Visual Studio ALM

Справочные материалы по управлению жизненным циклом приложения

Microsoft® Visual Studio® 2010

Сборник сценариев и описаний

Microsoft®
Visual Studio® 2010

Внедрение гибких практик

Производительность группы может повышаться либо понижаться в зависимости от используемого процесса. Но эффективный процесс упрощает взаимодействие участников группы на всех этапах жизненного цикла, от разработки до эксплуатации, улучшая возможности визуального контроля и согласованность действий всей группы.

Чтобы обеспечить непрерывное наращивание ценности для пользователя, группы могут постепенно переходить к гибким практикам, которые упрощают взаимодействие и сокращают потери.

Подтверждение качества рано и часто

Приложения становятся все более сложными, все большее распространение получает распределенная разработка, внедрение гибких подходов – все это требует от групп разработки и тестирования более тесного сотрудничества, чем когда-либо до этого.

Чтобы выявлять проблемы в ходе разработки, а не когда приложение проходит приемочное тестирование пользователями, группы должны контролировать качество приложений на протяжении всего жизненного цикла.

Переход на современную платформу ALM

Часто приходится внедрять современную платформу ALM на базе уже имеющихся инструментов. Это позволяет при выполнении проектов использовать преимущества современных подходов, не теряя предыдущие инвестиции.

Для достижения большей гибкости, качества и улучшения возможностей визуального контроля над ходом выполнения проекта, группам необходима общая среда взаимодействия, которая разрушит стены между ними.

Содержание

Внедрение гибких практик
Подтверждение качества рано и часто
Переход на платформу ALM

Преобразование данных в осознанные действия
Использование SQL, SharePoint и Visual Studio для управляемой данными разработки — Страница 4

Управление проектами и задачами
Управления проектами и портфелями проектов предприятия на всех уровнях — Страница 16

Согласование действий группы, использующей гибкий процесс
Обеспечение PMO возможностей для работы с группой, использующей гибкий процесс — Страница 25

Гетерогенный процесс управления жизненным циклом приложений
Интеграция Team Foundation Server с типовыми серверными инструментами — Страница 32

Гетерогенный процесс разработки
Поддержка Eclipse и Java с Team Explorer Everywhere — Страница 44

Тестирование производительности и нагрузочное тестирование
Раннее выявление проблем с масштабируемостью через тестирование на протяжении всего жизненного цикла приложения — Страница 48

Устранение ошибки, которую нельзя проследить
Использование возможностей IntelliTrace™ для отладки с ведением журнала — Страница 57

Понимание своих систем
Использование анализа архитектуры для понимания влияния изменений — Страница 64

Упрощение процесса разработки
Использование анализа архитектуры для понимания влияния изменений в коде — Страница 71

Создание качественных приложений
Интегрированный подход к достижению качества в три этапа — Страница 76

Устранение дефектов
Использование IntelliTrace и развернутых отчетов об ошибках для исключения сценария невозможности воспроизведения — Страница 85

Повышение производительности и эффективности тестирования
Управление рабочим процессом планирование-тестирование-прослеживание и оптимизация ресурсов тестирования — Страница 91

Интегрированное решение по управлению виртуальными лабораториями тестирования
Непрерывное развертывание и повышение ROI в лаборатории тестирования — Страница 97

Преобразование данных в осознанные действия

с Visual Studio 2010 и SQL Server 2008 R2

По мере роста объемов и доступности данных руководители предприятий начинают осознавать преимущества возможности доступа к данным там и тогда, когда они необходимы. Глобальная доступность данных означает, что в соответствующий момент специалисты могут преобразовать эти данные в значимую информацию, формируя бизнес-анализ. В данной работе мы продемонстрируем, как с помощью SQL Server 2008 R2 и Visual Studio 2010 создавать решения для преобразования данных в анализ и предоставления этого анализа на рассмотрение заинтересованным сторонам независимо от их положения в структуре организации, но с соблюдением принятых правил по доступу к данным. Соответствующие люди располагают соответствующими данными в соответствующий момент – это и есть новая бизнес-аналитика, основным принципом которой является «анализ бизнес данных для всех».

Get the free mobile app for your phone
http://gettag.mobi

ВОССТАНИЕ ДАННЫХ

Возмем любое крупное Веб-приложение. С большой степенью уверенности можно сказать, что своим успехом оно обязано не только пользовательскому интерфейсу. Хороший пользовательский интерфейс важен, но лишь в сочетании с полезными данными и десятками разнообразнейших интерфейсов для доступа к этим данным через API. Возьмите любой социальный сайт. Вы заходите на него со своего телефона, компьютера или даже мини-приложений на других сайтах! Веб и социальные приложения – это не просто один пользовательский интерфейс; это данные, доступные из множества настроенных соответственно выполняемым задачам приложений, что делает взаимодействие с ними столь захватывающим.

Этот переход от главенства конкретного приложения к превосходству данных имеет место и в мире корпораций. Данные уже не могут быть ограничены одним приложением. Это ограничивает глубину проникновения, так необходимую для деятельности динамической и конкурентной организации. Напротив, современные предприятия требуют открытых информативных источников данных, которые могут легко интегрироваться в новые инструменты – инструменты, обеспечивающие конкурентное преимущество.

По мере роста объемов и доступности данных руководители предприятий начинают осознавать преимущества возможности доступа к данным там и тогда, когда они необходимы. Глобальная доступность данных означает, что в соответствующий момент соответствующие специалисты могут преобразовать эти данные в значимую информацию, формируя бизнес-анализ. Для обеспечения этого руководители должны еще более глубоко внедрять возможности бизнес-аналитики в своих организациях. А ведь всего несколько лет назад о «бизнес-аналитике» и «анализе данных» говорили как о важной, но тайной технологии, предназначенной исключительно для руководителей. Были найдены важные схемы и приняты принципиальные решения. Но принципиальные решения принимаются на всех уровнях организации, и, сохраняя права доступа лишь за руководителями самого высокого ранга, мы лишаем сотрудников, ответственных за принятия решений на более низких уровнях, возможности визуального контроля в реальном масштабе времени, необходимого для реализации требований заказчиков, снижения затрат и развития бизнеса. SQL Server 2008 R2 и Visual Studio 2010 обеспечивают доступ к анализу данных соответствующим специалистам, независимо от их положения в структуре организации, но с соблюдением принятых правил по доступу к данным. Соответствующие люди располагают соответствующими данными в соответствующий момент – это и есть новая бизнес-аналитика, основным принципом которой является «анализ бизнес данных для всех».

ДАННЫЕ → ПРИЛОЖЕНИЯ → ИНФОРМАЦИЯ → ДЕЙСТВИЯ

Специалисты по обработке информации ежедневно принимают решения, оказывающие влияние на успешность их организации. Быстрый доступ к данным обеспечит более быстрое принятие более правильных решений, благодаря чему организация в целом сможет оперативнее реагировать на изменения на рынке.

Но просто доступа к данным не достаточно. Специалисты по обработке информации должны иметь возможность рассматривать данные «под разными углами». Необходима достаточная гибкость, чтобы видеть всю ситуацию в целом, углубляться в конкретные аспекты данных, выявлять шаблоны и причинно-следственные отношения. На практике это может означать просмотр обработанных данных в одном или более приложениях и затем углубление в необработанные данные для определенного анализа. Специалисты по обработке информации должны иметь быстрый доступ к необходимым данным через знакомые интерфейсы, будь то Веб-интерфейс, продукт Office или специальное клиентское приложение.

Рисунок 1 – Переход от данных к действиям

Процесс начинается с данных. Данные должны предоставляться в форматах, удобных для работы с ними, быть защищенными от повреждения или утраты, и доступ к ним должен быть быстрым. Для этого необходимо эффективное хранилище данных, такое как SQL Server 2008 R2.

Приложения должны обеспечивать легкий доступ к общим представлениям данных, создавая условия для анализа. Большинство приложений пишется профессиональными разработчиками в сотрудничестве со специалистами по работе с базами данных. Чтобы создавать высококачественные приложения быстро, требуются мощные инструменты разработки, такие как Visual Studio 2010 в сочетании с SQL Server 2008 R2.

“Наши специалисты располагают достаточными знаниями и опытом, и мы больше не хотели, чтобы они полностью зависели от ИТ-отдела в вопросах формирования и организации отчетов. Необходимо было предоставить им возможность самостоятельного анализа и составления отчетов.”

Стивен Уолкер, архитектор баз данных предприятия, Chevron

Несмотря на то что готовые приложения и отчеты могут обеспечивать достаточно большой объем сведений, часто именно анализ необработанных данных бывает особенно эффективным. Организации предоставляют данные как никогда широкому кругу специалистов, обеспечивая экспертов инструментами, необходимыми для проведения анализа самостоятельно. Возможность работы с данными обеспечивают два условия: легкодоступность данных, предоставляемых в удобных для работы с ними форматах, и наличие мощных удобных в использовании аналитических инструментов.

Как только в распоряжении сотрудников организации оказывается необходимая информация, полученная на основании актуальных и практически применимых данных, они могут уверенно действовать. И эти действия могут обеспечивать грандиозные результаты.

ОСНОВЫ РАБОТЫ С ДАННЫМИ

Создавать ориентированные на данные решения тяжело. Одной из основных причин этому являются частые блокировки данных в базах данных, когда они остаются доступными лишь ограниченному кругу специалистов по работе с базами данных и профессиональных разработчиков. Давайте сначала рассмотрим, как управляют базой данных специалисты по работе с базами данных.

РАСШИРЕННЫЕ ВОЗМОЖНОСТИ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

В ориентированных на дынные приложениях управление инфраструктурой данных может представлять сложности. Эти приложения не только могут содержать значительные объемы данных, часто они объединяют данные, которые располагаются в разных базах данных и, возможно, даже на разных компьютерах. И работа со всеми этими ресурсами должны быть координированной.

Для решения этой проблемы в SQL Server 2008 R2 был введен SQL Server Utility Explorer – инструмент для централизованной виртуализации и управления несколькими серверами, средами и ресурсами базы данных как управляемой группой. Этот инструмент также обеспечивает координацию совместной работы отделов разработки и ИТ посредством панелей мониторинга и представлений, отражающих состояние базы данных и приложения. Регулярное использование Utility Explorer позволяет выявлять узкие места в использовании ресурсов и возможности консолидации, что в результате обеспечивает повышение производительности, упрощение управления и увеличение времени безотказной работы.

УЛУЧШЕННАЯ ЦЕЛОСТНОСТЬ ДАННЫХ: MASTER DATA SERVICES

Одним из самых сложных аспектов в создании доверенного источника данных является поддержание актуальности и согласованности данных. Со временем данные устаревают, появляются ошибочные дублирования, повреждения из-за ошибок пользователей, изменение форматов данных может приводить к несогласованности. Чаще всего требуется создать доверенный поток данных, который может использоваться в рамках организации.

Службы Master Data Services (MDS), появившиеся в SQL Server 2008 R2, обеспечивают целостность критически важных данных с момента их добавления до удаления. С помощью MDS администраторы могут применять правила контроля качества данных, определять рабочие процессы изменения данных и создавать единый доверенный источник данных для множества пользователей. В сочетании с другими возможностями, которые обеспечивают предоставление данных в стандартных форматах, MDS гарантируют доступность стандартизованных данных широкой аудитории определенных пользователей.

УЛУЧШЕННАЯ ЦЕЛОСТНОСТЬ ДАННЫХ: СХЕМА БАЗЫ ДАННЫХ С УПРАВЛЕНИЕМ ВЕРСИЯМИ

Если схемы базы данных хранятся вне системы управления версиями, они быстро теряют синхронизацию с изменениями, вносимыми в код приложения. Особенно сильно это заметно при ветвлении кода приложения для поддержки множества версий одного и того же кода. При внесении изменений непосредственно в базу данных, будь то тестирование или производственная эксплуатация, связь с соответствующей версией кода приложения, использующего базу данных, теряется. Это затрудняет возврат к предыдущей версии приложения, отладку предыдущей версии или понимание причин внесения изменений в схему.

Visual Studio 2010 позволяет реконструировать существующие базы данных с размещением схемы в проекте базы данных с управлением версиями, что создает условия для параллельной разработки. То есть база данных и код приложения рассматриваются как единый модуль с поддержкой версий. Для каждой версии этого модуля создается архитектура, пишется код, выполняется тестирование и развертывание, но при этом приложение и база данных остаются синхронизированными. Это также означает, что схему базы данных можно изменять автономно и развертывать все изменения за одну атомарную транзакцию.

“В прошлом в файловой системе хранились сотни файлов сценариев, как правило, без поддержки версий. Теперь мы получили безопасный управляемый способ хранения исходного кода для всего хранилища данных отчетов, включая все схемы базы данных, пакеты интеграции, определения отчетов, сценарии и утилиты развертывания... Team Foundation Server дает нам гарантии и уверенность в том, что именно было развернуто в ходе того или иного обновления, обеспечивая управляемые изменения.”

Джефф Линч, руководитель ИТ-отдела Gulf Coast Seal

УЛУЧШЕННАЯ ЦЕЛОСТНОСТЬ ДАННЫХ: УПРАВЛЕНИЕ ИЗМЕНЕНИЯМИ СХЕМЫ БАЗЫ ДАННЫХ

Еще один ключевой компонент целостности данных – контролируемое управление изменениями схемы базы данных. И разработчики, и специалисты по работе с базами данных сталкиваются с одной общей проблемой: сравнение двух баз данных, не только с точки зрения схемы, но и данных подстановки. Например, при переводе версии 2 приложения в производственную эксплуатацию схема базы данных, скорее всего, претерпевает изменения, как и данные подстановки, такие как географическое распределение продаж или категории продуктов. Visual Studio 2010 предоставляет инструменты для сравнения схемы и данных двух баз данных, которые также обеспечивают формирование сценариев, необходимых для изменения схемы или данных целевой базы данных соответственно исходной.

