ЗМІСТ

ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Белобородов А.Ю. Система поиска MIDI-файлов по заданной мелодии

ПРИКЛАДНІ ДОМЕНИ І ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Дидук К.С. Визуализация автоматических процессов управления испытаниями систем космических аппаратов

Анохін А.С. Дослідження в умовах проектування програмного забезпечення в розподільчій системі для XML-орієнтовних СУБД

Можаровский П.Ф., Волошин А.В. Планировщик задач системы управления инфориационно-телекоммуникационной системой

Олексишин В.В. Дослідження ефективності використання програмного забезпечення в лікувальних закладах України

Павлов М.В. Система централизованного управления территориально разрозненной ИТ-инфраструктурой

Рябокінь Ю.М. Розробка інтерфейсу пульта інструктора АТ

ТЕОРЕТИЧНІ ОСНОВИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

СТРУКТУРУВАННЯ МОДЕЛЕЙ ПРЕДСТАВЛЕННЯ АРХІТЕКТУРИ ПРОГРАМНИХ СИСТЕМ

І.О.Галай

Національний авіаційний університет
(науковий керівник Харченко О.Г.)

При проектуванні архітектури програмних систем (ПС) використовується велика кількість різноманітних моделей представлення архітектури. При цьому відсутня систематизація цих моделей а також відсутні обґрунтовані рекомендації щодо їх використання на певних етапах життєвого циклу ПС. На основі аналізу використовуваних моделей та процесів проектування пропонується структурувати їх у вигляді трирівневої ієрархічної структури, з виділенням наступних рівнів: концептуальний, структурний, процесний. Процес проектування архітектури виконується в три етапи. Моделі концептуального рівня використовуються на самих ранніх етапах проектування і служать для декомпозиції ПС на великі складові. Цей етап проектування можна назвати концептуальним етапом. Показано, що найбільш адекватною і сучасною концептуальною моделлю є модель М.Фаулера, в якій ПС представляється у вигляді трьох шарів: шар представлення, шар бізнес-логіки; шар джерел-даних. Основною перевагою цієї моделі є можливість забезпечення незалежності шарів моделі, що полегшує внесення змін на наступних етапах проектування. В більшості моделей цього рівня компоненти ПС представляються у вигляді сукупності структурних моделей і зв’язків між ними. Найбільш широко використовуються на цьому рівні «зразки проектування» систематизовані Е.Гамма (Gamma). Пропонується їх використовувати для декомпозиції складових ПС, отриманих на етапі концептуального проектування.

Отримана ієрархічна сукупність моделей трьох рівнів і представляє модель архітектури ПС. Використовуючи принципи координації ієрархічних систем моделі узгоджуються по управлінню.

ВИМІРЮВАННЯ – ОСНОВНИЙ МЕТОД ЕМПІРИЧНОЇ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

О.П. Дишлевий
Національний авіаційний університет
(науковий керівник Сидоров М.О.)

Існують загально наукові та конкретно наукові емпіричні методи пізнання. До загально наукових належать експеримент, спостереження та вимірювання. До конкретно наукових методів програмного забезпечення належать контрольовані експерименти, дослідження ситуацій, дослідження-огляди.

Загально наукові методи в тій чи іншій мірі використовуються в конкретно наукових методах. Проведення експериментів та оглядів на сьогоднішній день є дуже дорогим та трудоємким методом. При проведенні експериментів в більшості випадків необхідна присутність розробників програмного забезпечення, так як вони в найкращій мірі зможуть пояснити всі процеси при розробці досліджуваного програмного забезпечення. Для проведення спостережень потрібні великі об’єми даних – досліджуваного програмного забезпечення. Це в свою чергу тягне високі витрати часу та фінансів.

Тому, виходячи із особливостей програмного забезпечення, вимірювання є основним загально науковим емпіричним методом досліджень програмного забезпечення. Вимірювання програмного забезпечення провести досить просто. Для цього використовуються спеціальні програмні засоби – вимірювачі. За їх допомогою можна отримати великий об’єм даних метрик. Метрики є інструментом досліджень при вимірюваннях. На сьогоднішній день метрики дозволять отримати достатньо необхідної інформації про досліджуване програмне забезпечення. Особливостям проведення вимірювань програмного забезпечення та використання метрик буде присвячена доповідь.

ЭКОСИСТЕМЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Сливкина С.

Национальный авиационный университет
(научный руководитель Хоменко В.А.)

В последнее время термин «экосистема» начал применятся в программном обеспечении для обозначения социально-технического комплекса объектов и субъектов, образующегося при его создании, эксплуатации и сопровождении. Термин «экосистема» в этом понимании можно найти в документах таких крупнейших производителей программного обеспечения как Oracle, SAP, Microsoft, а также в некоторых исследовательских работах. Все это свидетельствует о том, что определенные инвариантные структуры и закономерности экосистем могут использоваться при рассмотрении и исследовании систем, связанных с программным обеспечением. Аналогии между экосистемами и системами программного обеспечения обеспечиваются определенными сходными характеристиками – наличием развития, цикличностью процессов, потреблением ресурсов, обменом продуктов (вещества) и т.п. Специалисты в программном обеспечении, перенося вышеупомянутые термины в свою область, интуитивно чувствуют это сходство. Предполагая, что использование терминологического, понятийного и описательного аппаратов экологии имеет определенные перспективы и может быть полезно в инженерии программного обеспечения, автор данного доклада предлагает перечень задач, которые должны быть решены для их полноценного введения и использования в программном обеспечении, а именно: определение терминов экологии; исследование и описание экосистем и их элементов; создание обобщенных моделей экосистем; исследование существующих экосистем; отображение известных на сегодняшний день процессов и закономерностей инженерии программного обеспечения на модели таких экосистем. Предлагаются результаты исследования экосистем программного обеспечения, проведенные на основе опубликованных работ, общедоступных материалов и веб-ресурсов ведущих производителей программного обеспечения: базовый терминологический аппарат, перечень основных элементов и обобщенная модель экосистем программного обеспечения.

ЕВОЛЮЦІЯ І СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ИСТОРИЧЕСКАЯ МЕТАМОДЕЛЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

И.А.Дудник

Национальный авиационный университет
(научный руководитель Нечай А.С.)

Дефект проектирования – это несоответствие структурных характеристик элемента или фрагмента конструкции программы правилам проектирования. Для мониторинга эволюции дефектов проектирования программного обеспечения необходима специальная историческая метамодель объектно-ориентированного программного обеспечения. Историческая метамодель объектно-ориентированного ПО это точное определение сущностей программного обеспечения, их свойств и взаимоотношений, используемое для статического анализа. В качестве сущностей, кроме элементов конструкции программного обеспечения (метод, класс, пакет), в метамодель так же включены их версии, истории и дефекты. Метамодель используется для вычисления исторических метрик и построения набора представлений, с помощью которых имеется возможность диагностировать развивающийся во времени дефект проектирования. В предлагаемой метамодели дефект впервые моделируется как отдельная сущность, способная менять свои характеристики во времени. Степень развития дефекта проектирования это количественная характеристика отклонения проектного решения от правил проектирования. Дефект имеет степень развития, и диагностическим признаком является нестабильность его степени развития от версии к версии. Версия элемента конструкции программного обеспечения добавляет понятие времени к элементу, относя его к истории. Версия идентифицируется меткой времени и знает историю, частью которой является. Версия может существовать только в одной истории. История элемента конструкции программного обеспечения содержит множество версий. Историческая метамодель дает возможность использовать алгоритмы для вычисления исторических метрик и построить представления, необходимые для мониторинга эволюции дефектов проектирования.

