративный будет предпочтительнее.

Целью нашей работы является сравнение визуальных редакторов, полученных различными способами, но по одной метамодели, и выявление различий в скорости их работы, а именно, насколько велики различия при выборе какого-либо из методов по отношению к остальным.

Для этого была реализована утилита, сравнивающая редакторы, полученные каждым из способов, и скорость работы каждого из них. Далее было создано несколько различных метамоделей и были проведены эксперименты с ними. Сравнивались скорости работы двенадцати методов (пример измерений для трёх методов приведён в Таблице 1). В среднем в девяти из них генеративный режим по сравнению с интерпретативным был быстрее приблизительно в 500 раз. Это связано с тем, что в интерпретативном режиме все значения,

которые не требуются постоянно и не являются обязательными для существования метамодели, требуют более длинной цепочки вызовов вспомогательных методов для их получения.

Имя метода	**Генеративный режим**		**Интерпретативный режим**	
	Мат.ожидание (нс)	Среднеквад-е откл-е (нс)	Мат.ожидание (нс)	Среднеквад-е откл-е (нс)
Element description	64000	1642	17700000	1554840
Get property default values	59000	324	38660000	3080100
Is parent of	250000	25582	33000000	7845490

Таблица 1

УДК 004.4

А. В. Чапурина (4 курс, каф. ВТиИТ, СПбГМТУ),
В.А. Семенова-Тян-Шанская, к.т.н., доцент

РАЗРАБОТКА АНАЛИТИЧЕСКОГО ФУНКЦИОНАЛА ДЛЯ СИСТЕМЫ УПРАВЛЕНИЯ РОЗНИЧНОЙ СЕТЬЮ

Цель данной работы – улучшение качества управления, упрощение и автоматизация бизнес-процессов розничной сети путем разработки аналитического функционала. Система управления розничной сетью должна упрощать и в то же время объединять следующие блоки:

- Управление ассортиментом и ценообразованием
- Управление закупочной деятельностью
- Управление складом, распределительным и дистрибьюторским центром
- Анализ деятельности подразделений и розничной сети в целом

Один их самых популярных подходов к решению вышеперечисленных задач связан с использованием технологий *хранилищ данных* - (Data Warehouse), в основе которых лежит многомерная модель данных. Требуемая организация и оптимизация достигается в *системах оперативной аналитической обработки* – OLAP .

В хранилище данных данные OLTP обычно преобразуются и сохраняются в реляционной базе данных, а затем загружаются в многомерную базу данных для анализа. В работе используется ROLAP (Relational OLAP), в этом случаи аналитические запросы строятся над виртуальным многомерным представлением данных, реально их выполнение происходит над реляционной системой, то есть и детальные данные, и агрегаты хранятся в реляционной базе данных. В работе проанализированы звездообразные способы хранения данных. Каждая схема представляет собой различное расположение таблиц фактов и измерений. Проанализированы четыре наиболее часто встречающихся факта (*transaction facts, snapshot facts, line-item facts, event or state facts*). В нашем случае с многоуровневыми измерениями имеет смысл обратиться к расширениям схемы звезды - *схеме* "*снежинки*". В этом случае

отдельные таблицы фактов создаются для возможных сочетаний уровней обобщения различных измерений. Это позволяет добиться лучшей производительности, но часто приводит к избыточности данных.

Microsoft SQL Server 2008 представляет «среды» для облегчения выполнения задач разработки и управления: среда SQL Server Management Studio и среда Business Intelligence Development Studio. В среде Management Studio можно управлять развернутыми решениями служб Analysis Services. В среде BI Development Studio можно разрабатывать решения бизнес-аналитики: проекты служб Analysis Services используются для разработки кубов, измерений и структур интеллектуального анализа данных. Обе эти среды тесно взаимосвязаны с Microsoft Visual Studio, где и разрабатывался представляемый OLAP- функционал.

Данные, хранимые в Analysis Services, могут быть получены выполнением запросов на языке MDX (Multidimensional Expressions).

ЛИТЕРАТУРА:

1. А.А.Барсегян, М.С.Куприянов и др. Методы и модели анализа данных OLAP и Data Mining. БХВ-Петербург, 2004

УДК 004.891.2:629.05

И. А. Янчин (5 курс, каф. ВТиИТ, СПбГМТУ),
О. Н.Петров, к.т.н., доцент

НЕЙРОЭВОЛЮЦИОННЫЙ ПОДХОД К РАСЧЁТУ БЕЗОПАСНОГО МАРШРУТА СУДНА

В последние годы участились случае кораблекрушений по причине ошибок при навигации. Так, в январе 2012 года круизное судно «Costa Concordia», получив пробоину из-за столкновения с рифом, частично затонуло, что привело к гибели нескольких пассажиров и огромному материальному ущербу ввиду невозможности дальнейшей эксплуатации судна. Другим примером является гибель теплохода «Михаил Лермонтов» в феврале 1986 года, который из-за ошибок во время лоцманской проводки в порту ударилось днищем о подводные камни и затонуло. Бортовые навигационные системы, способные строить безопасные маршруты с учётом особенностей окружающей обстановки, а также обнаруживать опасные отклонения от таких маршрутов, способны снизить риски возникновения подобных катастроф.

Целью работы является исследование возможности расчёта оптимального безопасного маршрута движения судна по проливу со сложным рельефом дна с учётом изменяющихся скорости и направления течения, а также уровня воды. При такой постановке задачи рассматривается самый сложный вариант прокладки маршрута, так как разрабатываемая система должна учитывать динамическое изменение окружающей среды. Сложный рельеф дна, в свою очередь, может привести к тому, что судно будет вынуждено активно маневрировать, а в некоторых случаях может потребоваться остановиться и дождаться наступления благоприятных условий. Решение задачи о построении маршрута в реальном времени методами классической математики или полным перебором ввиду необходимости

учитывать рельеф дна в условиях непрерывного изменения динамики внешней среды практически невозможно. Выбор предпочтительной вычислительной технологии в рамках принципа конкуренции определил использование искусственных нейронных сетей для прогнозирования изменений окружающей среды и генетических алгоритмов для построения оптимального маршрута с учётом спрогнозированных изменений.

Разрабатываемая система относится к классу систем поддержки принятия решений. Эти системы обеспечивают оператора множеством допустимых альтернатив управляющих воздействий в случаях, если по тем или иным причинам невозможно использовать автоматическую систему управления. Принцип работы таких систем заключается в построении рейтинга всех возможных решений с последующим выбором того, которое лучше всех удовлетворяет предъявленным критериям.

Искусственные нейронные сети представляют собой систему с топологией ориентированного графа, где в узлах находятся искусственные нейроны, а рёбра графа задают связи между ними. Будучи относительно простыми, искусственные нейроны после объединения в сеть способны решать сложные задачи, приобретая способность распознавать, самообучаться, классифицировать и кластеризовать образы, определять закономерности между входными и выходными значениями и прогнозировать изменение величин.

Эффективным направлением решения задач многокритериальной оптимизации являются эволюционные модели, основанные на использовании генетического алгоритма – имитации биологического естественного отбора путём случайного комбинирования и изменения параметров конкретных решений с целью получения такого решения, которое способно привести к наилучшему (оптимальному) значению целевой функции. Генетические алгоритмы могут быть использованы совместно с нейронными сетями: для их обучения, определения оптимальной топологии сети и в составе гибридных интеллектуальных систем.

Таким образом, разработка систем поддержки принятия решений с использованием интеллектуальных технологий является перспективным направлением. Дальнейшим развитием таких систем могут быть разработка и внедрение систем автоматического судовождения наподобие систем автоматического пилотирования, применяемых в авиации.