Эти сценарии даже могут быть включены в автоматизированную сборку, выполняющуюся по плану или по требованию, что обеспечит автоматическое обновление схем баз данных в средах разработки или тестирования. Ежедневное развертывание изменений базы данных с большой степенью уверенности гарантирует целостность схемы и то, что развертывание этой схемы пройдет по плану.

ДОПОЛНИТЕЛЬНЫЕ ВАРИАНТЫ РАЗВЕРТЫВАНИЯ

Сегодня организации имеют богатый выбор вариантов развертывания: локально, инфраструктура как служба (infrastructure as a service, IaaS) и платформа как служба (platform as a service, PaaS). Это означает, что организации могут развертывать свои базы данных внутри организации, на компьютере, размещенном как образ виртуального диска (VHD) в Azure, или в SQL Azure. Тогда как локальное развертывание и развертывание как IaaS обеспечивает полный контроль над администрированием SQL Server 2008 R2, SQL Azure позволяет организации не заниматься задачами администрирования ИТ и сосредоточиться на создании базы данных.

Для локального развертывания предлагается множество разных редакций SQL Server 2008 R2, включая центр данных (Datacenter) и хранилище параллельных данных (Parallel Data Warehouse), обеспечивающие невероятную масштабируемость и высокую производительность. Для модели предоставления платформы как сервиса приложение может развертываться прямо в SQL Azure. Практически это означает, что развертывание может выполняться либо локально, либо в облаке. Но лишь пара простых изменений в конфигурации – и локальное решение базы данных может быть развернуто или перенесено в SQL Azure и наоборот.

Ключевым компонентом в обеспечении упрощенного развертывания, управления и обновления приложений баз данных, как локальных, так и размещаемых в SQL Azure, является приложение уровня данных (Data-Tier Application, DAC), веденное в SQL Server 2008 R2. DAC определяет всю схему ядра СУБД и объектов экземпляра (таких как таблицы, представления и регистрационные имена), необходимых для поддержки приложения. При использовании этого нового механизма создается единственный файл с контролем версий, который передается администратору базы данных (database administrator, DBA) для простого развертывания либо в локальных центрах данных, либо в облаке.

Visual Studio 2010 и SQL Server 2008 R2 обеспечивают довольно большую гибкость работы с приложениями DAC. Например, DAC могут создаваться из существующих приложений баз данных. Кроме того, есть возможность поддержки, выпуска и развертывания новых и последующих версий. После сборки следующей версии приложения уровня данных этот новый пакет DAC может использоваться как для развертывания нового экземпляра DAC, так и для обновления существующего экземпляра до новой версии. Мастер обновления приложения уровня данных сравнивает схему существующего экземпляра DAC и нового пакета DAC и динамически выполняет действия, необходимые для преобразования имеющегося экземпляра DAC в новую версию DAC. Единый пакет для развертывания одной и той же схемы в SQL Server 2008 R2 или SQL Azure можно создать и в Visual Studio 2010.

ПЕРЕСЕЧЕНИЕ ДАННЫХ И ПРИЛОЖЕНИЙ: СОЗДАНИЕ УСЛОВИЙ ДЛЯ ОСОЗНАННЫХ ДЕЙСТВИЙ

Для перехода к осознанным действиям необходима всеобъемлющая платформа, интегрирующая управление и инструментальную поддержку базы данных, которая позволит разработчикам создавать приложения для обеспечения конечных пользователей информацией там и тогда, когда она им необходима.

ИНТЕГРАЦИЯ РАЗРАБОТКИ БАЗЫ ДАННЫХ В ЖИЗНЕННЫЙ ЦИКЛ ПРИЛОЖЕНИЯ

Одной из основных возможностей Visual Studio 2010 является инструментарий для управления жизненным циклом приложений (Application Lifecycle Management, ALM), с помощью которого группы интегрируют и отслеживают свои действия. Эта настраиваемая поддержка ALM обеспечивается Visual Studio 2010 Team Foundation Server, который поддерживает систему управления версиями, распределение и отслеживание задач, автоматизацию сборки, прослеживаемость требований, отслеживание дефектов, проверку систем и формирование отчетов.

Visual Studio 2010 предлагает администратору базы данных некоторые из возможностей, которыми разработчики пользуются ежедневно. Например, автоматизированная сборка, обеспечивающая регулярную компиляцию, интеграцию, развертывание и тестирование кода, может также включать автоматизированное развертывание базы данных. Расширенные возможности Team Foundation Server 2010 для сборки и развертывания баз данных позволяют выполнять сборку, развертывание и тестирование схемы и изменений базы данных ежедневно вместе с кодом приложения. Это обеспечит более высокий уровень надежности всего приложения в целом.

Visual Studio 2010 также предоставляет специальные инструменты для разработчиков баз данных, способствуя их интеграции в общий процесс разработки и синхронизируя их работу с действиями остальной группы. DBA и другие ИТ-специалисты, занимающиеся эксплуатацией, часто работают практически «вне системы», напрямую внося изменения в базы данных или системы. Несмотря на простоту и распространенность, такой подход значительно увеличивает риски. Последствия ошибки могут быть весьма значительными. Visual Studio 2010 упорядочивает этот рискованный подход через интеграцию DBA и ИТ-специалистов в группу разработки, а также бизнеса и разработки. Отслеживая изменения базы данных соответственно конкретным требованиям и бизнес-задачам, группа разработки базы данных получает возможность видеть, почему и когда были внесены в систему те или иные изменения. Это также позволяет им координировать свои изменения с изменениями, вносимыми разработчиками приложения, что еще более способствует прослеживаемости. Такая интеграция группы – это ключ к сокращению неприятных неожиданностей в ходе производственной эксплуатации и незапланированных работ, а также повышению удовлетворенности пользователей.

Прослеживаемость, обеспечиваемая отслеживанием изменений соответственно требованиям, имеет намного более далеко идущие последствия для всей группы разработки. TFS используется не только разработчиками приложения и базы данных, но также тестировщиками и бизнес-аналитиками. Для нас это означает возможность получения интегрированного представления качества приложения в целом. Десятки встроенных отчетов позволяют отслеживать ход выполнения проекта, качество и риск. На рис. 2 представлен такой отчет для отслеживания трудоемкости или затрат времени, хода выполнения тестирования и количества ошибок для каждого отдельного требования. Инструментарий Visual Studio 2010 ALM делает разработку более надежной, последовательной и быстрой, позволяя при этом легко адаптироваться к изменениям. Эти инструменты могут использоваться с традиционными или гибкими процессами, или любыми промежуточными формами процессов. Инструменты координации совместной деятельности и практическая поддержка в групповых проектах, использующих Team Foundation Server 2010, помогут вашей группе выполнить радикальное изменение и начать поставлять разнообразные сложные ИТ-системы с более глубоким пониманием, управлением и прогнозируемостью.

Stories Overview

Helps you track how far each user story has been implemented. Shows each story's actual number of hours of work remaining and completed, its acceptance test results, and the number of bugs that are linked to each story.

Related Reports

- Bug Status
- Status on All Iterations
- Stories Progress
- Test Case Readiness
- Test Plan Progress

Completed / Remaining

Passed / Failed / Not Run / Active / Resolved

Title	Work Progress: % Hours Completed	Work Progress: Hours Remaining	Test Status: Tests	Test Status: Test Results	Test Status: Bugs
As a new customer, I want to order a meal.	79 %	12	3	22 % 62 %	1 2
As a new customer, I want to get an idea of what DinnerNow offers with a brief glance at the web site.	56 %	11	0		6
As a customer who has completed an order, I want DinnerNow to keep track of my meal preferences so that I can order the meal that I want more quickly in the future.	100 %	0	2	31 % 69 %	2 1
As a returning customer, I want to order one of the meals that I've recently ordered.	81 %	7	2	31 % 69 %	
As a new customer, I want to choose a meal from a specific provider.	60 %	2	0		
As a meal provider, I want to display photos and information about each of my meals so that I can distinguish them from other meal providers.		10	0		
As a meal provider that is a restaurant, I want my meals to link to my restaurant's web page to help attract in-person business.		0	0		
As an event planner, I want to let participants in my event choose meals from DinnerNow.		0	0		
As an event planner, I want to filter the menu to meet my constraints so that I can control the cost of the meals or so that I can offer only meals that are appropriate for the event.		0	0		
As new customer, I want the menu to be limited to those available for delivery to my location.		0	0		1
As a returning customer, I want to be able to override my default location so that I can order from DinnerNow when I'm on the road.		0	0		
As a delivery provider, I want orders to be submitted to my business at least 45 minutes before we pick the order up from the provider so that we can optimize the delivery.		10	0		
As a New User I want to enter my personal information	81 %	7	0		

Рисунок 2 – Сквозная прослеживаемость

МОЩНЫЕ ИНСТРУМЕНТЫ РАЗРАБОТКИ

Как только схема базы данных надежно защищена в системе управления версиями, можно переходить к использованию дополнительных возможностей разработки баз данных, таких как автоматизированный рефакторинг, модульное тестирование базы данных, статический анализ и профилирование взаимодействия слоев. С помощью этих инструментов разработчики базы данных получают возможность быстро изменять, тестировать и проверять существующие и вновь создаваемые базы данных. Ранее такого рода работы были подвержены большому количеству ошибок, и на них уходило много времени. Без эффективных инструментов рефакторинг объекта базы данных был полон опасностей и неожиданных последствий. Теперь с инструментами Visual Studio 2010 базы данных могут уверенно редактироваться и тестироваться.

Рефакторинг позволяет переименовывать объект базы данных или перемещать его в другую схему, автоматически полностью изменяя все объекты базы данных, переименовывая все ссылки базы данных или сервера и даже автоматически заменяя все знаки подстановки в выражениях SELECT.

Формирование данных обеспечивает мощный инструмент формирования данных для тестирования. Это критически важно в случаях, когда в базе данных, находящейся в производственной эксплуатации, хранится информация личного порядка (personally identifiable information, PII). Инструменты формирования данных позволяют создавать данные для тестирования на базе реальных данных, трансформируя при этом PII в данные, которые можно использовать для тестирования. Таким образом, разработчики и тестировщики могут проводить тестирование с использованием реалистичных данных, сохраняя при этом конфиденциальность данных пользователей из базы данных, находящейся в производственной эксплуатации.

Модульное тестирование базы данных обеспечивает средства для создания автоматизированных модульных тестов для тестирования правильности хранимых процедур и пользовательских функций. Эти модульные тесты могут выполняться ежедневно или по требованию с целью проверки отсутствия негативного влияния изменений схемы, данных или бизнес-логики на базу данных.

Статический анализ обеспечивает проверку Transact-SQL на наличие проблем с дизайном, именованием и производительностью. Это способствует проактивному выявлению проблем и обеспечению возможности обслуживания базы данных и поддержания ее производительности на необходимом уровне.

Рисунок 3 - Статический анализ баз данных

“SQL Server 2008 R2 изменил правила игры... Мы можем ошеломить наших пользователей, предоставляя функциональность в зависимости от географического местоположения, что превращает ‘где’ в ‘что’ для их приложений.”

Питер Хаммонд,
президент CyberSavvy

Профилирование взаимодействия слоев показывает, сколько раз приложение выполняло запрос к базе данных или хранимую процедуру, на основании чего можно проанализировать, сколько времени понадобилось ADO.NET для выполнения этого запроса. При тестировании Веб-приложения данные анализируются еще глубже, и можно получить сведения о том, сколько раз вызывалась та или иная страница, время ее обработки и запросы, выполняемые этой конкретной страницей. Можно даже получить текст запросов. Эти сведения будут полезны при реструктуризации кода для ограничения числа обращений приложения к базе данных или определения областей, в которых улучшение производительности могло бы иметь значительный положительный эффект.

УЛУЧШЕННЫЙ TRANSACT-SQL

Во многих случаях логика приложения или проверки реализуется прямо в хранимых процедурах. Это создает очень тесную связь бизнес-логики с хранилищем данных, обеспечивая очень высокую производительность. Но для людей, плохо знакомых с синтаксисом T-SQL, такое прямое взаимодействие с ним может быть пугающим и разочаровывающим из-за отсутствия сложных типов данных.

SQL Server 2008 R2 доработан для упрощения работы с T-SQL. Включен полностью интегрированный полнотекстовый поиск, разреженные столбцы и фильтрованные индексы для оптимизации скорости поиска. Кроме того, Transact-SQL расширен типами данных для управления практически любыми типами данных, включая точные дату и время, XML-данные и даже внешние документы и файлы.

Но, наверное, самым значительным из новых насыщенных типов данных являются пространственные данные, которые позволяют сохранять географические координаты в одном поле и выполнять запросы к ним с использованием улучшенных возможностей T-SQL. С помощью пространственных данных можно представлять географические координаты напрямую и создавать запросы для получения «близлежащих» записей: Выбрать всех пользователей, проживающих в радиусе 100 км от Пользователя А.

Кроме того, для помощи в написании логики T-SQL в Visual Studio 2010 доступен IntelliSense. Это может существенно повысить темпы разработки новых хранимых процедур, пользовательских функций и запросов, особенно для не имеющих достаточного опыта разработчиков баз данных.