ИНСТРУМЕНТАЛЬНОЕ СРЕДСТВО ДЛЯ МОНИТОРИНГА ЭВОЛЮЦИИ ДЕФЕКТОВ ПРОЕКТИРОВАНИЯ

И.В.Иванов

Национальный авиационный университет
(научный руководитель. Хоменко В.А.)

Одна из наиболее значительных проблем анализа эволюции программного обеспечения – это справится со сложностью, связанной с большим количеством данных, которые необходимо рассмотреть. При выполнении анализа программного обеспечения, визуализация дает возможность значительно упростить выявление важной информации о программном обеспечении и выполнять навигацию по ней. Визуализация программного обеспечения – это использование типографии, графического дизайна, анимации, кинематографии и современных технологий компьютерной графики для упрощения понимания и использования программного обеспечения. Ключевую роль при построении представлений для мониторинга эволюции дефектов проектирования играет степень развития дефекта. Степень развития дефекта проектирования это количественная характеристика отклонения структурных характеристик элемента или фрагмента конструкции программы от правила проектирования.

Сущность ранней диагностики дефектов проектирования программного обеспечения состоит в использовании специальной исторической модели объектно-ориентированного программного обеспечения для вычисления исторических метрик и построения набора представлений – «Рентгенограмма» и «Эволюция дефекта», с помощью которых имеется возможность диагностировать развивающийся во времени дефект проектирования.

Представление «Рентгенограмма» предназначена для общего анализа распада программного обеспечения. С ее помощью исследуется распределение дефектов проектирования по элементам конструкции и версиям программного обеспечения.

Представление «Эволюция дефекта» предназначена для детального анализа эволюции дефектов проектирования программного обеспечения. С ее помощью, исследуются конкретные элементы конструкции программного обеспечения с целью мониторинга эволюции дефектов проектирования во времени.

МЕТОД ПОСТРОЕНИЯ МОДЕЛЕЙ ДЕФЕКТОВ ПРОЕКТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

А.С.Нечай

Национальный авиационный университет
(научный руководитель Сидоров Н.А.)

Расходы на сопровождение ПО достигают 87-90% бюджета его жизненного цикла. Значительная часть усилий во время сопровождения направлена на противодействие распаду ПО. Поскольку распад ПО проявляется через дефекты проектирования, актуальной является задача их обнаружения и мониторинга. Сущность предложенного метода построения моделей дефектов проектирования заключается в том, что описание состояния элемента программного обеспечения, пораженного дефектом, осуществляется путем комбинирования функций нахождения интенсивности признаков дефекта на основе знаний объектно-ориентированного проектирования, что позволяет количественно оценивать степень развития дефекта. Для построения модели путем использования предложенного метода необходимо выполнить следующие шаги:

– сформулировать правило проектирования элемента программного обеспечения;

– выполнить анализ правила и определить признаки его нарушения;

– определить метрику для каждого простого признака;

– установить пороговое значение для каждой из метрик;

– построить функцию для оценки степени развития дефекта путем комбинирования функций нахождения интенсивностей его признаков.

Построенные с помощью предложенного метода модели могут быть использованы для мониторинга эволюции дефектов проектирования, который дает возможность определить следующее:

– образцы изменения степени развития дефекта проектирования, такие как увеличение, уменьшение, пульсирование или стабильность;

– версию, в которой появился дефект проектирования или множество версий, в которых происходил рост или уменьшение его степени развития.

МОДЕЛЮВАННЯ ДЕФЕКТІВ ПРОЕКТУВАННЯ

А.А.Шаповал

Національний авіаційний університет
(науковий керівник. Нечай О.С.)

В інженерії програмного забезпечення для дефекту, як і для якості, не існує єдиного визначення, однак найбільш часто дефект визначають як відхилення від специфікацій або очікувань, що може викликати збій або відмову програмного забезпечення.

Дефект проектування – це невідповідність структурних характеристик елемента або фрагмента конструкції програми правилам проектування.

Дефекти проектування класифікують наступним чином: за можливістю подальшого використання – критичні, значні, не значні; за рівнем абстракції ураженого елемента конструкції – рівня методу, рівня класу, рівня підсистеми; за здатністю до вимірювання – вимірювані, не вимірювані.

Для моделювання використовується метод побудови моделей дефектів проектування об'єктно-орієнтованого програмного забезпечення.

За допомогою методу побудовано наступні моделі дефектів: God Class, Brain Method, Shotgun Surgery.

God Class – дефект проектування рівня класу, що характеризується тенденцією до централізації обчислення, виконанням великої кількість роботи з власними атрибутами, та використанням даних з інших класів.

Brain Method – дефект проектування, рівня методу, що характеризується централізацією у собі майже всієї функціональності класу. Зазвичай це метод з дуже великою кількістю рядків коду, який важко зрозуміти і практично неможливо повторно використовувати.

Shotgun Surgery – дефект проектування, рівня методу, при якому зміни в одному методі ведуть за собою багато малих змін в інших методах чи класах.

Отримані моделі дефектів проектування були застосовані для пошуку дефектів у проектах з відкритим кодом Vuze та ArgoUML за допомогою розробленого засобу SEM. Адекватність моделей перевірена шляхом ручного перегляду вихідного коду.

КОЛЕКТИВНА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

УПРАВЛЕНИЕ ПРОГРАММНЫМ ПРОЕКТОМ С УЧЕТОМ КОММУНИКАЦИЙ МЕЖДУ ЕГО УЧАСТНИКАМИ

Т.Н. Назаренко

Национальный аэрокосмический университет им. Н.Е. Жуковского "ХАИ" (научный руководитель Федорович А.Е.)

Инженерия программного обеспечения остается сложной сферой деятельности человека. Microsoft Solutions Framework (MSF) – методология разработки ПО от компании Microsoft, опирающаяся на практический опыт компании и описывающая управление людьми и процессами в ходе разработки. Несмотря на все достоинства, на практике всегда существует необходимость адаптации общей методологии к специфике конкретного проекта. Важной задачей остается планирование коммуникаций и организация взаимодействия между участниками команды проекта, ведь именно благодаря коммуникациям достигается контроль и управление процессами разработки и становится возможным появление *синергетического эффекта* (эмерджентности), когда свойства целого (команды проекта) не сводятся к «сумме» свойств его частей. Поэтому разработка моделей, методов и инструментальных средств для оптимизации процессов управления программным проектом с учетом коммуникативных взаимодействий между его участниками является актуальной научной задачей. В докладе предложена системная модель управления проектом с учетом коммуникаций между его участниками, которая позволит формально представить взаимосвязь между уникальными характеристиками проекта, бизнес-процессами, происходящими при его реализации, участниками проекта и их взаимодействием, а также влияния этих взаимосвязей на результат проекта. Сформулирована постановка задачи оптимизации процессов коммуникаций в организационных структурах управления проектом как формирование множеств исполнителей и управленцев, а затем поиск такого их назначения на проектные роли в конкретных задачах, с последующим планированием коммуникаций между участниками, которое обеспечивает максимизацию интегрального показателя эффективности проекта.

ТЕСТУВАННЯ, ВАЛІДАЦІЯ ТА ВЕРИФІКАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

К.Резник

Национальный авиационный университет
(научный руководитель Сидоров Н.А.)