Секция «Подходы к разработке программного обеспечения на основе технологий EMC»

УДК 004.75

Е.А. Белокопытова (6 курс, СПБПУ, каф.ИУС),
С.А.Фёдоров, ст. преп. каф. ИУС СПбПУ

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ СИСТЕМЫ ХРАНЕНИЯ И УПРАВЛЕНИЯ ГЕНОМНЫМИ ДАННЫМИ

Стремительное развития области хранения и обработки больших данных позволяет снять ограничение в решении задач из различных технических областей. Например, это касается научных исследований в биоинформатике.

Биоинформатика занимается упорядочиванием и аннотированием геномов и наблюдаемых мутаций, анализом генетической информации. В этой области решаются задачи выравнивания последовательностей, нахождения генов, расшифровки генома, моделирования эволюции, сравнение различных генов с целью выявления рисков и предрасположенностей к заболеваниям, а также многие другие. Решение таких задач сопряжено с хранением, обработкой и управлением большими объемами данных. Для обеспечения этого применяются интенсивные вычислительных методы.

В связи с этим уместно и даже необходимо введение стандартов форматов геномных данных и протоколов доступа к ним, позволяющих множеству пользователей и организаций совместно обрабатывать эти данные на различных платформах. Такой стандарт был разработан сообществом Global Alliance for Genomics and Health (ga4gh). Это свободно доступный стандарт, использующий простой веб-интерфейс для работы и совместного использования данных о ДНК. Описанный API может быть использован в качестве источника данных в программном обеспечении для визуализации (например, геномные обозреватели), для веб-портала с геномными данными, как компонент рабочей последовательности в аналитической деятельности или как основа реализации интерфейса командной строки. Спецификация ставит целью преодоление несовместимости инфраструктур между

организациями и институтами и упрощает так совместную работу, а также поддерживает продвижение исследований генома и их клиническое приложение. В спецификации описаны форматы данных всех объектов предметной области: прочтения (read, read group), сборки геномных последовательностей (reference) , вариации (variant), выравнивания (alignment). На основе таких данных возможно решать большинство задач биоинформатики.

Структура геномных данных описывается следующим образом. Молекулы ДНК – длинные цепочки, состоящие из четырех нуклеотидов, организованные в хромосомы и свёрнутые в ядре клетки. С помощью специальных машин-секвенсоров из молекул ДНК выделяются последовательности описывающих их нуклеотидов, и кодируются в виде последовательностей – секвенций, – составленных из четырёх символов: ACTG (по названиям нуклеотидов). Набор данных, полученных с помощью одного режима работы секвенсора, называется группой прочтений (read group). Группы прочтений объединены в ReadGroupSet. Группы прочтений состоят из миллионов фрагментов — представлений молекул ДНК. При этом сами фрагменты содержат имя, размер и массив прочтений. Прочтения (Reads) представляют из себя непрерывную последовательность оснований, кодируемых символами ACTG, массив качеств оснований и информацию о выравнивании. Выравнивания представляют собой результат обработки прочтений выравнивающим алгоритмом.

Для хранения геномных данных было разработано несколько форматов. Sequence alignment/map format (SAM) – представляет собой формат, в качестве разделителя использующий табуляцию и состоящий из секции-заголовка и секции выравнивания. В заголовке могут быть определены различные метаданные и идентификаторы, такие как: технологии, по которым было произведено секвенирование, идентификатор сборки генома, программы, использованные для обработки read group и т.п.

Целью этой работы является эффективная система хранения и управления геномными данными в стандарте ga4gh. Под её эффективностью понимается:

- возможность хранения больших объемов данных (порядка 1Pb и более);
- защита данных от сбоя отдельных узлов так и от потери всего кластера (репликация данных);
- возможность масштабирования на большое число клиентов;
- время отклика, которое обеспечивало бы комфортную работу клиентов.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Анализ различных БД и выбор лучшего варианта для поставленной задачи.
2. Определение показателей эффективности системы.
3. Анализ и реализация схемы представления модели и оптимизация алгоритмов работы.
4. Измерение характеристик алгоритмов на реальных системах с различным количеством объектов и сравнение с эталоном (измерение нагрузки, отклика и т. д.)

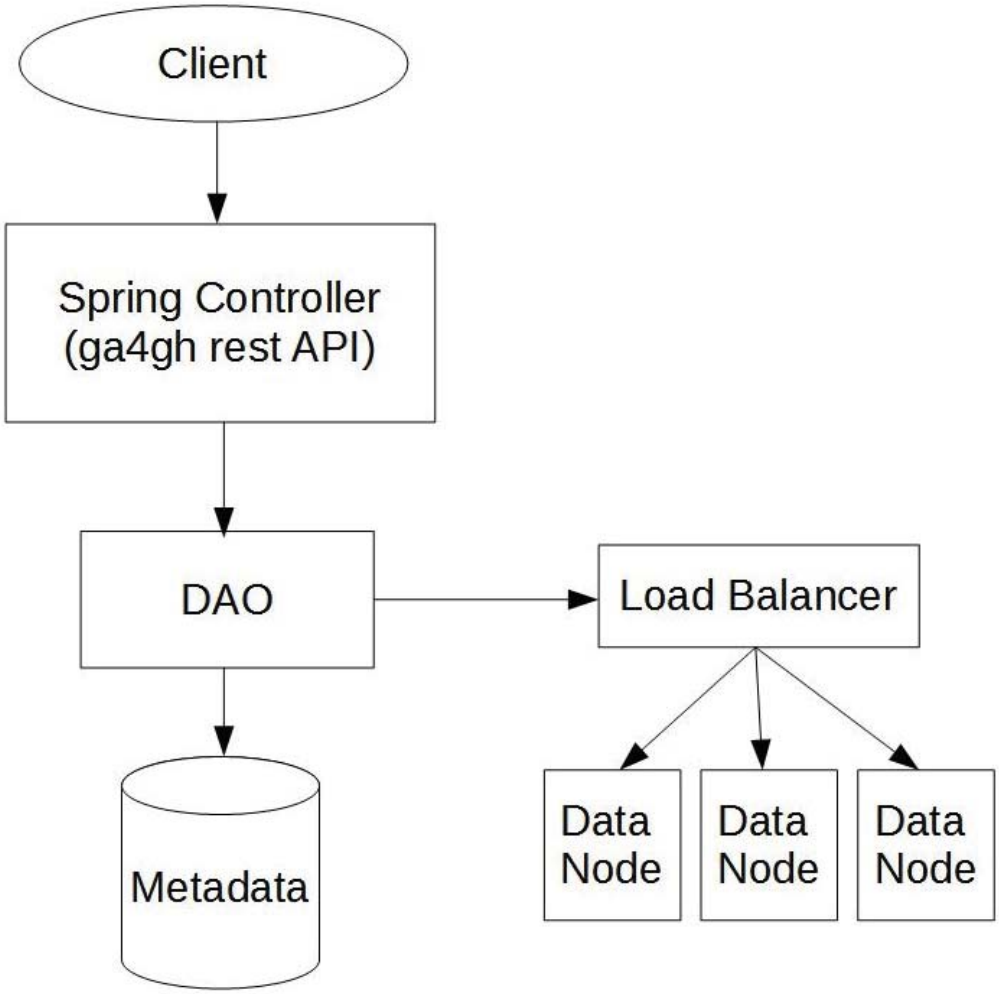


Рис. 1 Архитектура приложения

На рисунке представлена принципиальная схема системы. Клиент посылает REST запросы формата GA4GH, которые принимаются контроллером, который вызывает методы уровня сервиса. В свою очередь, уровень сервиса вызывает методы DAO, получает из базы данных метаданные по запрошенной информации, и запрашивает геномные данные в распределённой файловой системе с балансировщиком нагрузки. В случае использования HDFS в роли управляющей компоненты файловой системы, реализующей необходимый уровень репликации данных, и балансировщика нагрузки выступает NameNode. Из полученной информации выбираются необходимые данные, и отправляются клиенту в упакованном виде в стандартном формате JSON объекта.