УЛУЧШЕННАЯ ПОДДЕРЖКА ОБЪЕКТНО-РЕЛЯЦИОННОГО СОПОСТАВЛЕНИЯ

Заполнение пропасти между схемами реляционных баз данных и объектно-ориентированным дизайном приложений традиционно является непростой задачей. Прямой доступ к базе данных очень чувствительный и требует повторной компиляции при внесении любого самого минимального изменения в структуру базы данных; создание уровня доступа к данным, или даже повторное использование уже существующего, требует больших трудозатрат; и многие решения очень сложно эффективно тестировать с помощью инструментов автоматизированного тестирования.

Visual Studio 2010 и .NET 4.0 предоставляют Entity Framework, улучшенный инструмент объектно-реляционного сопоставления (object-relational mapping, ORM), значительно упрощающий взаимодействие между кодом и базой данных. Прелесть Entity Framework в том, что он поддерживает простое создание моделей данных методом «drag-and-drop», и при этом включает ряд очень мощных возможностей, таких как отложенная загрузка данных, поддержка множества уровней, тестируемые интерфейсы и полная поддержка встроенного языка запросов (language integrated query, LINQ).

Entity Framework также поддерживает различные стили написания кода. Разработчики, предпочитающие сначала создавать модели данных, могут воспользоваться преимуществами графических инструментов и генерировать .NET-классы из существующих схем базы данных. Эти классы уже включают загрузку, поддержку LINQ и другие

Рисунок 4 - Entity Framework с функциональностью «Drag-and-Drop»

преимущества и также могут расширяться напрямую для обеспечения еще более тонкого управления. Разработчикам, предпочитающим начинать с проектирования объектно-ориентированной структуры, Entity Framework позволяет создавать схему базы данных классов .NET, полагаясь на соглашение о конфигурации. Таким образом, разработчики ПО могут создавать и развертывать схемы баз данных без написания T-SQL.

АВТОМАТИЧЕСКОЕ ФОРМИРОВАНИЕ НАСТРАИВАЕМОГО И РАСШИРЯЕМОГО UI

Самый простой способ подключить пользовательский интерфейс (UI) к базе данных – методом «drag-and-drop» перенести запрос к базе данных или хранимую процедуру прямо в дизайнер. Эта функциональность поддерживается всеми основными UI, включая Веб, Silverlight, WebForms и WPF. Но для полнофункциональных ориентированных на работу с данными Веб-сайтов, которые сегодня являются основным ресурсом практически любой организации, предлагается еще более простое решение – ASP.NET Dynamic Data.

Visual Studio 2010 и ASP.NET Dynamic Data могут быть хорошим решением для ориентированных на работу с данными приложений, которые должны предоставлять данные базы данных быстро и просто. Благодаря развитой инфраструктуре

динамических шаблонов и повторному использованию элементов управления данными ASP.NET стала возможной быстрая разработка приложений. Просто зарегистрировав модель данных LINQ to Entity Framework в проекте (что занимает несколько минут), можно создать полнофункциональный Веб-сайт, который поддерживает все операции CRUD (создание, чтение, обновление и удаление) посредством щелчка кнопки. Получаемый в результате сайт и правила преобразования данных в визуальные представления могут настраиваться, предоставляя разработчику абсолютную гибкость. Например, всегда можно представить номер социальной страховки, хранящийся в виде 9-значного поля, в формате с тире (xxx-xx-xxxx).

Люди, имеющие опыт работы с другими инструментами автоматического создания Веб-сайтов, настроены скептически по поводу такой автоматизации, полагая, что тесное объединение UI и данных очень усложняет обслуживание. Но при использовании ASP.NET Dynamic Data UI не всегда подключается напрямую к базе данных. Может применяться идеально сконструированная модель Entity Framework, которая обеспечивает промежуточный слой между приложением и данными, демонстрируя неожиданную мощь и настраиваемость при минимальных усилиях. В самом деле, ASP.NET Dynamic Data отличается высокой гибкостью, поддерживает настраиваемый UI, настраиваемую проверку полей данных и настраиваемое представление данных в виде таблицы.

СТАНДАРТНЫЕ ФОРМАТЫ ДАННЫХ ДЛЯ УПРОЩЕНИЯ ПРЕДСТАВЛЕНИЯ ДАННЫХ: СОЗДАНИЕ УСЛОВИЙ ДЛЯ ОСОЗНАННЫХ ДЕЙСТВИЙ

Встроенные отчеты – замечательное средство предоставления информации в графическом формате, но бизнес-пользователям порой необходимо иметь доступ к необработанным данным, что позволит им использовать их по своему усмотрению. Возможность реализации сценариев «самообслуживания» имеет решающее значение для создания более гибкой организации.

Приемов предоставления данных SQL Server 2008 R2 в форматах, с которыми легко работать, множество, но самым популярным является OData. OData расшифровывается «Open Data» (открытые данные) и является открытым стандартом для совместного использования данных по протоколу HTTP. Он позволяет пользователям и приложениям выполнять доступ к опубликованным через конечную точку OData данным и запрашивать эти данные, используя REST (формат взаимодействия с данными с применением URL). Проще всего предоставить данные через службы отчетов (Reporting Services). По желанию в отчетах, создаваемых службами отчетов SQL Server, необработанные данные могут предоставляться как поток OData. Благодаря этому разработчики и специалисты по обработке информации могут использовать данные, на основании которых был составлен отчет, в собственных приложениях, отчетах и специальном анализе.

Многие приложения и среды разработки уже поддерживают OData. Например, PowerPivot для Excel 2010 может работать с каналом OData, объединять эти данные с данными из других источников и предоставлять конечным пользователям возможность работать с ними произвольным образом. Кроме того, большинство Веб-браузеров уже выполняют доступ к OData напрямую, и у разработчиков имеются библиотеки для работы с ними на платформах Silverlight 4, PHP, Windows Phone 7 и многих других.

Как будет показано в следующем разделе, возможность предоставления данных как канала OData имеет мощные последствия для организаций, заключающиеся в переносе критически важных для принятия решений данных на как никогда глубокие уровни.

КОМПЛЕКСНЫЙ АНАЛИЗ – REPORT BUILDER 3.0, EXCEL, POWERPIVOT И SHAREPOINT

Представление данных в виде отчетов — мощный механизм для их понимания, но он не подходит для экспертов, которые привыкли анализировать данные в динамическом режиме. Имея необходимые данные в сочетании с соответствующим инструментом, эксперт может провести такой анализ, который не обеспечит ни один самый тщательно продуманный встроенный отчет.

В прошлом предоставление конечным пользователям доступа к необработанным данным означало предоставление им разрешений на доступ к хранилищу данных напрямую. Также это означало, что они получали соответствующие ИТ-ресурсы для работы, преобразования и отображения данных.

Сегодня обработка данных конечными пользователями — реальность. Стивен Уолкер, архитектор баз данных предприятия в компании Chevron Data Management Solutions, четко обозначил задачу: «Наши эксперты располагают достаточными знаниями и опытом, и мы не хотели, чтобы они полностью зависели от ИТ-отдела в вопросах формирования и организации отчетов. Необходимо было предоставить им возможность самостоятельного анализа и составления отчетов.»[1]

Microsoft Excel, PowerPivot, Report Server 3.0 и SharePoint – все эти инструменты обеспечивают конечных пользователей мощными средствами сбора, анализа и отображения данных.

1 http://www.microsoft.com/casestudies/Case_Study_Detail.aspx?casestudyid=4000007043

МОЩНЫЕ ВОЗМОЖНОСТИ ПРОИЗВОЛЬНОГО АНАЛИЗА ДАННЫХ

Microsoft Excel всегда был мощным инструментом для анализа данных. Большинство конечных пользователей, занимающихся анализом данных, хорошо знакомы со сводными таблицами для быстрого анализа данных и сводными диаграммами для отображения результатов, а также богатым набором предлагаемых преобразований данных. В последней версии возможности этого почтенного инструмента были еще больше расширены.

Рисунок 5 - PowerPivot в Excel

Самым эффективным дополнением Excel является PowerPivot. PowerPivot расширяет Excel и позволяет пользователям загружать даже самые большие наборы данных практически из любого источника, обрабатывать большие объемы данных в считанные мгновения (при использовании многоядерных процессоров) и применять разнообразные новые аналитические возможности, такие как выражения для анализа данных (Data Analysis Expressions). И все это пользователи делают в так хорошо знакомом интерфейсе Excel..

PowerPivot может также обрабатывать данные в формате OData. То есть из готового отчета, размещенного в службах отчетов, конечные пользователи могут извлечь в Excel данные, на основании которых он был составлен. Потом эти данные могут объединяться с данными из другого источника, например онлайн инструмента для управления отношениями с заказчиками, и анализироваться совместно. Объединяя множество каналов данных, конечные пользователи могут создавать мощные решения, позволяющие взглянуть на данные по-новому.

Наконец, и Excel, и PowerPivot обеспечивают превосходные средства визуализации данных, с помощью которых создаются панели мониторинга, диаграммы и другие представления данных. Благодаря этому новые выводы или анализ можно представить в доступной для пользователей форме.

ПРОСТОТА СОЗДАНИЯ ОТЧЕТОВ ДЛЯ КОНЕЧНЫХ ПОЛЬЗОВАТЕЛЕЙ

Рисунок 6 - Report Builder 3.0 в действии

Визуальные отчеты является одним из самых распространенных способов создания условий для осознанных действий. Как правило, людям намного проще воспринимать и понимать диаграмму или рисунок, чем таблицу данных. Но создать действительно качественное визуальные представления не так просто.

В SQL Server 2008 R2 и Visual Studio 2010 процесс формирования отчетов продвинулся далеко вперед. Кроме имеющихся мастеров отчетов, возможностей OLAP-куба и других, службы отчетов SQL Server (SSRS) теперь поддерживают Report Builder 3.0.

Report Builder 3.0 имеет интерфейс в стиле приложений Office, с помощью которого бизнес-пользователи могут создавать новые отчеты самостоятельно, не обращаясь за помощью в ИТ-отдел. Пользователи задают, откуда взять данные, какие данные использовать, и как отображать эти данные. Для извлечения данных предлагается широкий набор всевозможных источников данных, включая данные из списков SharePoint и SQL Azure. Для выбора данных для отображения предлагаются интерактивные инструменты, такие как построители запросов. Для отображения данных в привлекательном виде предусмотрен комплект средств визуализации с богатой функциональностью, включая геопространственные соотношения и визуальные элементы, спарклайны и общие части отчетов. Кроме того, можно без труда активировать детализацию и обеспечивать пользователей более глубокими сведениями.

ПУБЛИКОВАТЬ АНАЛИЗ СТАНОВИТСЯ ПРОСТО

Возможность быстро и эффективно анализировать наборы данных и превращать их в бизнес-анализ уже сама по себе представляет огромную мощь. Но если к тому же этот анализ может быть без труда распространен в рамках организации и перекомпонован различными способами, он может быть обращен в конкретные действия и его ценность возрастет в разы. Именно перейдя к действиям, бизнес начнет извлекать пользу из этого анализа.

Обычно анализы бизнес-данных распространялись в рамках организации очень медленно. Доставить информацию тем, кому она предназначается, можно было по электронной почте, посредством Веб-сайта или портала. Но с появлением Visual Studio 2010 и SharePoint проблемы распространения канули в лету.

Конечные пользователи, владеющие Excel или PowerPivot, смогут работать с SharePoint 2010 эффективно и без проблем. SharePoint 2010 и службы данных Excel позволяют размещать любые графики или другие визуальные представления Excel прямо на странице SharePoint. То есть конечные пользователи могут создавать порталы для публикации множества разнообразных представлений имеющихся данных на самостоятельно созданной странице SharePoint, не обращаясь за помощью в ИТ-отдел. PowerPivot обеспечивает дополнительную поддержку SharePoint и простой механизм не только размещения визуальных представлений данных, но также и работы с данными прямо на странице SharePoint. Кроме того, PowerPivot для SharePoint поставляется с панелями мониторинга, с помощью которых руководители ИТ-отдела могут проводить мониторинг и отслеживать использование с течением времени. Эта возможность имеет решающее значение, поскольку благодаря ей организация может свободно предоставлять решения PowerPivot для конечных пользователей более широкой аудитории и по истечении некоторого времени видеть, какое из решений пользуется наибольшей популярностью. Таким образом, новые решения могут становиться частью стандартного набора отчетов и решений или даже критически важными сами по себе.

Рисунок 7 - Панель мониторинга для управления PowerPivot

Пользователи Report Builder 3.0 также могут загружать свои решения в SharePoint, делая их доступными через портал. Это обеспечивает практически те же преимущества, что и решения Excel и PowerPivot. Имеется и дополнительный вариант: Visual Studio 2010 включает несколько компонентов средства просмотра отчетов, которые могут выводить на экран отчеты служб отчетов в любом пользовательском интерфейсе, включая Веб, Windows и WPF. Компонент средства просмотра отчетов для Веб обеспечивает ряд мощных точек расширения, включая возможность написания сценария на JavaScript, использования AJAX для асинхронных обновлений данных и множество опций для расширенного управления представлением и функциональностью отчета. Кроме того, он поддерживается многими браузерами, в том числе IE 6 и более поздними версиями, Firefox 3.5 и более поздними версиями и Safari 4. Хотя обычно для этого требуется вмешательство со стороны группы разработки, но компонент средства просмотра отчетов позволяет встраивать созданный конечным пользователем отчет в Веб-страницу или приложение смарт-клиента, которые могут развертываться где угодно.

Будь то Excel, PowerPivot или Report Builder, но никогда до этого не было так просто создавать и развертывать новые решения для анализа данных. И SharePoint 2010, и Visual Studio 2010 обеспечивают конечным пользователям и разработчикам возможности для публикации этих решений конечных пользователей максимально широкой аудитории.