Тестирование как объект изучения может рассматриваться с различных чисто технических точек зрения. Однако наиболее важными при изучении тестирования представляются вопросы его экономики и психологии разработчика. Достоверность тестирования программы в первую очередь определяется тем, кто будет ее тестировать и каков его образ мышления, и уже затем определенными технологическими аспектами. Сформулируем основные принципы тестирования, используя главную идею статьи, что наиболее важными в тестировании программ являются вопросы психологии. Эти принципы интересны тем, что в основном они интуитивно ясны, но в то же время на них часто не обращают должного внимания. Описание предполагаемых значений выходных данных или результатов должно быть необходимой частью тестового набора. Ошибочные, но правдоподобные результаты могут быть признаны правильными, если результаты теста не были заранее определены. Здесь мы сталкиваемся с явлением психологии: мы видим то, что мы хотим увидеть. Несмотря на то, что тестирование по определению – деструктивный процесс, есть подсознательное желание видеть корректный результат. Один из способов борьбы с этим состоит в поощрении детального анализа выходных переменных заранее при разработке теста. Поэтому тест должен включать две компоненты: описание входных данных и описание точного и корректного результата, соответствующего набору входных данных.

Необходимость этого подчеркивал логик Копи в работе: «Проблема может быть охарактеризована как факт или группа фактов, которые не имеют приемлемого объяснения, которые кажутся необычными или которые не удается подогнать под наши представления или предположения. Очевидно, что если что-нибудь подвергается сомнению, то об этом должна иметься какая-то предварительная информация. Если нет предположений, то не может быть и неожиданных результатов».

ОЦІНКА ВИТРАТ ТА ВАРТОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КАТЕГОРІЇ ПІДХОДІВ ДО ОЦІНКИ ВАРТОСТІ ПРОЕКТІВ

Д.В.Баценко

Національний авіаційний університет
(науковий керівник Сидоров М.О.)

Існує шість категорій підходів до оцінювання вартості програмного забезпечення: 1) підходи, в основу яких покладено математичні моделі, які калібруються за відомими даними з попередніх проектів (COCOMO, SLIM, SEER, Checkpoint, ESTIMACS); 2) підходи, які засновані на думках експертів з певним досвідом оцінювання програмного забезпечення (Delphi, Rule-Based); 3) підходи, засновані на моделях, що здатні самостійно навчатися, використовуючи штучний інтелект та машинні методи навчання для обробки історичних даних по проектам (Neural, Case-Based); 4) динамічні підходи, що засновані на принципі постійної переоцінки вартості програмного забезпечення в процесі його розробки (Abdel Hamid-Madnick); 5) регресійні підходи, що використовують методи математичної статистики для побудови моделей на основі аналізу великих обсягів історичних даних по проектам (Ordinary Least Squares та Robust); 6) композитні підходи, що використовують кілька підходів з різних категорій (Bayesian, COCOMO II, Reliability Growth).

Перевагами першої категорії є математична обгрунтованість та прогнозованість результатів, можливість калібрування, чіткий процес. Перевагами другої категорії є врахування специфічних характеристик проектів, гнучкість процесу аналізу, відсутність потреби в калібруванні. До переваг третьої категорії відносяться формалізований підхід та можливості самовдосконалення моделі. Перевагами четвертої категорії є можливість використання в процесі роботи над проектами та швидкого корегування результатів оцінки в результаті змін, внесених до проекту. Перевагами п’ятої категорії є формалізованість підходу, аналіз попередніх проектів, можливість застосування результатів для калібрування математичних моделей. Переваги шостої категорії є результатом поєднання переваг усіх моделей, що застосовуються в процесі оцінки.

ЯКІСТЬ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ТЕХНОЛОГІЯ ОЦІНКИ ЯКОСТІ ПРОГРАМНИХ СИСТЕМ, ОРІЄНТОВАНА НА ФОРМАЛЬНИЙ АНАЛІЗ ВІДГУКІВ КОРИСТУВАЧІВ

І.В.Запорожець

Національний авіаційний університет
(науковий керівник Сидоров М.О.)

До теперішнього часу зроблені серйозні напрацювання в області стандартизації процедури управління якістю, а також зі створення технологій і методик оцінки якості програмних систем (ПС). На підставі проведеного аналізу (в контексті розвитку технології MSF) запропонована нова технологія оцінки якості: 1.Побудова ієрархічної моделі показників якості, про значення яких може судити користувач, за рахунок залучення експертів предметної області. 2. Побудова єдиної оцінки якості ПС. Кожному показнику приписується «цінність» (виходячи з параметричної теорії парних порівнянь, модель Терстоуна-Мостеллера), що дозволяє відповідно до ієрархічної моделі, доповненої операторами перетворення, зводити значення показників якості в єдину характеристику.3. Побудова математичної моделі забезпечення якості ПС. Задача забезпечення необхідного рівня якості ПС зводиться до оптимізаційної задачі при мінімізації трудомісткості робіт для забезпечення заданого рівня якості. 4. Формування показників якості на підставі відгуків. Для формування поточних значень показників якості необхідно проведення опитування користувачів, які дають оцінку якості по кожному показнику окремо. 5. Формування оптимальної стратегії. У випадку узгодженості оцінок користувачів можлива побудова оптимальної стратегії поліпшення. 6. Впровадження обраної стратегії, контроль виконання та коригування моделі відповідно до кроків 1-5. Важливим етапом є перевірка суджень експертів на несуперечливість, а також – оцінка узгодженості відгуків користувачів. В рамках розглядуваної технології пропонується використати теорію люсіанів (нечислова статистика). Таким чином, вищенаведена технологія оцінки якості ПС доповнює існуючі методи та засоби верифікації та валідації для комплексної оцінки якості ПС.

ОЦІНЮВАННЯ ЯКОСТІ ПРИКЛАДНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА ОСНОВІ МЕТРИК ТА МОДУЛЬНИХ ТЕСТІВ

П.П.Процик
Київський національний університет ім. Т.Г.Шевченко (науковий керівник Нікітченко М.С.)

Проблема забезпечення якості програмних продуктів одна з найгостріших проблем сьогодення, як з точки зору практики так і теорії програмування. Існують різні методики її розв'язання [1], які умовно можна розділити на формальні, тобто засновані на математичних методах, не формальні та їх комбінації.

Поряд із задачею забезпечення якості стоїть задача її оцінювання. Розглядається прагматико обумовлений підхід до розв'язання оцінювання якості ПО з використанням модульних тестів та метрик. В якості прикладної платформи використовується *Microsoft Visual Studio 2008* [2].

Виникає питання: «скільки і яких модульних тестів необхідно для забезпечення прийнятної якості програми?». Точної відповіді на це питання не існує, та можна дати асимптотичну оцінку ґрунтуючись на структурних властивостях програми. Цілком зрозуміло, що чим складніша структура програми тим більше необхідно тестів.

Найбільш поширеною метрикою складності програми вважається цикломатична складність [3] – яка дорівнює кількості лінійно незалежних шляхів у програмі. В теорії один модульний тест повинен відповідати одному такому шляху. Пропонується наближена оцінка кількості тестів: $U(C,k)=\alpha C+P(k)$, де C – значення цикломатичної складності, α – параметр довіри тесту, як правило дорівнює 1, k – кількість параметрів програми, $P(k)$ – деяка міра впливу параметрів на кількість тестів, наприклад $P(k)=2^k+k$. На практиці, $P(k)$ встановлюється емпіричним шляхом так, щоб тести покривали від 75% до 90% програмного коду. Степінь покриття тестами коду можна обчислити у середовищі розробки *MS Visual Studio 2008*.

Висновки. Розглянуто метод оцінки та забезпечення якості ПЗ на основі метрик та модульних тестів, запропоновано підхід до оцінювання мінімальної кількості.