Для реализации описанной архитектуры необходимо определиться с выбором её основных компонент: базы данных и распределённой файловой системы. В выборе базы данных для начала необходимо определиться с типом БД: SQL или noSQL. Выбор SQL обладает множеством плюсов классической системы хранения данных, среди которых наличие возможности вводить индексы для ускорения поиска в таблицах, однако необходимо принять во внимание известные проблемы с масштабируемостью и сложность их решения, что может оказаться существенно в случае геномных данных. Второй вариант БД предоставляет возможность легко получать данные древовидной структуры, что в случае с БД SQL

потребовало бы серьёзно продумать схему данных, а также обеспечить сложные и ресурсоемкие запросы с использованием объединений таблиц. Но существует критика не SQL в сязи с тем, что принципы ACID в них не всегда соблюдаются корректно.

В выборе файловой системы необходимо ориентироваться на обеспечение таких свойств, как высокая доступность, способность обеспечения заданного уровня репликации, простота горизонтального масштабирования, желательно отсутствие singe-point-failure. При выборе файловой системы необходимо принять во внимание суммарный объём данных (1Pb и более), а также размеры и количество SAM- и BAM-файлов, с которыми предстоит работать. Необходимо обеспечить возможность быстрого поиска файлов, и получения их необходимых частей.

УДК 004.65

И.И. Закирова (5 курс, каф. ИУС, СПбПУ),
П.Д. Дробинцев, к.т.н., доцент.

ПОСТРОЕНИЕ КЛАССИФИКАЦИЙ ПРИЛОЖЕНИЙ НА ОСНОВЕ ТРЕБОВАНИЙ К УРОВНЮ ОБСЛУЖИВАНИЯ

В современном мире информация является одним из самых ценных ресурсов. Объемы информационных потоков постоянно растут, а вместе с ними растет и сложность систем хранения данных(СХД). С увеличением сложности систем хранения возникает целый спектр новых задач. Одной из основных задач является подбор конфигурации Информационного Центра (ИЦ), удовлетворяющей требованиям заказчика. Требования к системе могут быть формализованы в виде набора характеристик приложений, которые должны быть запущены на этой системе. В общепринятом формате каждое приложение можно описать в терминах объема требуемой памяти, нагрузки на вычислительные ресурсы, интенсивности запросов к памяти, а также их типов.

Несмотря на то, что минимально допустимая конфигурация ИЦ в теории может обеспечить работу приложений, на практике велика вероятность сбоев в системе. Сбой в СХД может стать причиной временной недоступности данных, их потери или же остановки всей системы в целом, а как известно информация полезна только в том случае, если ее возможно доставить туда, где она необходима[1]. Последствиями недоступности данных, так называемому времени простоя, является потеря производительности всей организации, доходов и снижение репутации. Учитывая данные риски, заказчик помимо описанных характеристик задает требования к уровню обслуживания приложений (SLO[2]). На Рис.1 представлена схема, отражающая структуру входных данных, которые подаются в систему построения модели ИЦ.

Рис. 1 Структура входных требований

Требования к уровню обслуживания определяются не для всей системы в целом, а для каждого приложения по отдельности. Это связано с тем, что в рамках одного ИЦ могут работать, как важные приложения, потеря данных в которых приводит к значительным последствиям, так и служебные – простой в работе которых, а также потеря части данных не будет критичной.

В данной работе для задания требований к уровню обслуживания были выделены следующие характеристики: RTO, RPO, уровень резервирования, допустимое количество элементов, которые могу отказать и среднее время отклика. Введем определения.

RPO (Recovery Point Objective, Директивный срок восстановления) - это момент времени, к которому система должна быть восстановлена после простоя[3]. Также это понятие характеризует тот объем данных, который предприятие может позволить себе потерять: чем больше RPO, тем выше устойчивость бизнеса заказчика к информационным потерям.

RTO (Recovery Time Objective, Директивное время восстановления) - промежуток времени, в который система, приложения и функции восстанавливаются после сбоя[3]. Этот параметр определяет время, которое организация может позволить себя находиться в состоянии простоя и при этом избежать в дальнейшем серьезных последствий.

Параметр уровень резервирования (Reservation Level) определяет на каком уровне будет проходить добавление избыточного оборудования. Дополнительные устройства могут быть добавлены, как в сам сервер, в виде дополнительных дисков или ядер, так в стойку, в виде дополнительных серверов. Кроме того, можно создавать удаленные репликации защищающие приложения от глобальных территориальных сбоев. Таким образом, в ИЦ могут быть добавлены стойки с идентичным оборудованием, это позволит в нормальном режиме распределить нагрузку, а в случае масштабного сбоя полностью мигрировать приложения на удаленную часть ИЦ.

Характеристики RackHA и DataCenterHA определяют, сколько устройств может отказать, чтобы система продолжала бесперебойную работу. Отличаются они типом устройств, в первом случае речь идет о серверах, соединениях и коммутаторах, которые находятся в одной стойке. В случае DataCenterHA рассматриваются такие устройства как rack(стойка) и магистральные коммутаторы.

Время отклика (ART, Average response time) - это суммарное время запроса на операцию ввода/вывода, то есть общее время от прибытия заявки до отправки из системы. Среднее время отклика зависит как от времени обслуживания, времени затрачиваемого на обработку запроса, так и от загрузки системы. Несмотря на то, что основным критерием работоспособности системы является доступность информации, время отклика может быть настолько большим, что работу системы нельзя будет назвать приемлемой.

Требование пользователя к уровню обслуживания приложения имеет широкое влияние на результирующую конфигурацию ИЦ. Во-первых, как было описано выше, это отражается на объеме дополнительного оборудования, во-вторых, определяет структуру связей между элементами[4].

Для учета требований к уровню обслуживания в расчетах конфигурации ИЦ необходимо выразить это влияние более точно, по возможности в числах и правилах. В ходе сбора требований к системе расчета конфигурации ИЦ было принято решение использовать систему коэффициентов, обеспечивающих заданный пользователем уровень SLO. После расчета минимально необходимого оборудования, его следует умножить на корректирующий коэффициент. Кроме этого указание количества элементов, которые могу отказать в системе, напрямую определяет некоторые ограничения, а именно минимальное число элементов в системе и минимальное количество избыточного оборудования. На последние характеристики влияют и другие параметры, входящие в требования к уровню обслуживания, однако в меньшей степени.

В результате прогнозирования трудоемкости расчетов, было выявлено, что учет требований к уровню обслуживания каждого приложения в значительной степени усложняет расчет и построение ИЦ. В следствии этого, в данной работе предлагается рассматривать в рамках построения системы не требования каждого приложения в отдельности, а некоторые группы приложений, тиры, которые содержат приложения со схожими требованиями к уровню обслуживания. Это позволит не только снизить трудоемкость расчетов, но и проектировать ИЦ, в которых предусмотрено расширение, так как при добавлении нового приложения нужно будет внести только локальные изменения. Дополнительное оборудование будет добавлено к тому тиру, который удовлетворяет требованиям нового приложения по уровню обслуживания. Архитектура тира единообразна, поэтому специального анализа для нового приложения, можно не проводить.

Данное упрощение может быть применено после того, как приложения разбиты на группы. Эта задача не имеет тривиального решения, поскольку разбиение на группы должно быть осуществлено не по одному параметру, а относительно набора характеристик. Также нельзя забывать, что результирующие требования к обслуживанию группы должны покрывать требования всех приложений, входящих в эту группу. В данной работе было предложен следующий алгоритм.