НАЧИНАЕМ УЖЕ СЕГОДНЯ

Инструменты, необходимые для развертывания «комплексного анализа» в организации, уже существуют. Интегрируя возможности и инструменты SQL Server 2008 R2 и Visual Studio 2010, можно уже сегодня создавать решения, трансформирующие данные в анализ и действия. В этом документе было продемонстрировано, как объединение данных и приложений на платформе Майкрософт может обеспечить:

- Более высокое качество данных и приложений
- Надежные инструменты разработки и формирования отчетов, охватывающие всю организацию
- Меньшее время выхода приложений на рынок
- Самостоятельное формирование отчетов

Также было показано, как данные, опубликованные в организации, могут кардинально изменить способ ведения дел. Начните превращать данные в действия уже сегодня.

Больше информации можно найти по адресу
http://www.microsoft.com/visualstudio и http://www.microsoft.com/sqlserver

Microsoft Project Server 2010

Управление проектами и задачами

с Project Server 2010 и Team Foundation Server 2010

Чтобы обеспечить максимальные шансы успешного выполнения проектов, компании должны реализовать проактивное распределение приоритетов, планирование и прослеживание проектов. Но для большинства организаций управление проектами и ресурсами по-прежнему является непрекращающейся борьбой. В Майкрософт знают об этих проблемах между различными уровнями управления задачами и предлагают для их решения надежные инструменты уровня предприятия: Microsoft Project Server 2010 и Microsoft Visual Studio Team Foundation Server 2010. Данный документ представляет мощь управления портфелем, проектом и ресурсами, которая обеспечивается Project Server 2010, и интеграцию с Team Foundation Server 2010, который контролирует процессы разработки. Эта интеграция устраняет пропасть в управлении задачами, которая имеется сегодня между группами, использующими разные методики и процессы, и позволяет организациям использовать преимущества полной прослеживаемости проектов для всего портфеля проектов.

Управление проектами и задачами

Успешные компании поставляют проекты и продукты, соответствующие бизнес-требованиям. Если выполнение бизнес-требований не обеспечивается, в конце концов это приводит к краху компании. Чтобы обеспечить максимальное число успешных проектов, компании должны проактивно расставлять приоритеты, планировать и прослеживать проекты. Несмотря на то что принципы планирования и прослеживания проектов и задач, обеспечивающих эти проекты, просты, систематическое и успешное выполнение проектов является проблематичным для большинства организаций.

Как правило, бизнес-требования определяются руководством предприятия. Эти люди задают направление развития компании и определяют стратегию, которая обеспечит ее успех. Но реализацией этой стратегии обычно занимаются исполнители, находящиеся на несколько ступеней ниже в иерархической структуре компании. Руководитель, вырабатывающий стратегию, исполнитель и сотрудники всех промежуточных между ними уровней выполняют разные задачи для достижения общих целей. Все уровни важны, но очень различны с точки зрения подходов и требований. Именно эти различия и затрудняют планирование, управление и отслеживание задач.

Управление на всех уровнях

Для описания всех уровней процесса управления бизнес-целями через реализацию этих целей отдельными исполнителями используется несколько принятых в отрасли стандартных терминов:

- Управление портфелем проектов (PPM)
- Управление проектом
- Управление предприятием, включая:
  - Управление проектами предприятия (EPM)
  - Управление ресурсами

На каждом из этих уровней управление осуществляется разными ролями с применением разных инструментов и процессов. Организации могут иметь зрелый и развитой процесс на одном уровне, но совершенный хаос на другом. Это «слабое звено» на любом из уровней будет иметь негативное влияние на все остальные уровни и организацию в целом.

УПРАВЛЕНИЕ ПОРТФЕЛЕМ ПРОЕКТОВ

Термин управление портфелем проектов (Project Portfolio Management, PPM) используется для описания методов анализа и управления группой находящихся в разработке или планируемых проектов на основании ряда ключевых показателей. Основная цель PPM – обеспечение возможности анализа всех текущих и планируемых проектов для определения оптимального сочетания и последовательности выполнения для достижения общих целей организации. Целью организации могут служить жесткие экономические меры, стратегия развития бизнеса или стратегия технологического развития. Ограничения, налагаемые руководством или внешними факторами, также должны учитываться. При анализе проекта принимается во внимание множество факторов: затраты, риски, потребности в комплектации штата, стоимость проекта, объем работ и предполагаемые сроки выполнения. Нельзя забывать также о взаимозависимостях и связях с другими проектами.

УПРАВЛЕНИЕ ПРОЕКТОМ

Управление проектом – это комплекс отдельных мероприятий, выполняемых по отношению к одному проекту. К этим мероприятиям, направленным на достижение целей проекта, относится планирование, организация, обеспечение безопасности и управление ресурсами. Внесение изменений и управление рисками должно выполняться на постоянной основе.

УПРАВЛЕНИЕ ПРОЕКТАМИ ПРЕДПРИЯТИЯ

Управление проектами предприятия (Enterprise Project Management, EPM) объединяет в себе все мероприятия по отношению ко всем проектам предприятия. Одновременно на предприятии может выполняться несколько проектов, и в них могут быть задействованы общие ресурсы (сотрудники и др.). Организации, работающей над несколькими проектами одновременно, необходимо отслеживать и управлять мероприятиями и ходом выполнения всех проектов.

УПРАВЛЕНИЕ РЕСУРСАМИ ПРЕДПРИЯТИЯ

Управление ресурсами тесно связано с EPM и обеспечивает необходимость в управлении всеми ресурсами предприятия. Такими ресурсами являются финансовые ресурсы, имущество, квалифицированные сотрудники или консультанты, производственные ресурсы или ИТ-активы.

Организации могут финансировать большое количество разнообразных проектов, и все они управляются тем или иным образом. Любой отдельно взятый проект может быть посвящен строительству, производству, маркетингу, продажам, услугам или любым другим аспектам бизнеса.

Часто управление одной категорией проекта отличается от других. К управлению проектами по разработке приложений исторически применялся такой же подход, как и к любым другим проектам. В прошедшие десять лет разработка ПО и управление разработкой приложений интенсивно развивались. Эта эволюция позволила группам разработки приложений улучшить возможности по успешной доставке программных проектов.

Несмотря на то что современные практики управления проектами разработки приложений обеспечивают создание намного более успешных проектов, они также увеличили пропасть между руководством этими проектами и остальными сотрудниками организации.

Инструменты, которые улучшат управление задачами

Организации могут повышать успешность своих проектов и управления ими, сочетая практики, обучение и соответствующие инструменты. Майкрософт предлагает ряд инструментов, которые непосредственно ориентированы на разные уровни управления и помогут обеспечить эффективную координацию деятельности этих уровней.

MICROSOFT® PROJECT SERVER® 2010

Традиционно Project Server (Сервер проекта) решает проблемы, связанные с управлением проектами предприятия и управлением ресурсами. С выпуском Project Server 2010 теперь обеспечивается и поддержка управления портфелем проектов. Такое сочетание позволяет организациям запускать, выбирать, планировать и управлять проектами на протяжении всего их жизненного цикла.

Предлагаемые для выполнения проекты могут прямо вводиться через Веб-интерфейс Project Server. Project Server 2010 построен на базе SharePoint Server 2010 и, следовательно, обеспечивает полнофункциональный интерфейс для коллективного создания и управления проектами. Когда вносится новый проект, он сравнивается с существующими проектами, управление которых осуществляется с помощью Project Server, и на основании этого сравнения создается необходимый портфель проекта. Оповещение об изменении и управление текущим статусом этих проектов выполняется соответствующим образом.

В случае с проектами по разработке ПО Project Server часто критикуют за обременительность его использования: рабочие задачи проекта по разработке ПО могут меняться ежемесячно, а то и еженедельно или даже ежедневно. Группы разработки ПО начали внедрять практики управления проектами, учитывающие такие динамические изменения задач. Использование Project Server может вызывать сложности при попытках управлять и отслеживать статус проектов, не имеющих структурированного плана проекта. Более подробно об интеграции практик гибкой разработки ПО с более традиционным подходом к управлению проектами рассказывается в материале «Reconciling the Agile Team with Enterprise Project Management» (Согласование гибких практик разработки с управлением проектами предприятия).

MICROSOFT VISUAL STUDIO TEAM FOUNDATION SERVER® 2010

Microsoft® Visual Studio Team Foundation Server® 2010 создавался как узел координации совместной деятельности для групп разработки приложений. В отличие от ориентированного на управление проектами решения задачей Team Foundation Server является обеспечение успешной поставки через управление жизненным циклом приложения. Это может включать управление требованиями, планом выполнения проекта и объемом работ, отслеживание дефектов, автоматизированные сборки, развертывание, тестирование и другие элементы жизненного цикла приложения.

Предлагаемый Майкрософт подход к разработке ПО с применением управления жизненным циклом приложения (Application Lifecycle Management, ALM) позволяет сократить риски, упростить взаимодействие и устранить потери в ходе всего процесса поставки ПО. Применяя этот подход, группы разработки получают возможность работать на разных платформах разработки, как Майкрософт, так и других производителей, обеспечивая поставку стабильно качественного ПО с соблюдением сроков и бюджета. Работающие с Team Foundation Server 2010 группы разработки приложений могут отслеживать задачи как часть жизненного цикла приложения, используя возможность отслеживания рабочих элементов. Рабочие элементы решают не только проблемы классического планирования и отслеживания задач из других типов проектов, но также такие вопросы, как управление дефектами и требованиями.

Team Foundation Server 2010 создавался не как инструмент управления проектами. Team Foundation Server позволяет распределять задачи, управлять ими в ходе их выполнения вплоть до завершения и сообщать об их состоянии. Но Team Foundation Server не обеспечивает управление ресурсами проекта и не учитывает рабочие часы, праздники и

© Корпорация Майкрософт. Все права защищены

отпуска. Например, Team Foundation Server может отслеживать 2000 часов работы до завершения задачи, но только если весь объем работ закреплен за одним исполнителем и должен быть выполнен за неделю. В этом случае с Team Foundation Server не возникает никаких проблем. Традиционная система управления проектами обеспечила бы такие типы перераспределения ресурсов. Системы управление проектами также обеспечивают возможность выявления критического пути через обзор зависимостей между задачами. Team Foundation Server допускает зависимые отношения, но не проводит их анализа.

Одним из вариантов является использование подключаемого модуля Майкрософт для Microsoft Project, который позволяет синхронизировать данные плана проекта с Team Foundation Server. Это обеспечивает простой способ обновления данных. Так, например, невыполненные задачи будут вводиться в Team Foundation Server 2010 и обновляться в плане проекта. Изменения сроков, распределение ресурсов, предшествующие и последующие задачи – все эти данные синхронизируются, и отношения сохраняются. Благодаря этому руководители проектов могут управлять проектом и обновлять статус соответственно реальному состоянию дел.

ПРОПАСТЬ В УПРАВЛЕНИИ ЗАДАЧАМИ

Руководители проектов и отделы управления проектами (Project Management Office, PMO) могут использовать Project Server 2010 для устранения пропасти между высшим руководством, руководителями проектов и ресурсов предприятия и группами отдельных проектов. Project Server проявил себя как незаменимый инструмент для управления ресурсами и сроками.

Группы разработки приложений считают Team Foundation Server 2010 замечательным средством для отслеживания всех мероприятий по производству ПО, включая и традиционно управляемые задачи, и дефекты, которые обычно рассматриваются как незапланированные работы. Team Foundation Server 2010 позволяет без труда управлять, сортировать и переназначать задачи естественным для групп разработки путем с обеспечением необходимой гибкости. Благодаря Team Foundation Server вся группа работки ПО может взаимодействовать в контексте выполняемой задачи или задач, применяя привычные и хорошо знакомые инструменты.

Оба этих инструмента и подхода действительны и применимы, но отличаются по сути. Из-за этой разницы в подходах возникает пропасть между группами разработки, использующими Team Foundation Server, и руководителями проектов и заинтересованными сторонами, которые работают с Project Server. Процессы, механизмы отслеживания, терминология и инструменты, используемые группами разработки, отличаются от тех, которыми оперирует остальная организация.

Любая из сторон, занимающаяся или не занимающаяся разработкой, могла бы использовать инструменты, не оптимизированные для их ситуации; сотрудники могли бы делать записи в обеих несовместимых системах; одна или обе стороны могли бы полностью отказаться от управления и уповать на удачу. Попытка заполнить эту пропасть любым из упомянутых способов приведет к потере продуктивности, плохому и несогласованному взаимодействию, появлению множества ошибок и снижению производительности и морального духа сотрудников.

© Корпорация Майкрософт. Все права защищены

Устранение пропасти в управлении задачами

Инструменты оптимизированы для разных групп пользователей и для управления разными этапами или мероприятиями в процессе разработки ПО. В идеале, каждый пользователь должен иметь возможность участвовать в процессе, работая с привычными ему инструментами. Автоматизация может еще больше сократить время, которое уходит на ввод данных вручную, и количество ошибок, обусловленных человеческим фактором. Интеграция процессов и данных различных систем является ключевым фактором в улучшении процесса управления задачами.

Цель – управлять проектами от начала и до реализации, используя инструменты, отвечающие требованиям использующей их стороны и обеспечивающие полную прослеживаемость и видимость состояния.