ПОБУДОВА МОДЕЛІ ОЦІНКИ ЯКОСТІ СКБД

О.В.Тегельман

*(Національний авіаційний університет
науковий керівник Харченко О.Г.)*

Оцінювання якості систем керування базами даних(далі СКБД) проводиться як їх розробниками, так і незалежними лабораторіями й публікуються в літературі та в мережі Internet. Однак скористатись цими даними при виборі СКБД дуже складно, так як в них використовуються не стандартизовані,корпоративні критерії оцінки якості. А це призводить до неоднозначності тлумачення даних показників та підміні одних показників іншими. Для отримання об'єктивних стандартизованих показників якості й використанні їх при порівнянні та виборі СКБД пропонується використати модель якості стандарту ISO/IEC 9126. У відповідності зі стандартом, якість оцінюється сукупністю характеристик, підхарактеристик, атрибутів і метрик, які й складаються модель якості.

Для побудови моделі якості СКБД було визначено, на основі аналізу предметної області їх застосування, сукупність атрибутів якості. Для кожного атрибуту якості було визначено до якої стандартизованої характеристики він відноситься, а також були вибрані метрики. Таким чином було побудовано модель якості СКБД, яка включає 14 атрибутів, серед яких такі, як :

- функціональна повнота;
- захищеність;
- безвідмовність;
- зрозумілість;
- адаптивність;
- простота установки;
- сумісність та інші.

Побудована модель була використана для порівняння СКБД ORACLE та MS SQL Server 2005. Дані порівняння показали, що СКБД ORACLE за своїми можливостями перевершує MS SQL Server 2005 за більшістю вибраних атрибутів.

Таким чином модель якості стандарту ISO/IEC 9126 може успішно бути використаною для побудови моделі оцінки якості будь-якого програмного продукту, в тому числі і при оцінці якості СКБД.

CASE-ЗАСІБ ПІДТРИМКИ ПРОЦЕСУ ОЦІНЮВАННЯ ЯКОСТІ WEB- ЗАСТОСУВАНЬ

В.В.Яцишин

Тернопільський державний технічний університет ім. І. Пулюя
(науковий керівник: О.Г. Харченко)

Web-простір насичений безліччю сайтів, але якість більшості з них є досить низькою. Основною причиною такої ситуації є те, що більшість розробників або не оцінюють їх якість або використовують для цього корпоративні, нестандартизовані атрибути та процедури оцінювання. Іншою причиною такої ситуації є висока трудомісткість та вартість процедури оцінювання.

При оцінюванні якості web-застосувань пропонується використовувати модель, побудовану у відповідності з рекомендаціями стандарту ISO 9126-1. В моделі якості використовуються стандартизовані характеристики, підхарактеристики та метрики. Для відображення специфіки предметної області були сформульовані атрибути підхарактеристик, які для зручності було розбито на декілька груп. Механізм використання моделі якості при оцінюванні web-застосувань, зводиться до визначення атрибутів предметної області, класифікації атрибутів, тобто віднесення до відповідної характеристики (підхарактеристики) і порівняння фактичного значення показника атрибута з метрикою, що визначена у стандарті ISO 9126-2.

Для зменшення трудомісткості даної процедури було розроблено CASE-засіб автоматизації, який може використовуватись при розробці специфікацій системних вимог до ПС та оцінюванні якості. На фізичному рівні даний інструмент представляє собою систему, що складається з п’яти підсистем та репозиторію стандартизованих елементів моделі якості, модифікованих під предметну область. Підсистема діалогу з користувачем представляє собою інтерфейс, що реалізовує взаємодію з іншими компонентами. Для роботи з репозиторієм передбачено підсистему з розподілом прав доступу. Крім того, до складу CASE-засобу входить підсистема завантаження користувацьких вимог, підсистема класифікації вимог і атрибутів якості, підсистема автоматичного генерування специфікації та формування документації з оцінювання якості.

ТЕХНОЛОГІЇ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

МЕТОДИ СИНТЕЗУ ПРОГРАМ ТА ПОРОДЖУЮЧОГО ПРОГРАМУВАННЯ ЯК ЗАСОБИ ОТРИМАННЯ КІНЦЕВИХ ПРОГРАМНИХ ПРОДУКТІВ

В. В.Лозицький

Національний авіаційний університет
(науковий керівник Радішевський М. Ф.)

Короткий огляд досягнень показує лише незначне зростання продуктивності розробки програмного забезпечення (ПЗ) або якості програмних продуктів. Більша частина програм в наші дні досі пишеться вручну, починаючи з чистого паперу, з використанням трудомістких методів. В результаті розробка програм проводиться повільно, багато коштує і породжує продукти, що містять серйозні дефекти, які призводять до проблем зручності використання, надійності, продуктивності і безпеки. Ці проблеми – результат підвищених вимог сучасних замовників. Однак по мірі розвитку індустрія ПЗ неодмінно мусить вийти за рамки методів, які привели її до такого стану. Оскільки неодмінним показником зрілості є рівень її автоматизації то одним з рішень може бути використання напрямків автоматизованого синтезу програм (program synthesis) та породжуючого програмування (generative programming).

Після проведення відповідного опису та аналізу в рамках зазначених напрямків було встановлено, що автоматизований синтез програм перш за все є спорідненою ідеєю до породжуючого програмування. Основна відмінність породжуючого програмування від автоматичного синтезу програм полягає в тому, що породжуюче програмування робить можливим різні рівні автоматизації, в той самий час як ціллю автоматичного синтезу програм є тільки високорівнева оптимізація. Автоматичний синтез програм зазвичай використовує технологію штучного інтелекту і потребує великої кількості інформації стосовно предметної області навіть для практичних задач середнього розміру. З іншого боку, ідея породжуючого програмування полягає в наданні практичних засобів з реальної практики в технології розробки ПЗ. Важливим питанням при цьому лишається економічна доцільність.

MODEL-VIEW-VIEWMODEL: ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ PRESENTATION MODEL В ПРЕЗЕНТАЦИОННОЙ СИСТЕМЕ WPF

И.В.Малин

Национальный авиационный университет
(научный руководитель Сидоров Н.А.)

Разделение интерфейса пользователя от остальных элементов приложения – один из ключевых принципов правильно спроектированного приложения. Для реализации этого принципа были разработаны следующие шаблоны проектирования и их модификации: MVC (Passive Model, Active Model), MVP (Passive View, Supervising Controller), Presentation Model (Model-View-ViewModel). Решая одну и ту же проблему схожими способами тот или иной шаблон фокусирует свое внимание на определенном подходе к реализации разделения интерфейса и бизнес-логики. Стоит заметить, что во всех упомянутых шаблонах создатели вводят дополнительную сущность – посредника между бизнес логикой и представлением. Введение посредника позволяет определить общий программный интерфейс взаимодействия интерфейса пользователя и программной системы.

Шаблон проектирования MVVM призван решить данную проблему в среде .NET с использованием возможностей презентационной системы WPF. Он создан как специализация более общего шаблона Presentation Model.

Основной принцип шаблона проектирования Presentation Model заключается в представлении состояния и поведения приложения, независимо от конкретного набора элементов представления.

Обычно, интерфейс пользователя состоит из набора компонентов, отображающих состояние приложения (и соответствующей ему бизнес-модели). Если сохранять состояние приложения в свойствах компонентов пользовательского интерфейса, то доступ к этим свойствам с уровня бизнес-модели усложняется. Presentation Model выделяет состояние и поведение пользовательского интерфейса в отдельный презентационный класс, который является частью представления. Presentation Model коммуницирует с доменным уровнем (уровнем бизнес-логики), в то же время предоставляя

программный интерфейс уровню презентации. В таком случае, интерфейс пользователя не хранит в своих компонентах информацию о состоянии системы, а лишь отображает состояние Presentation Model.

В случае Model-View-ViewModel, ViewModel является аналогом сущности PresentationModel родительского шаблона. Отличие заключается лишь в том, что MVVM описан применимо к особенностям WPF-реализации. Концептуально шаблоны идентичны.