1. Сначала для каждого приложения определяется взвешенная сумма параметров требования к уровню обслуживания
   $\sum_{i=1}^{n} \alpha_i \, p_i$, где $n -$ *число параметров в требованиях,* $\alpha_1 .. \alpha_n -$ *весовые коэффициенты для каждого параметра,*
   $p_1 .. p_n -$ *параметры требований к обслуживанию одного приложения*
   На этом проходе определяются требования с минимальной и максимальной взвешенной суммой, они в дальнейшем будут использованы для формирования групп.

2. Осуществляется второй проход по всем приложениям. Для каждого из них определяется разница между его требованиями к уровню обслуживания и требованиями, определяющими группы.
   $\Delta SLO = \sum_{i=1}^{n} \alpha_i \, |p_i - \overline{p_i}|$, где
   $\overline{p_1}..\overline{p_n}$ – *параметры группоопределяющего требования к обслуживанию*
   Приложение помещается в группу, где разница ΔSLO была меньше всего. При этом пересчитываются параметры требований, определяющих группы
   $\overline{p_i} = \max(p_i, \overline{p_i})$, i=1.. n, (функция мах в данном случае подразумевает выбор наиболее строгого требования к системе)

Чтобы получить большее количество групп можно на первом шаге создавать больше количество требований, вычисляя промежуточные значения между максимальным и минимальным. Либо после распределения всех приложений сделать дополнительный проход и добавить к требованиям того приложения, которое имеет наибольшую дельту (ΔSLO) с требованием к тиру, в которых это приложение находится. Далее повторить шаг 2.

Поле	RPO	RTO	Reservation Level	RackHA	DataCenter HA	ART
Тип, значения	CONTINIOUS	SECOND	List	int	int	int
	SECOND	MINUTE	SERVER			
	MINUTE	HOUR	CLUSTER			
	HOUR	DAY	DATA_CENTER			
	DAY		GEO			

Рис. 2 Структура SLO в приложении DC Sizer

Данная методика была применены в приложении DC Sizer, которое выполняет автоматический подбор конфигурации ИЦ, согласно требованиям заказчика. Применение комплекса методов по классификации приложений на базе требований к уровню обслуживания, позволило упростить расчеты модели ИЦ и его конфигурацию, позволяя при этом проводить достаточно гибкий анализ требований и не добавлять в систему слишком много избыточного оборудования.

ЛИТЕРАТУРА:

1. П.Пильцнер. Безграничное богатство // Новая постиндустриальная волна на Западе. М.: Academia, 1999. 415 с.
2. G.Crump. Backup Basics: What do SLO, RPO, RTO, VRO and GRO Mean. [Электронный ресурс] URL: http://storageswiss.com/2014/01/22/ (Дата обращения 18.03.2015)
3. От хранения данных к управлению информацией. / EMC – СПб.: Питер, 2010. 544 с.
4. Ю.А.Семенов. Сетевая надежность. [Электронный ресурс] URL: http://citforum.ru/nets/semenov/4/45/ (Дата обращения 19.03.2015)
5. И.Г.Черноруцкий. Методы принятия решений. - Спб.: БХВ-Петербург, 2005. 416 с.

УДК 004.42

С.В. Морозов (4 курс, каф. Системного Программирования, СПбГУ),
В.М. Нестеров, д. ф.-м. н., проф.

РАЗРАБОТКА ВЕРИФИКАТОРА ДЛЯ S3-СОВМЕСТИМЫХ ОБЛАЧНЫХ ОБЪЕКТНЫХ ХРАНИЛИЩ ДАННЫХ

Современные облачные хранилища гарантируют высокую доступность данных и устойчивость к разделению системы на части. Но приходится ослаблять требования к согласованности данных, в силу невозможности гарантировать выполнение доступности, устойчивости к разделению и согласованности в распределенной системе, такой как облако, в соответствие с теоремой CAP. Поэтому, к примеру, наиболее известное облачное объектное хранилище Amazon Simple Storage Service использует ослабленную модель согласованности данных – «согласованность в конечном счете».

Тем не менее у пользователя может возникнуть вопрос: соответствует ли сервис облачного хранилища установленной в документации модели согласованности и выполняется ли вообще базовое требование к любому хранилищу – целостность данных? Одним из протоколов, позволяющим проверить выполнение требований целостности и указанного уровня согласованности для облачных хранилищ, является протокол VENUS.

Целью моей работы является анализ и реализация протокола VENUS с целью применения созданного программного обеспечения при тестировании S3-совместимых облачных хранилищ на соответствие заявленным уровням целостности и согласованности данных. Важной задачей является безболезненная интеграция реализованного слоя программного обеспечения в существующую инфраструктуру общения клиентских приложений с S3-совместимым облачным хранилищем.

Верификация данных осуществляется путем взаимодействия клиентов с т. н. верификатором – небольшим сервером, хранящим метаинформацию об осуществляемых клиентами операциях, а также путем взаимодействия клиентов между собой. Важным достоинством презентуемой реализации протокола является отсутствие необходимости «учить» существующие приложения исполнять протокол VENUS, достаточно выставить в настройках приложения адрес прокси-сервера. Прокси-сервер перехватывает HTTP запрос от приложения и вызывает на исполнение протокол VENUS, который и осуществляет правильное взаимодействие клиентов между собой, верификатором и облачной системой хранения. Это открывает большие возможности для использования представленной реализации в связке с несколькими генераторами потока запросов по протоколу S3 в целях тестирования облачного хранилища. Безусловно, это программное обеспечение будет полезно и при обыкновенной работе пользователей с общим набором данных.

Реализация протокола VENUS была выполнена на языке Java. Java, Go и Python являются языками, чаще всего используемыми при разработке облачных приложений. Язык Java был выбран как наиболее зрелый язык, являющийся корпоративным стандартом. Имеется огромное количество библиотек, написанных на Java, позволяющих не «изобретать велосипед», как это часто приходится делать в еще развивающемся языке Go.

За основу прокси-сервера был взят LittleProxy – высокопроизводительный HTTP прокси-сервер, написанный на Java, и предоставляющий хороший API для расширения функциональности. При разработке прототипа я старался пользоваться стандартными средствами Java. Использование большого количества сторонних библиотек только усложнит разработку проекта такого масштаба. Тем не менее для логирования была выбрана библиотека SLF4J, так как дает выбор разработчику какой системой логирования он будет пользоваться.

Я пользовался системой LogBack, так как она дает показатели производительности лучше, чем стандартное средство Java JUL или библиотека log4j. Для тестирования реализации протокола был написан генератор нагрузки на хранилище данных по протоколу S3 с использованием Amazon S3 SDK.

УДК 004.021

Д.И. Осенняя (5 курс, каф. ИУС, СПбПУ),
П.Д. Дробинцев, к.т.н., доцент.

ПОДХОД К РЕШЕНИЮ ЗАДАЧИ О РАСПРЕДЕЛЕНИИ ПРИЛОЖЕНИЙ ПО СЕРВЕРАМ В ИНФОРМАЦИОННОМ ЦЕНТРЕ

Информация играет огромную роль в современном мире. Объём информации, производимой компаниями и частными лицами, растёт с экспоненциальной скоростью. Значимость информации в мире бизнеса растёт стремительными темпами вместе с её объёмом. Успех в бизнесе зависит от быстрого и надёжного доступа к соответствующей информации. Некоторые коммерческие приложения по обработке информации включают системы бронирования авиабилетов это такие системы как, система рассылки счетов за использование телефона, системы электронной коммерции, использования банкоматов, проектировки изделий, управления материально-техническим снабжением, архивами электронной почты, веб-порталами, базами данных по клиентам, кредитными картами, науками и мировым рынком ценных бумаг и т.д.