Идеальный подход – использовать мощь Project Server и Team Foundation Server в интегрированном рабочем процессе. Этого можно достичь через интеграцию двух систем таким образом, который позволяет группам разработки использовать Team Foundation Server для повышения их продуктивности и качества, обеспечивая при этом предприятию возможность выбирать, управлять и отслеживать проекты с помощью Project Server 2010. Чтобы реализовать этот подход, Майкрософт инвестировали в интеграцию Team Foundation Server и Project Server. Пакет компонентов интеграции Microsoft® Team Foundation Server® 2010 Project Server позволяет тесно интегрировать процессы разработки с требованиями управления проектами предприятия. Данные свободно передаются между системами, благодаря чему группа разработки и другие группы, работающие над проектом, могут использовать соответствующие их нуждам интегрированные инструменты и процессы на всех уровнях. Это включает возможность регистрировать бизнес-цели, определяемые на уровне руководства, и управлять рабочим процессом через отдел управления проектами, передавая задачи вниз отдельным группам, работающим над проектом (включая группу разработки приложения), и возвращая результаты наверх руководству для их сравнения с исходными бизнес-целями.

ПРОСЛЕЖИВАНИЕ В ХОДЕ ВСЕГО ЖИЗНЕННОГО ЦИКЛА

Одной из целей Project Server 2010 было обеспечение возможности визуального контроля и прослеживания на протяжении всего жизненного цикла проекта. Идеи регистрируются в Project Server и доступны всем заинтересованным сторонам в рамках организации. Вводимая группой разработки информация о состоянии проектов по разработке ПО, для которых используется Team Foundation Server, может автоматически использоваться для отчетов о проектах предприятия. Это позволяет сравнивать и обосновывать ожидаемые выгоды для бизнеса и фактические затраты группы. Благодаря этим данным становится возможным ежедневно корректировать управление портфелем проектов и ход реализации проекта. Project Server обеспечивает единообразие в рамках всей организации, что помогает выполнять все проекты с использованием установленных лучших практик, предоставляя при этом гибкость и возможность настройки, необходимые отдельным группам.

ОТ ИДЕЙ К ПРОЕКТАМ

В основе всех проектов лежат идеи. Project Server 2010 предлагает множество возможностей для организаций и среди них возможность сохранять в централизованном хранилище идеи, которые сделают бизнес более успешным. Веб-инструменты, встроенные в Project Server, упростят не только понимание этих идей, но также видение того, как можно их включить в бизнес-цели и задачи.

Project Server делает возможным превратить идеи руководства и разработчиков стратегии в конкретные проекты, которые могут быть реализованы организацией.

МЕЖДУ ПРОЕКТАМИ И ПРЕДПРИЯТИЕМ

После того как проекты определены и расставлены приоритеты, должны быть решены все вопросы с распределением ресурсов. Более приоритетные проекты обеспечиваются всеми необходимыми для достижения успеха ресурсами. Project Server поддерживает процесс управления ресурсами предприятия, позволяя понять, как критически важные ресурсы одного проекта будут влиять на другие проекты. Используя эти сведения, можно проактивно управлять проектами, обеспечивая продвижение вперед соответствующих проектов. Также это поможет понять имеющиеся ограничения и то, какие ресурсы на самом деле нужны, чтобы перераспределять и перенаправлять их для максимальной выгоды бизнеса.

МЕЖДУ ПРОЕКТАМИ И ГРУППАМИ РАЗРАБОТКИ ПРИЛОЖЕНИЙ

Индустрия разработки ПО стала более зрелой, появились новые методики управления задачами в проектах по разработке ПО. Эти методики, несмотря на то что доказали свою успешность, часто не придерживаются классической структуры проекта и имеют более итеративную природу. Принуждение групп разработки, не использующих классическую методику управления проектами, к применению классических инструментов и практик управления проектами может иметь негативное влияние на проекты разработки, снижая качество и повышая риски проекта. Но с Team Foundation Server и отслеживанием рабочих элементов группы разработки могут рационально и эффективно выполнять проекты по разработке ПО.

Пакет компонентов интеграции Microsoft® Team Foundation Server® 2010 Project Server делает возможным обмен данными между руководством проекта и группой разработки приложения. Типовой сценарий, обеспечиваемый интеграцией Team Foundation Server и PS, мог бы включать:

1. В Project Server, используя Project Web Access (PWA) 2010 или Project Professional 2010, руководитель проекта фиксирует требования проекта и/или укрупненные задачи.
2. Группа разработки приложения получает эти требования в Team Foundation Server и с помощью Visual Studio разбивает их на более мелкие задачи. Оценки для этих задач автоматически суммируются и публикуются назад в Project Server.
3. Руководитель проекта просматривает предлагаемый перечень работ по видам, проводит оценку и утверждает их для включения в план проекта.
4. На протяжении всего проекта группа разработки приложений обновляет рабочие элементы в Visual Studio; эти изменения дублируются в Project Server для просмотра и утверждения руководителем проекта. Изменения, вносимые руководителем проекта, автоматически копируются в Team Foundation Server.
5. Руководители проекта и заинтересованные стороны могут просматривать проекты в Интернете, используя встроенные в Project Server отчеты и панели мониторинга. Встроенные в Team Foundation Server отчеты и панели мониторинга обеспечивают участников группы разработки приложений и других групп детальными сведениями о состоянии проекта в ходе жизненного цикла приложения.

В этом сценарии все участвующие стороны используют инструменты, соответствующие выполняемым ими задачам. Разработчики и тестировщики работают с семейством инструментов Visual Studio, с помощью которых могут без труда составлять отчеты и отслеживать состояния. И для этого им не приходится переключать контекст: покидать привычный инструмент и переходить в другие инструменты для управления проектом и выполнения других задач в Project Professional или Project Web Access. Изменения, вносимые любой из групп, автоматически синхронизируются во всех системах. Детали, значимые для группы разработки приложений, автоматически группируются и отслеживаются на соответствующем уровне группой управления проектом.

© Корпорация Майкрософт. Все права защищены

Заключение

Управление задачами – необходимое условие для обеспечения успеха и процветания бизнеса. Однако существует множество разных уровней управления, каждый из которых отличается целями, используемыми процессами и инструментами. В Майкрософт обратили внимание на проблемы с разными уровнями управления задачами и разработали для их решения два надежных инструмента уровня предприятий: Microsoft Project Server 2010 и Microsoft Visual Studio Team Foundation Server 2010. С выходом пакета компонентов интеграции Microsoft® Team Foundation Server® 2010 Project Server Integration Feature Pack организации на всех уровнях получили возможность эффективно сосредоточиться в соответствующих областях, используя подходящие инструменты и имея возможность беспрепятственного взаимодействия.

Ресурсы

ИНТЕГРАЦИЯ TEAM FOUNDATION SERVER И PROJECT SERVER 2010

- Решения для управления жизненным циклом приложения: http://www.microsoft.com/visualstudio/en-us/solutions/management
- Документация по продукту на сайте MSDN: http://msdn.microsoft.com/library/gg455680.aspx

РЕСУРСЫ MICROSOFT PROJECT 2010:

Информация по продукту

- Сайт продукта Project 2010: http://www.microsoft.com/project
- Блог группы разработки продукта: http://blogs.msdn.com/project
- Примеры внедрения: http://www.microsoft.com/project/en/us/customer-success.aspx

Справка по продукту для конечных пользователей

- Справка по Project 2010: http://office2010.microsoft.com/project-help
- Справка по Project Server 2010: http://office2010.microsoft.com/project-server-help

Интерактивные ресурсы – видео, выступления и Веб-трансляции

- http://www.microsoft.com/showcase/en/US/channels/microsoftproject
- http://www.microsoft.com/events/series/epm.aspx

Образ виртуального диска с демонстрацией Project Professional 2010 и Project 2010 Demo Image:

- Загрузка: http://go.microsoft.com/?linkid=9713956
- Внешняя виртуальная лаборатория тестирования: http://go.microsoft.com/?linkid=9713654

Для ИТ-специалистов - TechNet

- Центр технической поддержки: http://technet.microsoft.com/ProjectServer
- Блог по администрированию: http://blogs.technet.com/projectadministration

Для разработчиков - MSDN

- Центр разработчиков: http://msdn.microsoft.com/Project
- Блог, посвященный возможностям программирования: http://blogs.msdn.com/project_programmability

Появились вопросы? Находите ответы или спрашивайте на официальных форумах Майкрософт!

- Официальные форумы Майкрософт: http://social.msdn.microsoft.com/Forums/en-US/category/projectserver2010,projectprofessional2010/

Продукты SharePoint 2010

- http://sharepoint.microsoft.com

MICROSOFT VISUAL STUDIO TEAM FOUNDATION SERVER 2010

- Страница Team Foundation Server на сайте MSDN: http://msdn.microsoft.com/en-us/vstudio/ff637362.aspx
- Ознакомительная версия Microsoft Visual Studio Team Foundation Server 2010 – ISO: www.microsoft.com/downloads/details.aspx?FamilyID=3660cacf-f077-44d3-a9d9-97e801da2035&displaylang=en
- Блог группы разработки Team Foundation Server: http://blogs.msdn.com/b/team_foundation/
- Блог Брайана Харри: http://blogs.msdn.com/b/bharry/
- Видеоматериалы по Team Foundation Server на Channel 9: http://channel9.msdn.com/tags/TFS/

© Корпорация Майкрософт. Все права защищены

Microsoft Project Server 2010

Согласование действий группы, использующей гибкий процесс

посредством управления проектами на уровне предприятия

Результатом успешных проектов по разработке ПО является создание уникальных ценных приложений, обеспечивающих конкурентное преимущество. Сложности возникают, если для управления разработкой ПО и другими мероприятия по проекту в организации используются традиционные методики, обычно PMO, но при этом некоторые проекты по разработке ПО выполняются группами разработки, использующими гибкие подходы. Team Foundation Server 2010 обеспечивает все возможности применения подходов гибкого управления проектами, тогда как Project Server 2010 может использоваться для управления портфелем проектов предприятия, проектами и ресурсами с помощью более традиционного процесса водопада. Данный документ описывает, как могут выиграть организации от интеграции этих двух продуктов, что обеспечит актуальный анализ хода выполнения портфеля проектов, синхронизацию с основными целями и эффективное использование ресурсов благодаря автоматизации обмена и публикации сведений о проекте и улучшению координации деятельности групп, использующих разные методики.

Разработка ПО на предприятии

Программные системы занимают все более важную роль в управлении бизнес-процессами предприятий. Бронирование билетов или перевод денег на банковский счет – все это теперь осуществляется с помощью ПО. Успешные проекты по разработке ПО обеспечивают создание приложений, которые улучшают бизнес-процессы предприятия. Эти построенные на ПО бизнес-процессы повышают рентабельность и конкурентоспособность предприятий. Более подробно о процессах разработки ПО рассказывает статья «What is ALM?» (Что такое ALM?), которую можно найти по адресу http://www.microsoft.com/visualstudio/en-us/solutions/management

Понимание подходов к разработке приложений

Традиционные методики управления проектами требуют глубокого понимания объема работ по проекту, а также сроков выполнения и взаимозависимостей задач. Это крайне выгодно для проектов с фиксированной стоимостью и проектов, для которых стоимость переработки высока и не составляет труда оценить время, необходимое для завершения задач. Со становлением зрелости индустрии разработки ПО были внедрены итеративные подходы к управлению, которые допускают включение изменений, неизбежно возникающих в ходе проекта. Применение этих итеративных методик, названных «гибкими», способствует сокращению объемов переработки, повышению качества и сокращению времени выхода на рынок. Их общей концепцией является приоритизация поставленных задач и выбор подмножества задач, требующих немедленного выполнения. При этом анализ менее приоритетных задач откладывается на более позднее время. Сложности возникают, когда разработка ПО и другие мероприятия по проекту в организации управляются с применением традиционного подхода, но при этом некоторые проекты по разработке ПО выполняются по гибким методикам.

Требования предприятий к разработке приложений

На предприятиях предпочитают выбирать, финансировать и управлять всеми проектами по одной схеме. Для бизнеса исключительно важно иметь возможность проводить оценку состояния проектов и эффективно управлять изменениями и расходами. Поэтому предприятия хотят управлять проектами по производству ПО таким образом, чтобы отслеживать ход выполнения и общие затраты и сравнивать их с ожидаемыми. Эти данные могут использоваться для обоснования дальнейших инвестиций или отказа от реализации той или иной функциональности в создаваемом приложении. Также должна быть доступна информация и прогнозы об использовании ресурсов. Это позволяет планировать новые проекты на основании данных о распределении нагрузки для текущих проектов.

Практики управления проектами предприятия

Для обеспечения максимального успеха предприятия должны эффективно управлять своими инвестициями. Они должны приоритизировать проекты соответственно бизнес-целям, вырабатывать общие планы проектов с возможностью их детализации для участвующих групп. Возможность сообщать сотрудникам о распределении задач и отслеживать ход выполнения проекта является критически важной для эффективного управления проектами предприятия. Общая последовательность выполнения проекта такова:

1. Предложение идей для проекта
2. Проведение анализа проекта с точки зрения следующих аспектов
    a. Значимость
    b. Соответствие бизнес-целям
    c. Стоимость
    d. Риски
    e. Необходимые ресурсы и их наличие
3. Выбор проекта
4. Распределение ресурсов
5. Запуск работ
6. Отслеживание хода выполнения
7. Определение приоритетности проекта в сравнении с другими бизнес-приоритетами

© Корпорация Майкрософт. Все права защищены

Управление проектами предприятия с Project Server 2010

Microsoft Project Server 2010 – готовое решение для реализации всех требований организаций по управлению проектами предприятия. Будучи построенным на базе Microsoft SharePoint 2010, Project Server 2010 предлагает полнофункциональный Веб-портал для координации совместной работы над проектами в рамках организации. Выбор и управление портфелем проектов сочетается с управлением ресурсами и проектами предприятия. Прямо в Project Server 2010 вводятся заказы на выполнение проектов и создаются планы для выбранных проектов. Просмотр наличия ресурсов для всех рассматриваемых проектов осуществляется посредством отчетов, таким образом, планирование проектов может осуществляться с учетом ограничений по ресурсам.