В случае с MVVM все поведение пользовательского интерфейса системы выносится из представления (View) в т.н. ViewModel. Связывание View и ViewModel осуществляется декларативными связками (binding) определенными в XAML пользовательского интерфейса. Связки могут быть как двусторонними (установка и отображение значения), так и односторонними (установка или отображение значения). Обновление пользовательского интерфейса реализовано на основе модели подписки. Измененное значение в ViewModel рассылает нотификации всем подписанным компонентам, которые, в свою очередь, получают и обрабатывают новое значение. Компоненты не содержат в своих свойствах никакой информации о текущем состоянии системы, а лишь занимаются отображением и/или установкой значений ViewModel, обеспечивая, таким образом, разделение бизнес-логики и представления в приложениях WPF. Практически, MVVM позволяет без каких либо программных модификаций полностью менять вид пользовательского интерфейса, с одним лишь условием – поддержкой View (XAML ПИ) программного интерфейса ViewModel. Практически, в командах разработчиков это позволяет легко и четко разделить области ответственности. Часть разработчиков может трудиться над реализацией бизнес-модели, а другая половина реализовывать пользовательский интерфейс. При этом они независимы друг от друга. Средства WPF также позволяют иметь несколько наборов интерфейсов и менять их в зависимости от необходимости конечного пользователя в том или ином виде представлении информации.

ТЕХНОЛОГІЇ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СКЛАДНИХ РОЗРАХУНКІВ ВИКОРИСТОВУЮЧИ ВІДЕОАДАПТЕРИ

Є.І.Марчук

Національний авіаційний університет
(науковий керівник Нечай О.С.)

Багато прикладних галузей потребують високопродуктивні системи для виконання складних розрахунків, таких як суперкомп'ютери (мейнфрейм), проте доступ до таких обчислювальних потужностей досить обмежений. Існує потреба в не дорогих та ефективних рішеннях поставлених перед ними задач. Таким рішенням можуть стати програмно-апаратні технології, які базуються на використанні можливостей сучасних відеоадаптерів.

Центральні процесори є достатньо універсальними обчислювальними засобами, проте через універсальність вони мають обмеження в продуктивності при складних обчисленнях, які можливо розкласти на потоки. Рішенням проблеми можуть стати технічні можливості відеоадаптерів, які оптимізовані на обчислення незалежних потоків інформації, теоретично можна досягти приріст швидкості обчислень, впорівнянні з центральними процесорами, в десятки разів.

Розробники та спеціалісти, які працюють з ПЗ, яке виконує велику кількість паралельних та незалежних обчислень, можуть користуватися "настільним суперкомп'ютером" на базі відеопроцесорних систем. Такі рішення є значно доступнішими та енергоефективними в порівнянні з мейнфреймами, вони також забезпечують достатні обчислювальні потужності для вище вказаних потреб.

Складністю даного рішення є написання програмного забезпечення, яке б могло задіяти обчислювальні можливості відеоадаптерів. Це пов'язане з їх першочерговою орієнтацією на обробку графічних даних. Проте з розвитком ідей альтернативного використання відеоадаптерів почали розвиватися спеціалізовані технології та мови програмування, які дозволяють значно полегшити розробку такого ПЗ. Саме тому нами було систематизовано перспективні технології та мови програмування, які вже є діючими або плануються бути такими в майбутньому.

ТЕХНОЛОГИЯ РАЗРАБОТКИ АДАПТИРУЕМОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ИСПЫТАНИЙ КОСМИЧЕСКОЙ ТЕХНИКИ

Б.Б. Михнич
Национальный аэрокосмический университет «ХАИ»
(руководитель И.Б. Туркин)

Экспериментальная отработка космической техники – итерационный процесс, особенностью которого является проведение многократных тестовых отработок с возможной последующей доработкой конструкторской документации в соответствии с полученными результатами тестов. Такие доработки требуют в свою очередь изменения программного обеспечения (ПО) для автоматизации испытания. Таким образом, помимо традиционных требований к качеству и надежности ПО появляется необходимость обеспечить высокую степень его адаптируемости. Языково-ориентированный подход позволяет достичь этого свойства, идея которого основана на создании и применении специального проблемно-ориентированного языка, на котором создается описание технологических процессов испытаний с целью последующего его автоматического исполнения. Такой подход применялся при разработке ПО для автоматизации испытаний систем электроснабжения отечественных спутников, разрабатываемых ГКБ «Южное».

Современные инструментальные средства, например, конструктор Workflow Designer, позволяют решать подобную задачу на качественно новом уровне, когда у разработчика технологических процессов испытаний появляется возможность самостоятельно конструировать новые рабочие потоки контроля и управления испытаниями. Поддержка возможности динамического обновления моделей позволяет оперативно изменять ход текущего испытания и практически сразу проверить эффективность новых решений.

Языково-ориентированный подход в сочетании с современными средствами разработки ПО позволил разработать адаптируемый программный продукт, который в дальнейшем можно использовать не только для автоматизации стендовых испытаний малых космических аппаратов, но и для управления другими системами.

МОДЕРНИЗАЦИЯ МЕТОДА ВЫБОРА МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

О. В. Пащенко, А. В. Овчаренко
Национальный авиационный университет
(научный руководитель Дышлевый О.П.)

Обеспечение качества программного обеспечения представляет собой многокомпонентный процесс, зависящий от различных факторов разработки. Одним из таких факторов является модель жизненного цикла (ЖЦ), положенная в основу процесса разработки. Выбор модели ЖЦ, учитывающий особенности конкретного проекта, представляет собой первоочередную задачу подготовки процесса разработки.

Многолетний мировой опыт разработки программного обеспечения позволил выделить несколько общепринятых моделей: каскадная, V–образная, быстрого прототипирования, RAD–модель, инкрементная, спиральная.

Существующий метод выбора модели жизненного цикла базируется на классификации проектов по разработке программного обеспечения.

Его анализ показал, что: использование такого подхода приемлемо для достаточно крупных проектов, поскольку его использование для небольших проектов может привести к увеличению графика работ.

Проанализировав полученные выводы было принято решение о усовершенствованиии метода выбора модели жизненного цикла.

В качестве модернизации существующего подхода предлагается перейти к более обобщенным критериям, которые максимально будут влиять на процесс выбора адекватной модели.

Предлагаемая модернизация метода выбора модели жизненного цикла программного обеспечения позволит разрешить ситуацию возникающую при сходных или одинаковых показателях для нескольких моделей. Предлагаемый подход позволит не только выбрать наилучшую модель, но и ранжировать все остальные по обобщенному критерию полезности.

Скалова

АНАЛИЗ МЕТОДОВ РЕСУРСОСБЕРЕЖЕНИЯ НА ЭТАПЕ КОНСТРУИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

А.П.Ходаковская
Национальный авиационный университет
(научный руководитель Сидоров Н. А.)

Несмотря на стремительное развитие вычислительной техники (последние двадцать лет производительность ЭВМ увеличивалась согласно закону Мура), проблема сбережения ресурсов остается актуальной. Программное обеспечение, потребляющее для своей работы меньшее количество ресурсов, позволяет использовать современные мультизадачные среды более эффективно путем увеличения количества параллельных вычислений, и, в конечном итоге, снижает себестоимость вычислений, время и количество потребляемых энергоресурсов.

Для измерения ресурсоемкости программного обеспечения используются различные метрики. С точки зрения аппаратных ресурсов, основными являются объем оперативной памяти, загрузка вычислительного процессора, интенсивность ввода-вывода, объем занимаемой памяти на жестком диске, количество сетевых соединений и передаваемой информации. Если же проводить анализ программного кода, то метриками могут быть, например, количество вызовов и глубина вложенности подпрограмм, объем статически и динамически выделенной памяти и т.д.