Рост значимости информации для бизнеса усилил потребность в защите и управлении данными. Информацию необходимо эффективно хранить, защищать, оптимизировать и управлять ею.

Организации содержат информационные центры для централизованной обработки информации всего предприятия. Информационные центры хранят огромное количество данных, необходимых для выполнения различных задач, и управляют этими данными. Крупные организации часто содержат несколько информационных центров для распределения нагрузки по обработке данных и обеспечения резервного копирования в аварийной ситуации.

При этом каждой организации требуется определённая конфигурация информационного центра, которая отвечает её запросам, удовлетворяет специфике поставленной задачи и обеспечивает необходимую защиту данных её приложений. Поэтому процесс подбора конфигурации каждой системы трудоёмок и требует индивидуального подхода [1].

Для того, чтобы упростить процесс подбора конфигураций информационных центров, разрабатывается программный продукт, который позволит производить его автоматически. На вход системы должны подаваться требования приложений по вычислительной мощности, размеру оперативной памяти, размеру дисковой памяти, скорости, уровню защищённости от сбоев. А на выходе должно генерироваться несколько вариантов моделей информационного центра, подходящего под входные требования. Из этих моделей заказчик может выбрать одну, и она будет являться спецификацией его будущей системы.

Решения, генерирующиеся программным продуктом, должны иметь инфраструктуру hyper-convergence. Hyper-convergence — это тип инфраструктуры системы с архитектурой,

базирующейся на программном обеспечении, в которой вычислительные ресурсы, ресурсы хранения, виртуализации и другие ресурсы тесно связаны между собой в наборе оборудования одного производителя [3]. То есть, информационный центр будет состоять не из специализированных устройств, предназначенных для вычислений или для хранения данных, а из обычных серверов, которые будут являться как вычислительными устройствами, так и хранилищами данных.

При разработке данного программного продукта возникла следующая подзадача: задача распределения приложений по серверам таким образом, чтобы серверов требовалось как можно меньше, но при этом требования каждого приложения были удовлетворены (под приложением в данном случае понимается некий процесс, который не может быть распределён по нескольким серверам). То есть, чтобы каждый из серверов был по максимуму заполнен по следующим параметрам:

1. возможное количество процессоров
2. возможное количество дисков
3. возможное количество RAM
4. возможное количество портов

Данная задача напоминает задачу о рюкзаке с несколькими ограничениями, которая формулируется следующим образом [2]:

$$f(X) = \sum_{j=1}^{n} c_j x_j \to max$$

$$\sum_{j=1}^{n} w_{kj} x_j \le W_k$$

$$x_j = \{0,1\}; \quad j = \overline{1,n}; \quad k = \overline{1,m}; \quad c_j > 0; \quad 0 \le w_{kj} \le W_k$$

- Имеется n предметов, каждый из которых имеет веса по m параметрам (w_{kj}– вес предмета j по параметру k) и стоимость c_j. Вместимость рюкзака ограничена величинами W_k по каждому критерию. x_j показывает, взят предмет j в рюкзак (1), или нет (0).
- Требуется наполнить рюкзак предметами таким образом, чтобы сумма стоимостей выбранных предметов была максимальна, но при этом не была превышена его вместимость ни по одному параметру.

Задача распределения приложений по серверам была формализована следующим образом: приложение рассматривается как предмет, сервер — как рюкзак, а параметрами рюкзака являются: количество процессоров, количество дисков, количество RAM, количество портов.

$$f(X) = \sum_{j=1}^{n} c_{kj} x_j \to C_k$$

$$\sum_{j=1}^{n} w_{kj} x_j \le W_{kr}$$

$$x_j \in \{0,1\}; \quad k = \overline{1,m}; \quad c_{kj} = w_{kj}; \quad C_k = W_k; \quad r = \overline{0,i}, \quad i \to min$$

где j – номер предмета, n – количество предметов, k – номер параметра, c_{kj} – стоимость предмета j по параметру k, w_{kj} – вес предмета j по параметру k, W_k – вместимость рюкзака по параметру k, C_k – максимальная стоимость рюкзака по параметру k, r – номер рюкзака, i – количество рюкзаков.

Видно, что решаемая задача имеет следующие отличия от классической задачи о рюкзаке с несколькими ограничениями:

- Вес предмета по каждому параметру является и его стоимостью по этому параметру.
- Нет некоторой общей стоимости предмета, сумму которых надо максимизировать. Вместо этого, необходимо, чтобы стоимости по каждому параметру стремились к некоторому ограничению (которое равно ограничению по весу по данному параметру).
- Количество рюкзаков не ограничено, но требуется, чтобы их потребовалось как можно меньше.

Из-за этих отличий не удаётся применить к рассматриваемой задаче алгоритмы для решения классической задачи о рюкзаке.

Предлагается следующий подход к решению поставленной задачи:

На первом шаге алгоритма в рюкзак кладётся первый по порядковому номеру предмет. А на шаге *y* предмет будет выбираться таким образом, чтобы разницы между степенями текущей утилизации рюкзака по всем критериям были минимальными, то есть, чтобы рюкзак заполнялся как можно более равномерно по всем критериям. То есть выбираем тот предмет, для которого следующее значение — минимально:

$$\min_f \left(\max \left(\frac{w_{ky}}{W_k} - \frac{w_{hy}}{W_h} \right) \right), \quad k = \overline{1,4}, \quad h = \overline{1,4}, \quad k \neq h$$

$$w_{ky} = w_f + \sum_{j=1}^{y-1} w_{kj}$$

$$y = 2 \ldots$$

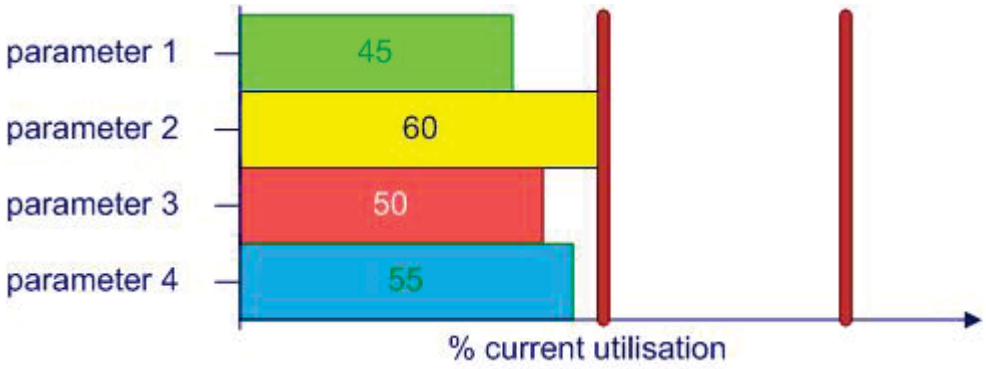
где f – предмет-кандидат на выбор, y – номер шага алгоритма

Таким образом, не должно возникнуть ситуации, при которой рюкзак полностью заполнен по одному параметру, а по другим остаётся ещё много свободного места.

После заполнения текущего рюкзака переходим к заполнению следующего, и так должно продолжаться, пока не будут размещены все предметы.

Для того чтобы не приходилось на каждом шаге перебирать все оставшиеся предметы, можно в начале алгоритма разделить их на m! групп, в каждой из которых содержатся предметы с одинаковым порядком утилизаций $u_{kj}=w_{kj}/W_k$ по разным параметрам. Тогда, если

на очередном шаге имеются, например, такие текущие утилизации рюкзака по разным параметрам:



то искомый предмет должен иметь порядок утилизаций u2, u4, u3, u1 и находиться в соответствующей группе.