Гибкие практики для успешной реализации приложений

Изменение требований и приоритетов представляет проблему для любого проекта. Чтобы обеспечить возможность таких изменений, была выработана новая методика управления задачами проектов по разработке ПО. Выяснилось, что группа разработки, постоянно тесно взаимодействующая с бизнес-группой, намного быстрее реагирует на изменение требований, если не ограничена никаким статическим планом проекта. Итеративные подходы к разработке ПО существуют десятилетиями и берут свое начало с модели спиральной разработки, описанной Барри Боэмом. В последние годы эти разнообразные методики итеративного управления проектами получили название «гибкие». Гибкий процесс формально описан в документе «Manifesto for Agile Software Development» (Манифест гибкой разработки ПО), опубликованном в феврале 2001 года. Основная цель гибких методик – обеспечить группе разработки возможность регулярного общения с бизнес-пользователями, сделать это общение беспрепятственным и создать условия для беспроблемной адаптации к меняющимся требованиям.

Гибкие практики разработки приложений

Скрам – одна из гибких методик управления проектами, получившая значительную поддержку в последние годы. Скрам реализует высоко итеративный подход к управлению проектами, при котором требования заказчика пересматриваются и повторно приоритизируются с регулярной периодичностью. Такой итеративный пересмотр критически важен для достижения успеха при использовании гибких практик разработки ПО. При использовании скрам все предлагаемые для выполнения задачи проекта по разработке ПО помещаются в один большой перечень невыполненных работ. Задачи выполняются в ходе итераций равной продолжительности, называемых спринтами. В ходе каждого спринта выполняется лишь часть задач перечня невыполненных работ. Например, если для проекта по разработке ПО в перечне невыполненных работ запланировано 20 задач, эти задачи приоритизируются на основании бизнес-требований и загруженности группы, заданной для каждого отдельного спринта. По завершении спринта выполняется пересмотр приоритетов задач перечня невыполненных работ, и в ходе следующего спринта выполняются самые значимые из оставшихся в перечне невыполненных работ задачи. Таким образом, приоритеты задач пересматриваются после каждого спринта, что обеспечивает группам разработки большую гибкость для реагирования на изменение бизнес-требований. Только запланированные для данного спринта задачи исследованы достаточно, чтобы точно установить объем работ. Из-за этого очень сложно планировать долгосрочные проекты и устанавливать точные даты завершения работ.

Гибкая разработка приложений с Team Foundation Server

Team Foundation Server поддерживает гибкие подходы к управлению проектами, скрам в том числе. При настройке группового проекта в Team Foundation Server выбирается процесс разработки ПО. По умолчанию можно выбрать MSF для гибкой разработки ПО или MSF для улучшения процесса CMMI. Оба этих процесса допускают итеративное управление проектом. По желанию можно загрузить и использовать дополнительные процессы разработки ПО. Можно также создать собственный пользовательский процесс. Майкрософт поставляет Microsoft Visual Studio Scrum 1.0, шаблон процесса скрам для Team Foundation Server, который можно загрузить по адресу http://msdn.microsoft.com/en-us/vstudio/aa718795.aspx. Используя рабочие элементы Team Foundation Server, можно создать перечень невыполненных работ по продукту, определить приоритеты элементов и затем определить спринт. Такой подход очень эффективен для разработчиков в Visual Studio и Eclipse, поскольку предлагаемая с Team Foundation Server прямая интеграция позволяет просматривать и редактировать перечень невыполненных работ прямо в среде разработки. Выполнив поставленную задачу, разработчик отмечает соответствующий ей рабочий элемент как завершенный. На основании достигнутой производительности отчеты о выполнении работ (рис. 1) обеспечивают прогноз вероятности завершения оставшихся элементов в ходе спринта .

Рисунок: Отчет о выполнении работ, предоставляемый Team Foundation Server

Гибкий процесс и предприятие – лучшее двух миров

С организационной точки зрения исключительно важно видеть, как проект разработки поддерживает бизнес-требования, и иметь возможность сообщать о ходе выполнения работ и использовании ресурсов. Team Foundation Server и Project Server 2010 позволяют интегрировать данные так, что группа разработки, использующая гибкие подходы, получает возможность с помощью Team Foundation Server максимально повысить свою производительность, при этом предоставляя отчеты на уровне проекта.

Интеграция EPM и инструментов гибкой разработки

Инструменты интеграции Team Foundation Server 2010 и Project Server позволяют извлекать данные из Team Foundation Server и синхронизировать их с Project Server 2007 или Project Server 2010. Благодаря этому на сервере проектов формируются отчеты, в то время как группа управляет своим спринтом в Team Foundation Server. Элементы перечня невыполненных работ по продукту могут вводиться либо в Team Foundation Server, либо в Project Server, и данные синхронизируются. Это позволяет владельцам продукта использовать привычные средства работы с группой разработки. Назначение приоритетов и управление перечнем невыполненных работ по продукту выполняется в Team Foundation Server. После приоритизации рабочие элементы из перечня невыполненных работ могут быть внесены в перечень невыполненных работ спринта, и группа разработки приступает к работе.

© Корпорация Майкрософт. Все права защищены

Управление гибким процессом и проектом предприятия в действии

Управлять взаимодействием пользователей Project Server и пользователей Team Foundation Server относительно просто. Инициируется проект в Project Server. Когда проект выбран и создан в Project Server, можно вводить укрупненные элементы перечня невыполненных работ по продукту. В Team Foundation Server создается соответствующий групповой проект, и в результате синхронизации в Team Foundation Server создается перечень невыполненных работ. Составление графика выполнения задач или их разложение на подзадачи выполняется не в Project Server. Группа, использующая гибкий процесс, определяет приоритеты задач в Team Foundation Server в Excel, Team Web Access, Team Explorer или любом другом инструменте. Как только задачи приоритизированы, выбираются рабочие элементы для первого спринта и работа начинается. Элементы перечня невыполненных работ спринта могут разбиваться на подзадачи, как угодно и удобно группе для управления выполняемыми работами. Рабочие элементы, создаваемые в результате разбиения более высокоуровневых элементов, должны включать поля оставшаяся работа и выполненная работа, которые будут синхронизироваться с Project Server. Время, затрачиваемое на выполнение подзадач, суммируется. Регистрируется и синхронизируется с Project Server только суммарное время и состояние рабочего элемента перечня невыполненных работ. Эта синхронизация обеспечивает возможность просматривать в Веб-приложении проекта данные о состоянии (активный, блокированный, пр.), ходе выполнения (какие задачи осталось выполнить) и завершенности (сделано или не сделано) работ. Такая прозрачность и прослеживаемость обеспечивают возможность визуального контроля, так необходимую бизнес-подразделениям и отделу управления проектами.

Управление гибким процессом и ресурсами предприятия

Организациям также необходимо иметь возможность отслеживать загруженность ресурсов, что позволит им видеть, когда ресурсы освобождаются, и назначать им новые задачи, соответственно. В Project Server эти данные обрабатываются без труда. А вот для группы, использующей гибкий процесс, это может представлять определенные сложности. После завершения текущего спринта информация об отдельных задачах, выполняемых участником группы, не сохраняется. В группах, использующих гибкий процесс, участники закрепляются за группой либо как постоянные, либо как внештатные сотрудники. Для планирования ресурсов предприятия задачи должны создаваться в Project Server, чтобы обозначить их назначение группе, использующей гибкий процесс.

Рассмотрим сценарий предстоящего 4-недельного спринта, описанный в Project Server. Для регистрации распределения участников для данного спринта будут созданы задачи. Для каждого участника группы должны создаваться отдельные задачи для каждого спринта. Эти избыточные данные обеспечивают предприятие средствами, позволяющими отчитываться о распределении кадровых ресурсов. А это, в свою очередь, облегчает планирование будущих проектов и потребностей в найме сотрудников. Как показано ниже, если просматривать данные о доступность ресурсов в Project Server, Анна и Павел будут показаны как загруженные работой неполностью, т.е. они могут быть привлечены для выполнения задач по другим проектам.

Имя	Роль	Загруженность	Общее кол-во часов	Задача
Анна	Тестировщик	75%	120	продолжительность 4 недели, 75%
Дейв	Разработчик	100%	160	продолжительность 4 недели, 100%
Колин	Разработчик	100%	160	продолжительность 4 недели, 100%
Пол	Разработчик БД	50%	80	продолжительность 4 недели, 50%

Сами участники группы используют отслеживание рабочих элементов в Team Foundation Server для распределения и управления своими задачами для спринта. Эти элементы могут включаться или не включаться в план проекта.

Заключение

Планирование и анализ делают возможным реализацию желаний и целей предприятия, при этом позволяя группам разработки работать в наиболее эффективном для их продуктивности режиме. Интеграция Team Foundation Server 2010 и Project Server обеспечивает беспрепятственную интеграцию данных между Team Foundation Server и Project Server. Этим обеспечиваются следующие ключевые возможности:

- Актуальный анализ хода выполнения портфеля проектов, использования ресурсов в проектах разработки ПО и синхронизацию с основными целями путем применения количественных данных, хранящихся в разных системах.
- Автоматизацию обмена и публикации данных проекта для групп и улучшение координации совместной работы групп, использующих разные методики, такие как водопад и гибкий процесс, через общие данные и согласованные показатели.
- Использование группами разработки и управления проектом привычных инструментов, таких как Microsoft Visual Studio, Project и SharePoint, для координации совместной деятельности и публикации графика реализации и хода выполнения проекта.

На основании этих задач по управлению портфелем проектов, проектами и ресурсами уровня предприятия формируются цели для группы разработки. При этом группа разработки предоставляет данные о состоянии для составления отчетов о ходе выполнения проектов предприятия. Такой непрерывный поток информации обеспечивает максимальную эффективность проектов по разработке ПО в рамках организации.

Ресурсы

Ресурсы пакета интеграции Team Foundation Server и Project Server 2010

- Решения для Application Lifecycle Management: http://www.microsoft.com/visualstudio/en-us/solutions/management
- Документация по продукту на сайте MSDN: http://msdn.microsoft.com/library/gg455680.aspx

Ресурсы Microsoft Project 2010:

Информация по продукту

- Сайт продукта Project 2010: http://www.microsoft.com/project
- Блок группы разработки Project 2010: http://blogs.msdn.com/project
- Примеры внедрения: http://www.microsoft.com/project/en/us/customer-success.aspx

© Корпорация Майкрософт. Все права защищены

Microsoft Project Server 2010

Справка по продукту для конечных пользователей

- Справка по Project 2010 http://office2010.microsoft.com/project-help
- Справка по Project Server 2010 http://office2010.microsoft.com/project-server-help

Интерактивные ресурсы – видео, выступления и Веб-трансляции

- http://www.microsoft.com/showcase/en/US/channels/microsoftproject
- http://www.microsoft.com/events/series/epm.aspx

Образ виртуального диска с демонстрацией Project Professional 2010 и Project 2010:

- Загрузка: http://go.microsoft.com/?linkid=9713956
- Внешняя виртуальная лаборатория тестирования: http://go.microsoft.com/?linkid=9713654

Для ИТ-специалистов - TechNet

- Цент технической поддержки: http://technet.microsoft.com/ProjectServer
- Блог по администрированию: http://blogs.technet.com/projectadministration

Для разработчиков - MSDN

- Центр разработчиков: http://msdn.microsoft.com/Project
- Блог, посвященный возможностям программирования: http://blogs.msdn.com/project_programmability

Появились вопросы? Находите ответы или спрашивайте на официальных форумах Майкрософт!

- Официальные форумы Майкрософт: http://social.msdn.microsoft.com/Forums/en-US/category/projectserver2010,projectprofessional2010/

Продукты SharePoint 2010

- http://sharepoint.microsoft.com

Ресурсы Microsoft Visual Studio Team Foundation Server 2010

- Страница Team Foundation Server на сайте MSDN: http://msdn.microsoft.com/en-us/vstudio/ff637362.aspx
- Ознакомительная версия Microsoft Visual Studio Team Foundation Server 2010 – ISO: www.microsoft.com/downloads/details.aspx?FamilyID=3660cacf-f077-44d3-a9d9-97e801da2035&displaylang=en
- Блог группы разработки Team Foundation Server: http://blogs.msdn.com/b/team_foundation/
- Блог Брайана Харри: http://blogs.msdn.com/b/bharry/
- Видеоматериалы по Team Foundation Server на Channel 9: http://channel9.msdn.com/tags/TFS/

Гетерогенный процесс управления жизненным циклом приложений

с Visual Studio 2010

Ежегодно на разработку ПО тратятся миллиарды долларов. Основная масса сведений о процессе разработки складируется в разрозненных системах, лишая организации возможности анализировать свои риски, формировать объективные отчеты о ходе выполнения проектов и вынуждая их тратить бессчетное количество часов на сведение данных, чтобы получить лишь смутное представление о процессе в целом. Инструменты для сквозного управления жизненным циклом приложения, такие как Microsoft Visual Studio 2010 и Team Foundation Server 2010, помогают визуализировать весь процесс разработки, обозначить возможности для изменения, вскрыть риски и обеспечить руководство сводным представлением, объективным, актуальным и точным.

Ежегодно на разработку ПО тратятся миллиарды долларов. Основная масса сведений о процессе разработки складируется в silos, лишая организации возможности анализировать свои риски, формировать объективные отчеты о ходе выполнения проектов и вынуждая их тратить бессчетное количество часов на сведение данных, чтобы получить лишь смутное представление о процессе в целом. Инструменты для сквозного управления жизненным циклом приложения, такие как Microsoft Visual Studio 2010 и Team Foundation Server 2010, помогают визуализировать весь процесс разработки, обозначить возможности для изменения, вскрыть риски и обеспечить руководство сводным представлением, объективным, актуальным и точным.