Важным фактором является исследование изменения метрик в зависимости от входных данных и в процессе нагрузочного тестирования.

Согласно известному правилу 90/10, всего 10% программного кода потребляют 90% вычислительных ресурсов. Именно эти участки и являются, очевидно, «узким местом» продукта. Их выявление и оптимизация являются первостепенной задачей при улучшении производительности приложений.

В работе проводится анализ методов разработки программного обеспечения с целью нахождения максимально эффективных алгоритмов организации структур данных и их обработки.

УНИФИЦИРОВАННЫЙ ПРОЦЕСС РАЗРАБОТКИ ПРИ МОДЕЛИРОВАНИИ ЖИЗНЕННОГО ЦИКЛА ИТ-ПРОЕКТА

А.В. Ходыревская

Харьковский национальный экономический университет
(научный руководитель Золотарева И.А.)

Целью бизнес-моделирования является исследование бизнес-среды, в которой будет эксплуатироваться система. Набор задач, требующих решения в ходе моделирования, следующий:

необходимо исследовать бизнес-среду организации, в которой будет размещена разрабатываемая система;

определить задачи, которые заказчик будет решать с помощью данной системы;

убедиться в том, что группа разработчиков и заказчик одинаково видят бизнес-среду организации;

определить требования заказчика к системе [1].

Для изучения высокоуровневых требований к системе унифицированный процесс фирмы Rational предлагает создание бизнес-модели Use Case, включая создание диаграммы Use Case и сопровождающей документации [1, 2].

Существует три основные модели, которые необходимо разработать организации для описания инфраструктуры:

1. Корпоративная модель требований. Эта модель отражает высокоуровневые требования организации и описывает услуги, которые организация предоставляет в своей внешней среде.

2. Модель архитектуры предметной области, отражающая высокоуровневую бизнес-структуру разрабатываемой системы.

3. Модель технической архитектуры, представляющая собой высокоуровневую техническую инфраструктуру [3, 4].

Важной причиной проведения сеанса моделирования требований совместно с заинтересованными лицами является достижение общего видения бизнес-среды.

ОСВІТА ТА ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КОМПЬЮТЕРНАЯ ПОДДЕРЖКА РЕШЕНИЙ В УЧЕБНОЙ ИНФОРМАЦИОННОЙ ТЕХНОЛОГИИ СРЕДСТВАМИ ИНЖЕНЕРИИ КВАНТОВ ЗНАНИЙ

А.В. Григорьев, С.М. Тустановский

Национальный аэрокосмический университет «ХАИ»
(руководитель Сироджа И.Б.)

Целью работы является описание созданной учебной информационной технологии КВАНТ-УЧ для повышения качества проведения лабораторных работ в техническом университете по курсу «Системы искусственного интеллекта» по специальности «Программная инженерия». Ядро интеллектуальной поддержки принятия решений базируется на средствах инженерии квантов знаний предложенной профессором Сироджей И.Б. с использованием метода разноуровневых алгоритмических квантов знаний (РАКЗ-метод). В РАКЗ-методе, реализована строгая формализация используемых знаний в классе М содержательных алгоритмических структур (квантов знаний) различных уровней сложности (0-й уровень: число, символ; 1-й уровень: вектор, функция; 2-й уровень: матрица, композиция функций). Кванты знаний (к-знания) образуются алгоритмически из терминальных структур посредством операторов суперпозиции и конкатенации. Обучение квантовых сетей вывода решений (КСВР) осуществляется по выборочным таблицам эмпирических данных (ТЭД) и/или по сценарным примерам обучающих знаний (СПОЗ) исследуемой предметной области. Обучение завершается построением КСВР, как базы квантов знаний (БкЗ), и её оптимизацией по критерию структурной избыточности. Индуктивный вывод (посредством обучения ЭВМ) БкЗ и дедуктивный вывод исходных идентификационных и прогнозных решений, опираясь на БкЗ, обеспечивает компьютерную поддержку условия учебного материала студентом, а также требуемый преподавательский контроль. Принятие идентификационного решения сводится к определению с заданной надежностью η неизвестного значения целевого классификационного признака, по которому объект принятия решения (ОПР) относится к определенному классу (категории) на основе анализа значений наблюдаемых признаков ОПР.Принятия прогнозных решений состоит в определении с заданной надёжностью η неизвестных r значений экстраполируемых признаков ОПР по известным (n-r) значениям наблюдаемых признаков**,** опираясь на имеющуюся БкЗ.

АНАЛИЗ ТЕКСТОВ УЧЕБНЫХ МАТЕРИАЛОВ ДЛЯ ЗАПОЛНЕНИЯ ТЕЗАУРУСА СИСТЕМЫ ДИСТАНЦИОННОГО ОБУЧЕНИЯ

А.Л.Данченко
Східноукраїнський національний університет імені Володимира Даля
(кер. Ульшин В.О.)

Построение системы дистанционного обучения (СДО) на основе семантических связей между знаниями дистанционных курсов (ДК) предоставляет расширенные интеллектуальные возможности адаптивного обучения.

Существующие СДО, такие как Blackboard, WebCt, Moodle, IBM LearningSpace и т.д. поддерживают международные стандарты дистанционного обучения, используют технические возможности информационных технологий, однако не применяют технологии знаний. Целью статьи является разработка методов извлечения знаний из учебных материалов для заполнения тезауруса СДО, основанной на онтологической модели представления знаний.

Формирование семантических связей между знаниями СДО основывается на онтологической модели представления знаний.

Автоматизированная семантическая разметка текста выполняется в два этапа. С помощью средств автоматизации выполняется вероятностное аннотирование текста, определение Терминов текущего ДК, построение гипотез по установке отношений между Терминами текущего ДК и другими Терминами тезауруса СДО. Затем выполняется ручная корректировка, которая состоит в подтверждении, модификации или удалении предлагаемых Терминов и гипотез.

Для получения входных параметров, необходимых при кластеризации учебных материалов, используются словарь стоп-слов, базовый морфологический словарь, данные Тезауруса, частотные характеристики и данные о расположении лексем в текстах.

Климук

ЗАХИСТ ТА ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ИНФОРМАЦИОННО-УПРАВЛЕНЧЕСКАЯ АРХИТЕКТУРА ОРГАНИЗАЦИИ И ЗАЩИТА ИНФОРМАЦИИ

В.И.Андросов

Восточноукраинский национальный университет им. В.Даля
г.Луганск
(научный руководитель.. Динич В. Н.)

Защита производственной тайны на современном предприятии – достаточно сложная задача и для ее эффективного выполнения требуется согласование информационной и управленческой архитектур организации [1].

Только при этом условии будет достигнут эффект синергии, взаимного усиления и дополнения организационных и информационных подходов к защите коммерческой тайны[2].

Была разработана система надстроек для серверной операционной системы Windows 2003 Server, которые позволяют разграничивать доступ к информационным ресурсам на основе архитектуры предприятия.

Основная функциональность надстройки.

1. Возможность привязывать рабочие станции к подразделениям в которых они расположены. В результате пользователи из других отделов не смогут получить доступ к локально хранимой информации.
2. Создание особых групп безопасности, привязанных к подразделениям. Таким образом становится возможным назначать права доступа к ресурсам подразделениям. Поддержка актуальности составов групп делается автоматически.

Реализация более тонкого механизма делегирования полномочий на модификацию модели архитектуры организации.