Для того чтобы можно было быстро находить нужную группу, можно построить дерево такого вида:

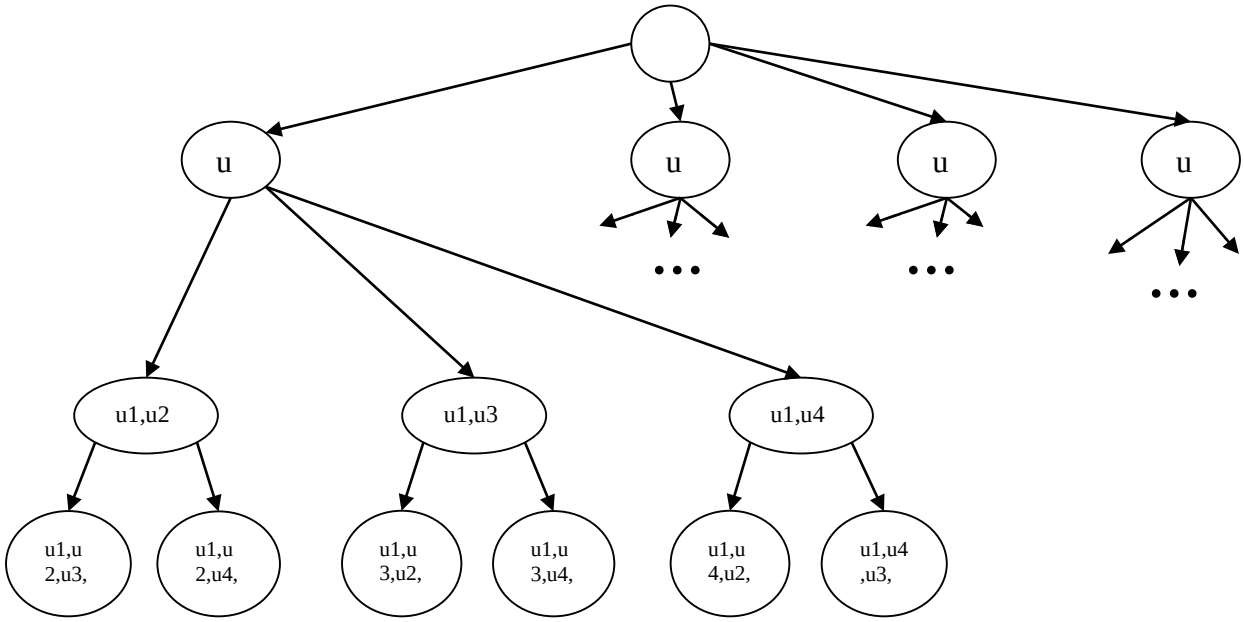


Рис.1 Дерево поиска групп

После нахождения нужной группы остаётся перебрать только предметы, находящиеся в ней.

С помощью предложенного алгоритма можно в первом приближении решить поставленную задачу. Учитывая, что требования приложений к ресурсам сервера являются оценками, а не точными значениями, можно считать этот результат приемлемым.

В дальнейшем, планируется либо дорабатывать описанный алгоритм, либо попытаться привести задачу к какой-либо модификации задачи о рюкзаке, и решать её одним из существующих алгоритмов.

ЛИТЕРАТУРА:

1. От хранения данных к управлению информацией / EMC – СПб.: Питер, 2010. 544 с.

2. Р. Р. Кагиров. Многомерная задача о рюкзаке: новые методы решения. – Вестник Сибирского государственного аэрокосмического университета им. академика М.Ф. Решетнева, №3, 2007. с. 16-20.

3. What is hyper-convergence ? - Definition from WhatIs.com: [Электронный ресурс]. URL: http://searchvirtualstorage.techtarget.com/definition/hyper-convergence (Дата обращения: 19.03.2015)

УДК 681.3

Л. А. Скороспелов (3 курс, каф. МО ЭВМ, СПбГЭТУ),
И. А. Гречко (4 курс, каф. ВТ, СПбГЭТУ),
К.В. Кринкин К.В., к.т.н, доц. каф. МиИТ СПбАУ РАН.

ИССЛЕДОВАНИЕ ОРГАНИЗАЦИИ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ

В процессе роста сложности сетей и процесса конфигурирования её составляющих, появились программно-конфигурируемые сети. Данный вид сетей отличается тем, что уровень управления данными отделён от уровня передачи, и на уровне управления происходит настройка сетевых устройств. Такое отделение даёт возможность полностью виртуализировать сеть, предоставляя возможности настройки сетевого оборудования в целом напрямую. Для такого конфигурирования часто используется протокол Openflow, где с помощью специальной программы (контроллера) все устройства, которые поддерживают указанный способ настройки, получают их правила маршрутизации.

Openflow - стандартизованный протокол управления процессом обработки данных, реализующий концепцию SDN. Коммутатор OpenFlow может быть чем угодно – от простого коммутатора до сетевого экрана и маршрутизатора, но он должен удовлетворять следующим требованиям:

- иметь таблицу потоков (flow table);
- иметь безопасный канал (secure channel), для реализации соединения с контроллером;
- поддерживать протокол OpenFlow.

Существуют различные реализации OpenFlow коммутаторов. Их можно разделить на программные и аппаратные, OpenFlow-only и OpenFlow-hybrid коммутаторы.

В качестве объекта исследования используется OpenFlow коммутатор Open vSwitch. Open vSwitch - многоуровневый виртуальный гибридный коммутатор, разрабатываемый под лицензией Apache 2. Open vSwitch поддерживает OpenFlow protocol 1.0, но последняя версия Open vSwitch 2.3 поддерживает протокол OpenFlow вплоть до версии 1.3.

В процессе выполнения исследования технологии организации программно-конфигурируемых сетей использовалась реализация виртуального свича Open vSwitch: с

помощью виртуальных машин была настроена сеть, обмен данными между которым происходил через единый шлюз с установленной копией Open vSwitch. На отдельной машине была запущена одна из реализаций контроллера Beacon, куда и обращался шлюз для конфигурации таблиц маршрутизации внутри запущенного виртуального свича.

В рамках данного проекта дальнейшей целью является исследовать Open vSwitch на предмет накладных расходов при прохождении через него трафика по протоколу iSCSI. Замеряемые характеристики производительности:

- пропускная способность (количество запросов, обрабатываемых коммутатором в секунду - потоков/секунда)
- задержка коммутатора (время, затрачиваемое на обработку запроса)

Результаты данного исследования предполагается использовать при проектировании и реализации систем хранения данных, где планируется использование протокола iSCSI.

УДК 004.4'24

А. Д. Янкович (5 курс, ИУС, СПбПУ),
И. В. Никифоров (ассистент, к.т.н., ИУС, СПбПУ)

СБОРКА МНОГОМОДУЛЬНОГО RCP-ПРИЛОЖЕНИЯ НА ПЛАТФОРМЕ OSGI

Разработка ПО - трудоемкий процесс, который состоит из большого числа этапов. Одним из этапов является сборка конечного программного продукта. Сборка представляет собой процесс преобразования файлов с исходным кодом и их компиляции в артефакты, составляющие приложение, такие как бинарные и исполняемые файлы.

На сегодняшний день разработано множество инструментов для сборки различных программных продуктов. В данной работе будут исследованы возможности и проблемы сборки многомодульных RCP-приложений на платформе OSGi[1], так как применение спецификаций данной платформы является крайне широким: от разработки встроенных систем до построения многофункциональных декстоп-приложений и Enterprise-систем. Исследуемое приложение представляет собой графический редактор бизнес-процессов, реализованный на языке Java в среде разработки Eclipse с помощью набора модулей - Eclipse RCP[2], реализованных на основе спецификаций OSGi.