ПРЕИМУЩЕСТВА СКВОЗНОГО УПРАВЛЕНИЯ ЖИЗНЕННЫМ ЦИКЛОМ ПРИЛОЖЕНИЯ

Сквозное управление жизненным циклом приложения обещает предоставить практически применимую возможность визуального контроля всего жизненного цикла производства ПО, обеспечивая прогнозируемость, управление проектом, управление рисками и повышение качества. Также это позволяет получать объективные данные, на основании которых можно строить стратегии по улучшению процесса.

Для этого множество инструментов должны интегрироваться в единый целостный рабочий процесс, который охватывает весь жизненный цикл приложения и включает такие артефакты: создание идеи, сбор и управление требованиями, проектирование, написание кода, тестирование, сборка и развертывание, управление выпуском и управление внесением изменений после выпуска версии.

Сквозной ALM обеспечивает эти преимущества через отслеживание и формирование отчетов о состоянии артефактов ALM по мере их прохождения по процессу. Рассмотрим простой пример прослеживания требований. Требования разбиваются на задачи, выполняется и фиксируется декомпозиция работ. В ходе работы отслеживается ход выполнения проекта. Выполняются тестовые случаи, и регистрируются ошибки для каждого требования, что позволяет оценивать качество описания требований. Выполняются сборки и их развертывание в тестовых средах и затем в среде производственной эксплуатации, проводится проверка требований, которые были поставлены. Одним словом, даже этот простой пример прослеживаемости показывает, что мы можем эффективно в любой момент оценить выполненную работу и качество реализации требований на текущий момент.

Это лишь небольшой пример. Если сбор данных ALM ведется на комплексной расширяемой платформе ALM, возможен намного более глубокий анализ. Скажем, конкретные объективные данные могут вскрыть зависимость между большой интенсивностью работ и низким качеством. Сквозное управление жизненным циклом приложения является мощным инструментом, обеспечивающим улучшения широкого спектра бизнес и технических мероприятий, и эти улучшения могут быть выражены количественно.

Рисунок 1 - Прослеживаемость требований от выполнения задач до контроля их качества

К сожалению, сквозное управление жизненным циклом приложения затруднено применением огромного числа разрозненных инструментов ALM. Сегодня многие популярные инструменты ALM существуют только в виде разрозненных систем, т.е. их данные недоступны для использования в сквозном представлении. Чтобы реализовать все преимущества сквозного ALM, эти инструменты необходимо либо заменить, либо интегрировать в комплексное решение.

Но даже несмотря на все эти сложности, организации должны внедрять интегрированное решение ALM, благодаря которому все участники разработки ПО могут публиковать процесс для разных проектов и платформ. Единое унифицированное представление обеспечит возможность формирования отчетов на протяжении всего жизненного цикла приложения, позволит управлять рисками глобально и выполнять улучшения процесса на основании реальных данных.

ВВЕДЕНИЕ В ГЕТЕРОГЕННЫЙ ПРОЦЕСС ALM

Для современных организаций, занимающихся разработкой ПО, жизнь редко бывает простой. На большинстве предприятий используется ряд устаревших систем, которые необходимо интегрировать, передовые инструменты, которые требуется включить, а также разнообразнейшие платформы и операционные системы, которые должны быть учтены. Во многих случаях эти инструменты используются без какой-либо интеграции, что обусловливает разделение процесса разработки на разные группы и негативно влияет на возможность формирования сквозных отчетов. Переход, особенно ускоренный, всех участников процесса на единый интегрированный инструмент ALM, как правило, невозможен. Поэтому, чтобы использовать преимущества сквозного управления жизненным циклом приложения, применяемые разрозненные инструменты приходится интегрировать для разных операционных систем, функциональности и групп.

Реальность требует интеграции. Существует огромное число инструментов, которые могут интегрироваться во многих аспектах. Для достижения сквозной прослеживаемости решение должно интегрировать все: от управления идеями до управления выпуском, контролем над изменениями и службы поддержки. Сюда входят такие аспекты, как сбор требований, проектирование архитектуры и создание модели, управление проектом, документированием и автоматизация тестирования. Число артефактов ALM очень велико, и количество отслеживаемых связей между ними увеличивается в геометрической прогрессии.

В данном документе мы исследуем преимущества тесной интеграции инструментов и то, как Team Foundation Server может выступать в роли платформы, объединяющей ряд таких интеграций и обеспечивающей единое интегрированное представление. Также будет рассмотрено четыре наиболее типовых подхода к интеграции инструментов ALM с примерами. Наконец, мы познакомимся с различными способами интеграции других инструментов ALM в Team Foundation Server 2010.

TEAM FOUNDATION SERVER КАК ИНТЕГРИРОВАННАЯ ПЛАТФОРМА ALM

Введение в Team Foundation Server 2010

Как говорилось ранее, Team Foundation Server 2010 является интегрированной платформой ALM, которая в значительной мере обеспечивает сквозную прослеживаемость, необходимую для решения ALM предприятия. Во многих случаях Team Foundation Server вместе с Visual Studio 2010 могут обеспечить преимущественное большинство требований по ALM, предъявляемых основной массой пользователей. Team Foundation Server – это основа разработки. Он обеспечивает:

- Взаимодействие через интеграцию всех ролей группы
- Регулярную и практически применимую обратную связь, что позволит сократить напрасные траты времени и усилий
- Возможность использования соответствующих поставленной задаче инструментов
- Использование по желанию лучших практик прозрачного и гибкого процесса разработки

Тесная интеграция с другими инструментами ALM

Team Foundation Server 2010 обеспечивает тесную интеграцию с другими инструментами ALM через специальные адаптеры. Поскольку Team Foundation Server выступает в роли платформы ALM, для каждого инструмента требуется создать всего один адаптер. Интеграции можно создавать с нуля, используя Team Foundation Server API или, реже, Веб-службы Team Foundation Server. Еще один подход – использование платформы интеграции TFS. Это универсальный инструмент, во многом обеспечивающий общую инфраструктуру, необходимую для интеграции разных систем.

ИНТЕГРАЦИЯ СУЩЕСТВУЮЩИХ ИНСТРУМЕНТОВ С TFS 2010

Чтобы понять широту разных интеграций с Team Foundation Server 2010, полезно рассмотреть примеры инструментов из разных областей ALM. Многие из этих инструментов расширяют возможности стандартного инструментария ALM, а некоторые охватывают части жизненного цикла разработки приложения, которые остались без внимания в Team Foundation Server 2010.

Сбор требований: stpsoft

Существуют различные подходы к сбору требований, и stpsoft поддерживает многие из них. Три их продукта – Requirements Definer, Storyboarding и ReqSheet – обеспечивают мощную интеграцию с Team Foundation Server 2010 и интегрированный механизм сбора и управления требованиями. Например, Requirements Definer и Storyboarding расширяют пакет ПО Visio, обеспечивая поддержку визуального описания требований и панель мониторинга состояния выполнения пользовательских историй.

По завершении описания требований Requirements Definer подключается к Team Foundation Server и создает новый рабочий элемент требований. После этого он изменяет все диаграммы процесса, созданные для поддержки рабочих элементов сопоставления требований в тестовые случаи в Team Foundation Server. В Team Foundation Server каждое требование связано с тестовыми случаями, созданными интеграцией, посредством отношений «Tested By» (Протестирован). Оказавшись в Team Foundation Server, требования и тестовые случаи становятся частью цепочки, обеспечивающей глубокую прослеживаемость всех артефактов разработки. В этот момент за дело берется ReqSheet, обеспечивая инструментарий для эффективного управления требованиями.

Управление проектами: Urban Turtle

Urban Turtle от Pyxis – один из многих инструментов управления проектами, которые тесно интегрируются с Team Foundation Server. Основная задача Urban Turtle – повысить эффективность работы групп, использующих скрам. Этот инструмент имеет простой в использовании Веб-интерфейс, напрямую интегрированный в Team Web Access (предоставляемый Team Foundation Server Веб-интерфейс). Интеграция с Team Foundation Server очень глубокая, поскольку Team Foundation Server обеспечивает хранилище полных данных о спринтах, пользовательских историях и задачах, которое может управляться через Urban Turtle.

Другими замечательными примера интеграции с Team Foundation Server в этой области являются Team Companion от Ekobit (интеграция напрямую через Microsoft Outlook), встроенная интеграция с Microsoft Project и предоставляемая бесплатно интеграция с Microsoft Project Server (можно найти на CodePlex).

Написание кода: TeamReview

Эффективное рецензирование кода всегда представляло проблему, не говоря уже о рецензировании распределенного кода. Проект TeamReview создан Дж.Б.Брауном и доступен бесплатно на сайте ресурсов с открытым исходным кодом CodePlex. TeamReview упрощает рецензирование кода, даже распределенное, через использование Team Foundation Server для отслеживания всех аспектов, принимаемых во внимание при рецензировании.

TeamReview – превосходный пример глубокой интеграции с Team Foundation Server. Реализация TeamReview включает новые типы рабочих элементов Team Foundation Server, интеграцию с Visual Studio 2010 и новые показатели для составления отчетов о мероприятиях по рецензированию кода. В Team Foundation Server отдельному сотруднику или группе назначается провести рецензирование кода. Чтобы добавить комментарий в ходе рецензирования, просто щелкните правой кнопкой мыши в интегрированной среде разработки Visual Studio. Впоследствии обзор кода может быть воспроизведен, и рецензированный код с комментариями будет автоматически выделен цветом.

Особенно привлекательной эту интеграцию делает то, что весь ее исходный код полностью доступен на сайте CodePlex. Таким образом, ее можно использовать как пример для справки при создании интеграций с Team Foundation Server 2010.

Управление выпуском версий: InRelease

Управлять выпуском версий в неоднородной среде и сложно, и затратно по времени. Продукт InRelease от компании InCycle Software помогает упростить управление выпуском версий через предоставление автоматизированного развертывания в разных средах на основании активной или пассивной модели развертывания. Также для каждой выпущенной версии полностью сохраняется журнал аудита, включая номера версий, утверждения на выпуск и показатели производительности.

Team Foundation Server 2010 обеспечивает хорошую поддержку автоматизированной сборки и развертывания, но InRelease идет еще дальше и предоставляет инфраструктуру уровня предприятия для управления выпуском версий поверх существующей функциональности Team Foundation Server. По умолчанию Team Foundation Server включает достаточное количество сведений относительно содержимого выпускаемой версии. InRelease использует эти данные для обеспечения процесса выпуска, добавляя при этом еще один уровень возможности проведения аудита и управления для поддержки автоматизированного развертывания во многих средах, управления совместно используемыми компонентами и выполнения автоматизированных тестов для каждой среды развертывания.

ЧЕТЫРЕ ПОДХОДА К ГЕТЕРОГЕННОМУ ALM

Существует три основных подхода к поддержке гетерогенного процесса управления жизненным циклом приложения. Каждый из них имеет преимущества и недостатки, но все обеспечивают интеграцию разных решений ALM на определенном уровне. В данном разделе мы кратко рассмотрим эти четыре подхода, начиная с обеспечивающего самую слабую интеграцию.

Интеграция клиентов

Первый метод – объединение данных нескольких хранилищ ALM на клиентском компьютере. В этом случае решения ALM не интегрируются на уровне сервера, а все отношения описываются на стороне клиента. Например, подключаемый модуль Mylyn от компании Tasktop выполняется на настольных компьютерах всех разработчиков или в их интегрированной среде разработки и предоставляет ряд модулей подключения к разным инструментам ALM. Каждый разработчик в отдельности может создавать отношения между элементами разных инструментов ALM. Скажем, разработчик может подключить задачу, над которой работает (которая отслеживается в одном инструменте ALM), к требованию, которое может отслеживаться другим инструментом ALM. Этот контекст сохраняется на уровне отдельного разработчика, но также может быть упакован и предоставлен для совместного использования другим участникам группы.

Преимущества

- Контекст свободно доступен участнику, работающему над задачей
- Есть возможность установления отношений между разными инструментами ALM
- Не требует серверной инфраструктуры
- Может помочь при переходе от одной системы к другой, например, с устаревшей системы SCM

Недостатки

- Невозможность формирования интегрированных отчетов
- Отношения устанавливаются и сохраняются на уровне отдельного разработчика; даже если разработчик подключает требования к исправлению дефекта, над которым работает в настоящее время, это отношение сложно понять на уровне группы
- Слабая интеграция инструментов ALM
- Связи, устанавливаемые клиентом, обычно очень хрупкие: изменение артефакта в одном из инструментов ALM часто приводит к тому, что ссылка на этот артефакт в другом клиенте становится недействительной

Заключение

Несмотря на всю его привлекательность на уровне отдельного разработчика, этот подход не обеспечивает возможности эффективного взаимодействия в рамках процесса ALM на уровне предприятия. Очень сложно установить и поддерживать структурированные отношения между системами, особенно если все эти отношения отслеживаются и поддерживаются на индивидуальной основе. Составлять отчеты о структуре системы практически невозможно, поскольку ни одна из этих систем не обеспечивает никакой возможности объединения данных множества отдельных «пакетов».

Двухточечная интеграция

Решения с двухточечной интеграцией устанавливают прямые связи между различными инструментами ALM. Этот подход используется несколькими производителями, и большинство из них ограничивают число поддерживаемых инструментов, поскольку с добавлением поддержки очередного инструмента число интеграций, требующих реализации, растет в геометрической прогрессии. Из-за этой сложности лишь несколько интегрированных решений ALM используют этот подход, и мы не будем останавливаться на нем здесь подробнее.