АНАЛИЗ МЕТОДОВ ИСПОЛЬЗУЕМЫХ ПРИ ВЗЛОМЕ ПРОГРАММНОГО ПРОДУКТА

Ю.Н. Безкоровайная

Национальный авиационный университет
(научный руководитель Радишевский Н.Ф.)

При создании программных продуктов (ПП) разработчик часто незаконно использует части кода, созданные другими компаниями. Другими словами «заимствует» программный код. Это позволяет разработчику уменьшить времени создания собственного ПП и уменьшить стоимость.

Создано ряд программных (цифровая подпись) и законодательных мер (Закон Украины "Об авторском праве и смежных правах") по защите авторского права ПП. Например, в законодательстве Украины ПП охраняются с одной стороны как литературное произведение, с другой – как набор инструкций. Для реализации закона по защите авторских прав необходимо иметь механизм, с помощью которого можно идентифицировать создателя ПП.

Задача идентификации ПП производится на основе теории проверки статистических гипотез. При этом помимо статистических свойств авторского кода необходимо знать статистические свойства «шумов» вносимых при взломе ПП. Для этого необходимо знать способы и методы «взлома» ПП.

Взлом программного обеспечения означает создание из исполнительного кода, который был разработан другими программистами без соответственного на то разрешения, библиотеки и/или использование функций для написания собственного ПП. Методы взлома вылились в специфическое направление, которое называется крэкинг (от англ. crack - ломать).

Рассмотрены следующие методы взлома:

а) взлом программы посредством введения правильного регистрационного ключа, полученного нелегальным способом;

б) изменение определённых фрагментов программы в оперативной памяти сразу после её загрузки в эту память, но перед её запуском;

в) использование эмулятора ключа.

Показано влияние методов взлома на свойство программного кода.

БАЗИ ДАНИХ, БАЗИ ЗНАНЬ ТА ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

СИСТЕМА ПОИСКА MIDI-ФАЙЛОВ ПО ЗАДАННОЙ МЕЛОДИИ

А.Ю.Белобородов

Национальный аэрокосмический университет им. М.Е. Жуковского "Харьковский авиационный институт"
(научный руководитель Вовковый А.В.)

Возможности современных компьютеров постоянно расширяются. На сегодняшний день трудно представить себе такой ПК, который умел бы только обрабатывать текстовые данные. Современный ПК ко всему еще и мультимедийный центр.

Сейчас поиск текстовой информации не составляет больших трудностей, если критерием поиска является строка.

Трудности возникают при поиске аудио-, видео- данных.

Первый вариант разрешения данной задачи состоит в том, чтоб каждому мультимедийному файлу «присоединить» текстовое описание. Тогда картинки, звуки, видео можно будет искать путем отбора файлов, опираясь на их «присоединенные» описания. Другим вариантом разрешения данной задачи является написание алгоритмов, способных анализировать файлы по внутреннему их содержанию.

В данном докладе речь пойдет о поиске мультимедийных файлов по содержимому. В работе рассматриваются алгоритмы поиска заданной мелодии в файлах формата MIDI, которые были положены в основу для создания системы поиска MIDI-файлов.

Во вступлении говорится о необходимости и перспективности существования систем подобного рода.

Далее речь пойдет про общее устройство системы поиска, необходимость использования файлов формата MIDI и об особенностях работы каждого из алгоритмов поиска.

Результатом проделанной работы является разработка программной системы поиска MIDI-файлов по содержимому, т.е. по фрагменту искомой мелодии, которая будет выступать в роли критерия поиска.

ПРИКЛАДНІ ДОМЕНИ І ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

ВИЗУАЛИЗАЦИЯ АВТОМАТИЧЕСКИХ ПРОЦЕССОВ УПРАВЛЕНИЯ ИСПЫТАНИЯМИ СИСТЕМ КОСМИЧЕСКИХ АППАРАТОВ

К.С. Дидук

Национальный аэрокосмический университет «ХАИ»
(руководитель И.Б. Туркин)

Современный космический аппарат (КА) имеет развитый комплекс бортовой аппаратуры, объединяемой в системы и подсистемы: управления, телеметрии, связи, терморегуляции и т.п. Все их можно отнести к классу сложных систем, так как они обладают рядом соответствующих признаков (целостность, членимость, связанность, организованность, интегрированность). В процессе жизненного цикла КА значительное внимание уделяют организации испытаний, особенностью которых является разнообразие их видов и типов. В настоящее время прибегают к автоматизации испытаний систем КА. Однако участие человека-оператора остается необходимым. Использование компьютерной визуализации в программном обеспечении испытаний систем КА позволяет упростить контроль оператора над процессом испытаний. Являясь знаковым процессом, компьютерная визуализация непосредственно связана с визуальными языками. В данной работе предлагается визуальный язык, позволяющий представить разные аспекты процесса испытаний. Представим процесс испытаний как набор режимов, проверок и переходов между режимами. Предлагаемый визуальный язык состоит из двух типов диаграмм. Первый тип представляет полную картину процесса испытаний в виде диаграммы состояний, на которой состояниям соответствуют режимы испытаний. Второй тип позволяет представить ход выполнения параллельно функционирующих проверок режима, а также его предысторию. Для предложенного визуального языка построена формальная грамматика на основе расширенной формы Бэкуса-Наура.

Построенную формальную грамматику языка предполагается использовать как в качестве инструкции пользователя, так и при создании программных инструментов.

ДОСЛІДЖЕННЯ В УМОВАХ ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В РОЗПОДІЛЬЧІЙ СИСТЕМІ ДЛЯ XML-ОРІЄНТОВНИХ СУБД

А.С.Анохін

Національний авіаційний університет
(науковий керівник. Оленін М.В.)

Як відомо, останнім часом в Україні дуже зросла потреба у використанні СУБД. Використання MSSQL для поточних задач виконувалося дуже ефективно. Але слід зауважити, що застосовувати можна не тільки СУБД, але можна, навіть, слід не виключати із цього списку XML-орієнтовні СУБД. Причина цьому - малі розміри та швидкісне виконання трансакції при застосуванні XML-документу. На першому етапі моделювання було проаналізовано роботу сучасної реляційної СУБД та XML-СУБД. Порівняння результатів виконання XQUERY запитів в режимі реального часу виявило, що для більш гнучкої роботи із напівструкторованими даними XML-СУБД є більш сприятливою[1]. Підтримка XML в РСУБД реалізується у вигляді деякої надбудови над реляційної моделлю. Але реальна обробка XML-запитів відбувається в реляційному ядрі СУБД. Коли XML-запит доходить до реального виконання, він вже транслював у набір SQL-запитів. Чого не можна сказати про XML-СУБД, де виконується перевірка документу по DTD.

Під час другого експерименту проводилася робота по виявленню здатності виконання XQUERY запитів із документом, що описується не за допомогою DTD Schema. Для цього була використана XML schema (XSD). Це більш потужний засіб, що використовується для описання XML-документу. Але найбільшою перевагою є те, що XML schema написана на язику XML[2]. Третій експеримент було проведено для виявлення змоги роботи XML-СУБД в розподільчій системі із урахуванням доступу багатьох користувачів. Для цього було встановлено XML-СУБД на різних локальних комп'ютерах. Використання написаного WEB-сервісу дало змогу виконувати XQUERY запити в розподільчій системі. Таким чином, застосування нових стандартів дає змогу використовувати XML-СУБД у розподільчих системах і дає змогу у створенні системи управління довготривалими трансакціями (СУДТ) в розподільчій системі.

ПЛАНИРОВЩИК ЗАДАЧ СИСТЕМЫ УПРАВЛЕНИЯ ИНФОРИАЦИОННО-ТЕЛЕКОММУНИКАЦИОННОЙ СИСТЕМОЙ

П.Ф.Можаровский, А.В.Волошин

Национальный технический университет Украины «КПИ»
(научный руководитель Ролик А.И.)