Целью работы является выявление наилучшей комбинации инструмента для сборки выбранного приложения, а также способа сборки, обеспечивающего наилучшие результаты по производительности.

Основными инструментами для сборки приложений на языке Java являются: Ant[3] и Maven[4] от компании Apache. Как правило эти инструменты используются для разных целей и в том или ином случае разработчики отдают предпочтение одному из них. Ant управляет сборкой с помощью специального xml-файла (build-файл), состоящего из набора целей (Targets), которые содержат вызовы заданий (Tasks). Между целями могут быть определены некоторые зависимости, что говорит о том, что определенная цель не выполнится до того как не выполнится цель из зависимостей. Но независимо от задач решаемых сборкой существуют определенные преимущества Maven перед Ant, а именно в управлении зависимостями и интеграция со средами разработки. Рассмотрим каждое из преимуществ отдельно.

Управление зависимостями

Ни один современный проект не создается без использования сторонних библиотек (зависимостей). Эти сторонние библиотеки зачастую тоже, в свою очередь, используют библиотеки разных версий. Maven позволяет управлять такими сложными зависимостями, позволяет разрешить конфликты версий. В случае необходимости предоставляет возможность легко переходить на новые версии библиотек. Maven анализирует все зависимости прописанные в pom.xml (файл сборки Maven) и в случае необходимости автоматически подгружает требуемые зависимости из центрального репозитория Maven, либо из других , указанных в сценарии сборки.

Интеграция со средами разработки

В отличии от Ant Maven хорошо интегрируется со всеми основными средами разработки. В любой среде проект открывается сразу настроенным, что удобно, когда над проектом работает команда разработчиков из нескольких человек и пользуется в разными средами разработки.

Описанные преимущества говорят о том, что при работе с большим проектом при наличии большого числа разработчиков Maven является более удобным и функциональным инструментом для сборки.

Для выполнения процесса сборки выбранного приложения используется дополнительный функционал для Maven, а именно плагин Tycho [5], который позволяет собрать специфическую информацию о модулях приложения, обусловленную использованием платформы OSGi. Алгоритм сборки приложения для общего случая представлен на Рис.1.

Рис. 1. Алгоритм сборки

1. Разработка и настройка файла сборки для каждого отдельного модуля проекта.
2. Редактирование файла сборки с описанием всех сторонних зависимостей.
3. Создание «родительского» файла сборки, в котором прописывается сценарий сборки, состоящий из команд, которые выполняют специальные функциональные дополнения инструмента Maven(плагины), а так же который связывает все остальные файлы сборки, присутствующие в проекте, между собой
4. Запуск «родительского» файла сборки инструментом Maven

Реализованный в работе подход по сборке многомодульного RCP приложения на платформе OSGi представляет собой комбинацию работы инструмента сборки Maven, а так же функционального дополнения к Maven под названием Tycho, который позволяет инструменту Maven работать с приложениями на платформе OSGi. Рассмотренный в работе подход позволяет собирать многомодульные приложения на основе платформы OSGi любой сложности, что говорит об его универсальности и широкой применимости.

ЛИТЕРАТУРА:

[1] Osgi Service Platform, Release 3, ©2003 IOS Press, Inc.

[2] Казарин С.А. Среда разработки Java-приложений Eclipse / С.А. Казарин, А.П. Клишин– М. : Астрель, 2008. – 77 с.

[3] Steve Holzner Ant: The Definitive Guide, 2nd Edition / Steve Holzner. O'REILLY, 2005 – 318 p.

[4] Sonatype Company Maven: The Definitive Guide/ Sonatype Company. O'REILLY, 2008 – 470 p.

[5] Tycho-Documentation [Электронный ресурс]. Режим доступа: // https://eclipse.org/tycho/documentation.php

УДК 004.4'2

А. А. Янушевский (5 курс, каф. ИУС, СПбПУ)
П.Д. Дробинцев, к.т.н., доцент

РАЗРАБОТКА АРХИТЕКТУРЫ ПРОГРАММЫ ПОДБОРА КОНФИГУРАЦИИ ДЛЯ СИСТЕМ ХРАНЕНИЯ ДАННЫХ HCI

В настоящее время количество хранимой информации растет экспоненциально. Одной из причин такого роста является то, что количество средств, позволяющих создавать информацию в электронном формате, многократно возросло, кроме этого данные средства стали доступны огромному количеству пользователей, в том числе фирмам и организациям, которые стали использовать электронный документооборот внутри своих структур. В силу чего, для хранения и обработки такого большого объема информации, персональные компьютеры стали малополезны или непригодны, поэтому началось создание систем хранения данных, основанных на специализированной клиент-серверной архитектуре. Таким образом все большую актуальность приобретают вопросы, связанные с построением и выбором типа данной архитектуры, а также вопросы, связанные с выбором используемого внутри оборудования.

На текущий момент времени одним из наиболее распространенных подходов, является создание систем хранения данных, основанных на инфраструктурах типа HCI. Hyper-converged infrastructure (HCI) – это тип инфраструктуры, основанный на программно-ориентированной архитектуре, которая тесно интегрирует вычислительные, дисковые, сетевые и виртуальные ресурсы, с другими технологиями в готовый для использования аппаратный комплекс[1]. Для систем HCI характерно то, что основным блоком системы является стойка, которая будет содержать аппаратные ресурсы, при этом каждая стойка является универсальным модулем, предназначенным, как для хранения, так и для обработки данных, в отличие от предыдущих подходов, когда модули решали четко заданную им

функциональность. При этом для объединения этих блоков в единую систему хранения данных используется специализированное программное обеспечение, которое позволяет ускорить процесс обработки данных.

Таким образом, задача конфигурации систем на базе архитектуры HCI, заключается в сборе информации о требованиях приложений, которые необходимы пользователю для обработки данных, и наполнении стойки необходимым, для выполнения требований, оборудованием, в состав которого входят серверы, коммутаторы, дисковые хранилища и т. д.

В рамках работы предлагается создание программы подбора конфигурации для систем хранения данных на основе архитектуры HCI. Архитектура данной программы представлена на рис.1.

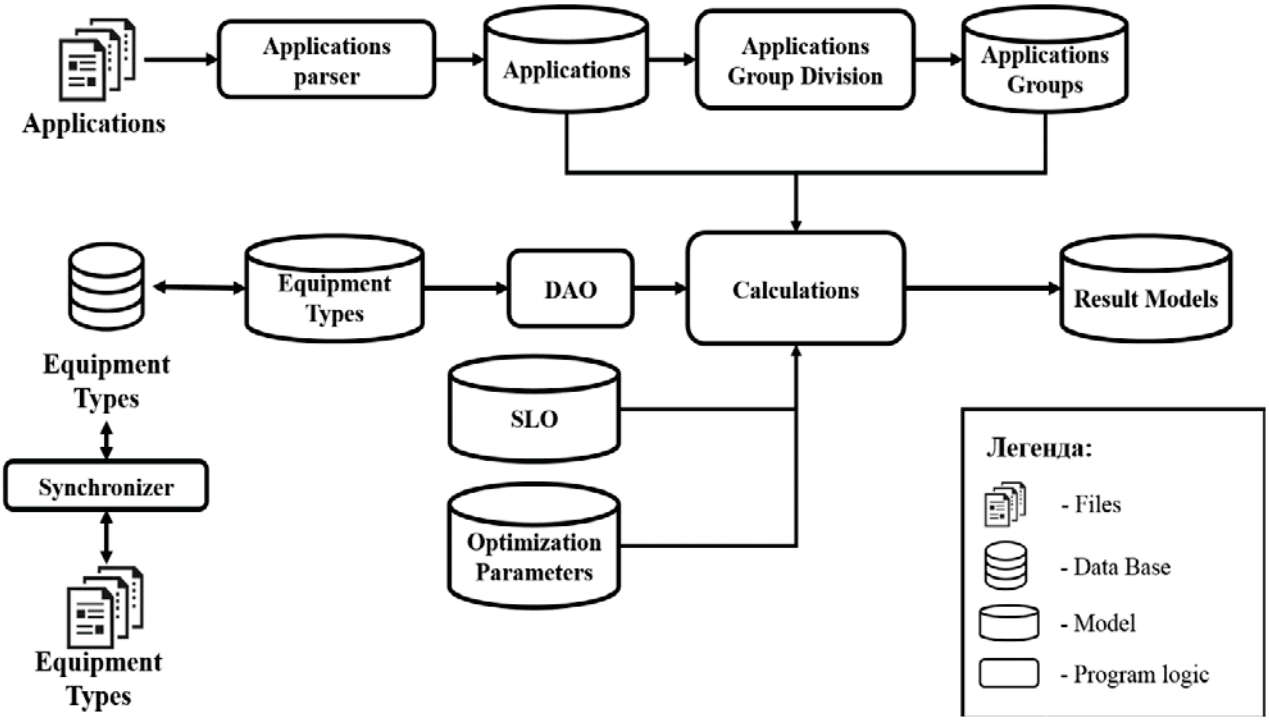


Рис. 1. Архитектура программы

Рассмотрим подробнее составные части архитектуры:

- *Applications* — это файлы или внутренняя модель программы, содержащая в себе описание приложений, которое делится на: общее описание приложения (имя, требуемый размер хранилища и другая дополнительная информация) и WorkLoad.
- *WorkLoad* — это требования приложения к серверу, каналам связи и хранилищам информации на которых данное приложение будет функционировать. В общем виде WorkLoad содержит в себе требования на количество IOPS, размеры одной операции чтения/записи, процент операций чтения/записи проводимый на произвольные/последовательные части памяти и т. д.
- *Equipment Types* — это файлы, база данных и внутренняя модель программы, хранящие в себе информацию по типам оборудованию, которое будет

использоваться при расчете конфигурации системы. К типам оборудованию относится: стойки, серверные платформы, коммутаторы, процессоры для серверов, типы памяти для серверов, хранилища данных, порты.

- *SLO* – это требования пользователя ко всей системе в целом, относящиеся к надежности системы и допустимому времени простоя системы в случае отказа.
- *Optimization Parameters* — это один или несколько параметров, которые указываются конечным пользователем и влияют на результат. К таким параметрам, к примеру, относится стоимость, то есть в данном случае пользователь хочет минимальную стоимость
- *Applications Groups* — это требования к приложениям, разделенные на группы по SLO, а затем по WorkLoad.
- *Applications parser* — это логический блок, который отвечает за преобразование информации о требованиях приложений из файлов во внутреннюю модель программы.
- Applications Group Division — это логический блок, который делит требования приложений на группы, учитывая SLO, а затем WorkLoad.
- *Synchronizer* — это логический блок, в котором происходит синхронизация базы данных с информацией, содержащейся в файлах. В данном блоке сначала происходит преобразование из файлов во внутреннюю модель, а затем из внутренней модели в базу данных. База данных связана со внутренней моделью при помощи технологии Hibernate.
- *DAO (Data Access Objects)* – это логические блоки, которые позволяют извлекать информацию из базы данных во внутреннюю модель. В данной программе для связи используется технология hibernate[2,3].
- Result Models — это набор вариантов конфигурации системы хранения с архитектурой HCI.
- *Calculations* — это логический блок, который берет себе на вход группы приложений, SLO, типы оборудования и оптимизационные параметры и, при помощи специальных алгоритмов, получает список возможных конфигураций систем хранения данных.

Так как основная работа по расчету системы хранения и подбору подходящего оборудования происходит в блоке Calculation, то рассмотрим архитектуру данного блока подробнее, она представлена на рис. 2.

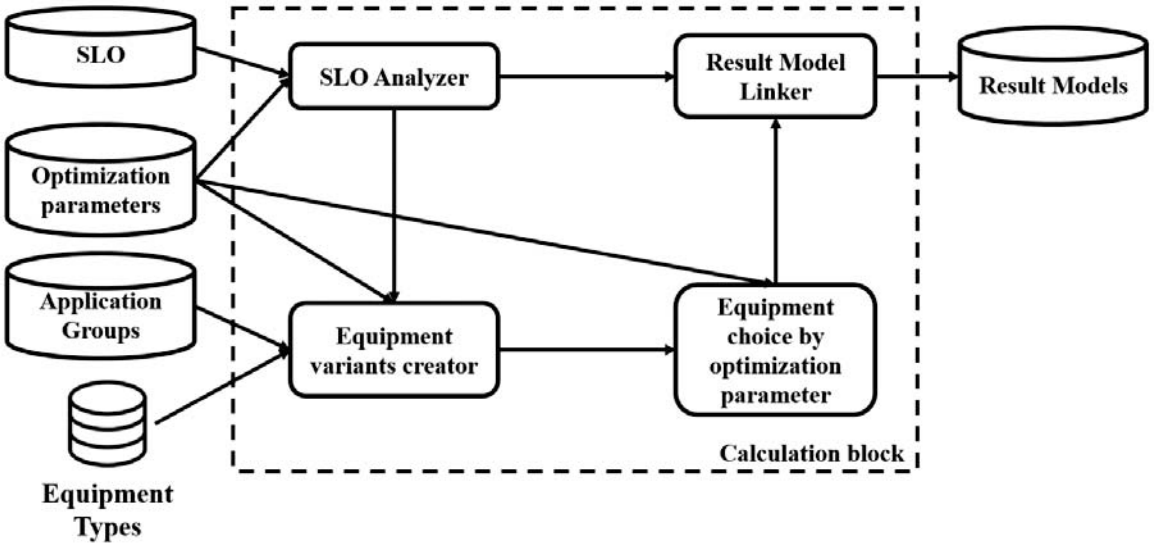


Рис. 2. Архитектура блока Calculation

Теперь рассмотрим составляющие части данного блока:

- *SLO Analyzer* – это блок, который отвечает за анализ SLO и формирование требований к комбинациям оборудования и допустимых связей между оборудованием в системе, которые будут учитываться при подборе оборудования и расчета связей в системе.
- *Equipment variants creator* – это блок, который отвечает за расчет возможных вариантов комбинаций оборудования (подбирается только оборудование, на данном этапе не происходит добавление связей между элементами системы) на основе требований приложений разбитых на группы и оптимизационных параметров.
- *Equipment choice by optimization parameter* – этот блок отвечает за уменьшение возможных вариантов комбинаций оборудования на основе оптимизационного параметра.
- *Result Model Linker* – данный блок отвечает за формирование конечного результата. Он берет на вход оставшиеся варианты комбинаций оборудования и формирует связи между оборудование для каждого варианта. В итоге получаются итоговые модели систем хранения данных HCI.

В результате проведения работы была реализована архитектура программы, позволяющей получить конфигурацию системы хранения данных с архитектурой HCI, на базе информации о требованиях приложений и базы данных типов доступного оборудования и на основе данной архитектуры был реализован прототип приложения, разработанный в Spring Tool Suite[4], с использование технологий Hiberbate и Jackson[5].

ЛИТЕРАТУРА:

1. http://searchvirtualstorage.techtarget.com/definition/hyper-convergence

2. C McKenzie. K Sheehan. Hibernate Made Easy: Simplified Data Persistence with Hibernate and JPA Java Annotations. – USA: PulpJava, 2008. 442 с.

3. http://hibernate.org/

4. https://spring.io/

5. http://jackson.codehaus.org/

СОДЕРЖАНИЕ