Для обеспечения бесперебойной и эффективной работы информационно-телекоммуникационной системы (ИТС) предложена система управления, реализованная на основе агентского подхода. Подход подразумевает наличие на объектах ИТС специального служебного программного обеспечения (ПО), называемого агентом, решающего задачи мониторинга и управления объектами ИТС, а также обнаруживающего неисправности и оповещающего о них серверную часть служебного ПО. При этом локализация неисправностей рассматривается как случайное событие, а мониторинг и управление являются детерминированными процессами, подлежащими планированию. Для решения системой управления задач планирования разработан планировщик заданий, осуществляющий опрос объектов ИТС с целью получения информации о показателях работы объектов, ее предварительную обработку, а также управление передачей данных на сервер системы управления. Агенты приводят к сравнимым величинам значения параметров функционирования объектов ИТС и осуществляют их начальную оценку в сопоставлении с предыдущими измерениями. Расчет производится по дереву объектов мониторинга и управления, когда с приближением к корню увеличивается значимость данных. Временные интервалы формируются на этапе начальной оценки таким образом, чтобы загрузка каналов связи была минимальной, а точность и частота передаваемых на сервер данных обеспечивала адекватную реакцию на текущее состояние оборудования и ПО объектов ИТС. Уменьшение объема передаваемых данных обеспечивается за счет оценки их значимости агентом. Планировщик реализован на функциях абстрактной библиотеки с использованием механизма рефлексии типов платформы .NET, что обеспечивает гибкость и простоту опроса независимо от характера показателей. Функции абстрактной библиотеки замещаемы и могут содержать предварительные вычисления, несмотря на то, что являются нижним уровнем сбора информации на агенте. Планировщик обеспечивает безопасность опроса, предотвращение зависаний и выполнения небезопасного кода на объектах ИТС.

ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В ЛІКУВАЛЬНИХ ЗАКЛАДАХ УКРАЇНИ

В.В.Олексишин

Національний авіаційний університет
(науковий керівник Поперешняк С.В.)

Багато державних та приватних структур на сьогоднішній день уже мають мережу комп'ютерів з деяким програмним забезпеченням. Але кожне підприємство має власну галузь роботи і знайти необхідні програми для роботи складно. Часто дані установи користуються уже існуючим програмним забезпечення з усіма його недоліками, що проявляються в процесі виконання конкретної роботи.

Лікарні, як одні з державних структур теж користуються комп'ютерами для ведення бухгалтерського обліку, обліку пацієнтів та ін. Однак, як правило, програмне забезпечення, що використовується для для виконання даної роботи морально застаріле, оскільки розроблялось для комп'ютерів з частотою процесора не вище 300 МГц, на яких використовувалась операційна система DOS. Отже можна зробити висновок у доцільності розробки нового програмного забезпечення для лікарень.

У статті розглядається доцільність розробки нового програмного забезпечення, яке повинно підвищити продуктивність праці працівників закладу освіти, та забезпечити більш надійне збереження інформації. Даний програмний продукт передбачатиме заміну величезних архівів історій хвороб на невелику базу даних та каталог файлів. Це зменшить навантаження на працівників відділу реєстрації пацієнтів, та полегшить роботу лікарів, оскільки переглядати історію хвороби, отримуючи дані з сервера зручніше, ніж «розбиратись» в тому, що нерозбірливо написано в картці пацієнта. Передбачається аутентифікація кожного працівника закладу. Відділ реєстрації матиме можливість створювати електронну чергу пацієнтів. Розробку програмного забезпечення передбачається виконати на мові C#. Саме програмне забезпечення матиме вигляд Web-прикладної програми на ASP.NET, з використанням .NET Framework версії 3,5 або 4,0.

СИСТЕМА ЦЕНТРАЛИЗОВАННОГО УПРАВЛЕНИЯ ТЕРРИТОРИАЛЬНО РАЗРОЗНЕННОЙ IT-ИНФРАСТРУКТУРОЙ

М.В.Павлов

Национальный авиационный университет
(научный руководитель Авраменко Е.А..)

Рост коммерческих и государственных организаций ставит задачу удалённого управления их IT-ресурсами. Существующие технологические средства, такие как Microsoft Systems Operations Manager и IBM Tivoli не предоставляют полный набор функций, необходимых системным администраторам в их работе, а именно, визуальный доступ и управление целевой рабочей станцией или сервером, не прекращая сеанс текущего пользователя в системе; функционирование в сетях с каналами связи от 19200 кбит/сек. в пределах глобальной вычислительной сети; целостность передаваемых данных; поддержка нескольких одновременных сеансов связи; совместимость с операционными системами (ОС) семейства Microsoft Windows и Unix.

В данной статье рассматривается система централизованного управления территориально разрозненной IT –инфраструктурой, архитектура которой основана на модели клиент-сервер и состоит из следующих компонентов: центральный сервер (ЦС), клиентский модуль (КМ) и рабочее место системного администратора (АДМ). ЦС обеспечивает маршрутизацию сеансов связи от АДМ к КМ и предоставляет возможность одновременной работы нескольких сеансов связи. КМ состоит компонента VNC (Virtual Network Computing), который отвечает за передачу изображений пользовательского интерфейса обслуживаемой ОС на сетевой порт, и сервиса, который передает данные с сетевого порта на ЦС, обеспечивая целость данных с помощью WCF (Windows Communication Foundation). АДМ позволяет системным администраторам получать доступ к клиентскому компьютеру удалённо, не прекращая при этом сеанс работы пользователя.

Система разработана с использованием технологий, как WCF, .Net 3, языка программирования C# и впоследствии интегрируется в более обширное решение, в задачи которого входит постоянный мониторинг обслуживаемых компьютеров, резервное копирование, контроль аппаратного обеспечения.

СОЗДАНИЕ ИНТЕРФЕЙСА ПУЛЬТА ИНСТРУКТОРА

Ю.М. Рябоконь

Национальный авиационный университет
(научный руководитель Радишевский Н.Ф.)

Пульт инструктора в составе авиационного тренажера выполняет задачи отображения информации о полете, о состоянии тренажера и управляет условиями полета. Основная информация о полете отображается пультом в виде показаний приборов.

Существующие пульты большинства наследуемых тренажеров реализованы аппаратно и изношены в процессе длительной эксплуатации. Новые пульты инструктора создаются, как аппаратно-программные средства на основе современных компьютерных технологий. В качестве аппаратного обеспечения используется персональные или промышленные компьютеры общего назначения. Программное обеспечение (ПО) пульта включает системное ПО (операционные системы, протоколы обмена, драйверы) и прикладное ПО, которое реализует функциональность пульта.

Элементы пульта авиационного тренажера составляют пользовательский интерфейс между тренажером и инструктором. Набор и вид элементов такого интерфейса индивидуальны для тренажера самолета каждого типа, однако существенно похожи. Каждый элемент интерфейса определяется двумя составляющими – типом информации и формой ее представления. Элементы интерфейса пульта разделяют на управляющие, предназначенные для ввода значений(переключатели режимов) и индикаторы, предназначенные для отображения информации (индикаторы скорости, высоты).

В статье для создания элементов интерфеса пульта предлагается использовать язык разметки XAML (eXtensible Application Markup Language) входящий в состав WPF(Windows Presentation Foundation). Использование технологии WPF сосмесно с XAML дает возможность изменять масштабируемость индикаторов без потери качества изображения, а также динамически создавать новые индикаторы и их оформление не используя программирование. Достаточно определить конфиграцию прибора в XML-файле и при запуске приложения прибор будет отображаться.