Поскольку для поддержки большого числа инструментов ALM требуется огромное число модулей подключения, самым популярным подходом к двухточечной интеграции является использование стандартных интерфейсов. Этот подход используется инициативой Открытые сервисы для взаимодействия в рамках жизненного цикла (Open Services for Lifecycle Collaboration, OSLC), продвигаемой IBM. Данная инициатива еще находится на ранних этапах развития и предлагает использование стандартных построенных на принципах REST URL для установления связей между разными системами. Для установления связи между двумя наборами инструментов ALM вынужденно используется набор возможностей, получивших самое широкое распространение. Например, система отслеживания дефектов поддерживает только ссылку в виде URL на регистрацию изменений в системе управления версиями. Это позволяет системе отслеживания дефектов знать о регистрации изменений, но, кроме URL, не обеспечивает практически никаких других данных. Поэтому пользователю, работающему с конкретным дефектом, приходится переходить по URL, чтобы получить более подробные сведения об изменениях прямо из инструмента управления версиями. Причиной этому является неспособность стандартного набора данных включать сложные метаданные о дефекте.

Преимущества

- Стандартные интерфейсы ограничивают число оконечных точек интеграции, которые должны быть созданы
- Каждый инструмент ALM должен реализовать только один адаптер для обмена данными с другими инструментами ALM
- Связь между инструментами представляется как простой в использовании URL

Недостатки

- Чрезвычайно низкая связанность между инструментами ALM
- Большинство инструментов ALM до сих пор не поддерживают OSLC или другой стандарт
- Для получения более широкой поддержки основанные на стандартах интеграции стремятся предоставить минимальное число общих свойств
- Возможность составления сквозных отчетов очень ограничена из-за двухсторонней природы решения и отсутствия централизованного хранилища данных

Заключение

Этот подход обеспечивает немного больше данных, чем инструменты интеграции на стороне клиента. Двухточечные решения, даже основанные на стандартах, по-прежнему очень сложно использовать в решении уровня предприятия. Частично из-за его незрелости, но также из-за природы предлагаемой реализации. Для эффективного формирования отчетов о сквозном процессе разработки требуется создать некоторый механизм отчетов, умеющий собирать данные об артефактах разработки и отношениях между ними. Но универсальность открытых стандартов, которые применимы ко всем системам ALM, вынуждает использовать только базовые данные, т.е. расширенные возможности отдельных инструментов могут не поддерживаться. Это приводит к ограничению данных, которые могут использоваться для поддержки возможности формирования сквозных отчетов. В OSLC делается попытка обойти эту проблему через использование ссылок в виде URL на другие системы. Переход по этим ссылкам приводит к представлению полных данных, предоставляемых системой управления версиями.

Интегрированная платформа ALM (с полнофункциональными адаптерами)

Самую тесную интеграцию процесса ALM обеспечивает интегрированное решение ALM с полнофункциональными адаптерами. При таком подходе выбирается платформа ALM, обеспечивающая самые глубокие связи между артефактами жизненного цикла. Как правило, платформа обеспечивает большую часть общих требований ALM, позволяя подключение любых дополнительных инструментов ALM посредством полнофункциональных адаптеров.

Интегрированная платформа ALM выступает в роли узла интеграции систем. При ее использовании для каждого дополнительного инструмента требуется отдельный адаптер. Но поскольку большинство инструментов ALM встроены в платформу, адаптеры необходимы только для инструментов ALM, не предоставляемых платформой, устаревших инструментов, которые должны быть интегрированы по соображениям бизнеса, или для инструментов, являющихся предпочтительными для решений, предлагаемых интегрированной платформой.

Таким образом, данный подход сочетает в себе преимущества расширяемой серверной части, обеспечивая при этом готовые решения для многих типовых задач ALM. Он концептуально отличается от двухточечного решения, поскольку полнофункциональная интеграция выполняется для каждого инструмента ALM однократно. Это решение лучше, чем двухточечное решение, потому что не требует создания огромного числа интеграций, и лучше основанных на стандартах двухточечных решений, потому что интеграции могут не ограничиваться лишь самыми распространенными типами метаданных.

Team Foundation Server 2010 от Майкрософт – главный пример такого типа интеграции на сегодняшний день. В него по умолчанию включена основная масса необходимых инструментов ALM, а для любых дополнительных интеграций создаются полнофункциональные адаптеры (Майкрософт или сторонними производителями). Это обеспечивает тесную взаимосвязанность всего процесса разработки, допуская при этом интеграцию с другими инструментами ALM. Кроме того, поскольку Team Foundation Server 2010 обеспечиваются возможности формирования отчетов, внешние инструменты ALM могут быть глубоко интегрированы в инфраструктуру отчетов.

Преимущества

- Глубокая интеграция между артефактами создает условия для превосходной сквозной прослеживаемости
- Централизованное хранение упрощает формирование отчетов
- Согласованная архитектура и единообразие взаимодействия с пользователем в ходе всего жизненного цикла
- Требуется меньше адаптеров, чем для двухточечных решений
- Поддерживаются полнофункциональные адаптеры
- Может обеспечить сокращение общего числа используемых инструментов

Недостатки

- Есть сложности с поддержкой «наилучших» сочетаний отдельных инструментов ALM
- Проектировщики и разработчики адаптеров должны хорошо разбираться в обеих системах

Заключение

Единая интегрированная платформа ALM с адаптерами – подход, обеспечивающий самую высокую интеграцию. Большая часть подробных сведений об артефактах и их взаимоотношениях обеспечивается интегрированной природой самой платформы ALM. Кроме того, любой адаптер может демонстрировать полную функциональность, что позволяет через интеграцию расширять возможности платформы или интегрироваться с существующей инфраструктурой. Производителям требуется лишь создать одну полнофункциональную интеграцию, чтобы использовать основные службы платформы. Дополнительное преимущество в том, что инфраструктура отчетов уже существует и без труда расширяется для включения новых данных, получаемых от интеграции.

Они не только объединяют разные аспекты процесса разработки, но также способствуют сокращению числа используемых инструментов ALM. Тогда нет необходимости инвестировать в дополнительное промежуточное ПО, что влечет за собой экономию средств как на лицензирование, так и на обслуживание. Наконец, с интегрированными платформами ALM обычно проще работать благодаря общей архитектуре и шаблонам использования.

Благодаря обеспечению полной функциональности, потенциалу для экономии средств и простоте использования интегрированные платформы ALM часто являются лучшим выбором для решений интеграции ALM на уровне предприятия. Поэтому мы продолжим рассмотрение решения интеграции ALM, делая упор на Team Foundation Server 2010.

СОЗДАНИЕ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕГРАЦИЙ

Несмотря на развитую партнерскую экосистему Team Foundation Server, по-прежнему может возникать множество ситуаций, когда пользователям захочется разработать собственную интеграцию. Один из наиболее очевидных случаев – интеграция собственных решений. Еще один вариант – обеспечение интеграции двух систем на короткий период времени для упрощения перехода с одного инструмента на другой. Наконец, может просто возникнуть желание расширить возможности Team Foundation Server дополнительными инструментами ALM.

Существует три основных набора инструментов и методов для реализации интеграции с Team Foundation Server. Все они могут использоваться по отдельности или в сочетании друг с другом. Этими инструментами являются платформа интеграции TFS, открытые API и Веб-службы и адаптеры хранилища данных.

Платформа интеграции TFS

Платформа интеграции TFS – лучший выбор для большинства интеграций на стороне сервера. Она обеспечивает достаточное количество встроенных возможностей, благодаря которым создавать адаптеры становится относительно просто. Для создания и развертывания пакета платформы интеграции TFS не требуется глубокого знания внутренних компонентов TFS, хотя они могут обеспечивать чрезвычайно мощные возможности.

В настоящее время из всех наборов возможностей поддерживаются только рабочие элементы и управление версиями. Но уже только они обеспечивают большинство всех потребностей интеграции ALM. Некоторые возможности, такие как сборка, пока что не реализованы, но планируются для будущих версий. Однако даже несмотря на отсутствие поддержки всех возможностей, платформа интеграции TFS обеспечивает такую обширную и мощную базу, что имеет смысл расширять ее, а не создавать собственные интеграции. Платформа интеграции TFS выполняется как служба либо на Team Foundation Server, либо на другом компьютере сети.

Чтобы создать интеграцию, используя платформу интеграции TFS, необходимо создать адаптер для интегрируемого инструмента. Этот процесс может быть как предельно простым, так и относительно сложным, в зависимости от того какой тип синхронизации (одностороннюю или двухстороннюю) предполагается обеспечивать. Но в любом случае платформа интеграции TFS обеспечивает несколько встроенных профессиональных адаптеров для различных версий TFS. Они абстрагируют в себе все сложные аспекты интеграции, способствуя упрощению и ускорению создания пользовательских интеграций.

Одно из основных предназначений платформы интеграции TFS – упрощение постепенной миграции с одной системы на другую. Некоторые из существующих на данный момент адаптеров позволяют синхронизировать данные Team Foundation Server и выполняющихся параллельно с ним инструментов ALM. Таким образом, обеспечивается плавный переход от устаревших инструментов к Team Foundation Server. Например, и ClearCase, и ClearQuest имеют адаптеры для платформы интеграции TFS, благодаря чему эти две системы могут синхронизироваться. Такая возможность исключительно полезна для групп, которые сделали крупные вложения в автоматизацию на базе ClearQuest, но хотят уже сегодня перевести группы разработки на Team Foundation Server, чтобы обеспечить сквозную поддержку жизненного цикла приложения. Другими примерами интеграций, построенных на платформе интеграции TFS, являются адаптеры системы управления версиями для Subversion, Perforce, and SharePoint.

Платформа интеграции TFS обеспечивает массу преимуществ по сравнению с созданием собственной платформы. Прежде всего, платформа и основная инфраструктура сложных интеграций легко расширяемые. Как известно разработчикам адаптеров интеграции, самой большой сложностью в любой интеграции является обработка крайних случаев, на что уходит больше всего сил и времени. Разработчики платформы интеграции TFS сталкивались со многими такими проблемами в течение многих лет и выработали решения для них, которые интегрировали в данную платформу. Кроме того, платформа интеграции TFS создана для интеграции крупномасштабных решений и, таким образом, хорошо масштабируется и имеет ряд возможностей корректировки ошибок.

Рекомендуется применять

- Для создания надежной интеграции между двумя системами
- Когда требуется обеспечить глубокую интеграцию с внутренними компонентами среды TFS
- Для размещения службы, координирующей работу двух систем

Не рекомендуется применять

- Если требуется легковесная интеграция и есть желание потратить время и усилия на это
- Для создания очень простой интеграции между двумя системами
- Если размещение службы как части инфраструктуры невозможно (клиентские решения)

Заключение

Платформа интеграции TFS – это мощная платформа для интеграции системы с Team Foundation Server. Практически всегда возможность ее применения должна рассматриваться в первую очередь. И только в крайних случаях интеграции ALM предприятия должны создаваться с нуля.

Открытые API Java и .Net

Применение открытых API для Team Foundation Server обеспечивает возможность быстрой низкоуровневой интеграции с платформой Team Foundation Server. Существует несколько основных API, каждый из которых охватывает определенные аспекты платформы Team Foundation Server, такие как управление версиями, отслеживание рабочих элементов, сборки. Многие интеграции сторонних производителей, которые выполняются на клиенте, используют API для работы напрямую с Team Foundation Server. Например, Team Companion, замечательный подключаемый модуль для Microsoft Outlook, обеспечивающий полнофункциональную клиентскую часть для Team Foundation Server, использует API TFS для связи с TFS сервером напрямую.

Для прямого обмена данными с Team Foundation Server предпочтительнее пользоваться этими API. Кстати, Майкрософт рекомендует никогда не работать с базами данных напрямую, поскольку они могут меняться от версии к версии. Кроме того, изменения баз данных могут иметь неожиданное негативное влияние на производительность или стабильность Team Foundation Server. Для групп разработки, работающих на платформах .Net и Java, существуют бесплатные API.

Рекомендуется применять

- Для создания очень простой интеграции на стороне сервера (такой как единоразовая передача данных в Team Foundation Server)
- Для создания клиентских интеграций с Team Foundation Server
- Для произвольных сценариев для работы с Team Foundation Server
- Для создания интеграции из системы, работающей под управлением другой операционной системы, не Windows
- Если требуется создать легковесное решение интеграции, обеспечивающее тесное связывание, и для его реализации может быть выделено достаточно ресурсов

Не рекомендуется применять

- Для создания серверной интеграции (в этом случае лучше использовать платформу интеграции TFS)

Заключение

API являются мощным низкоуровневым инструментом интеграции с Team Foundation Server. Они производительны, просты в использовании и обеспечивают создание разнообразнейших интеграций. Но при их использовании для создания полнофункциональной интеграции придется писать много кода. Поэтому для работы с Team Foundation Server рекомендуется применять платформу интеграции TFS и прибегать к API, только если это невозможно.

Глубокая интеграция ALM

Иногда инструменты ALM требуется интегрировать в общий интерфейс, так чтобы они выступали в роли полностью интегрированных полноправных составляющих платформы Team Foundation Server. Для таких случаев предусмотрены дополнительные точки интеграции, которые могут применяться для создания намного более тесной интеграции, чем обеспечивается API платформы интеграции TFS или Team Foundation Server.

Одним из основных различий между TFS и другими решениями ALM является то, что TFS обеспечивает интегрированное хранилище, в котором размещаются данные с поддержкой непротиворечивости через использование транзакций. Это хранилище данных периодически перерабатывается в многомерный OLAP-куб, что значительно упрощает создание специальных отчетов. OLAP-куб предоставляет множество отношений между артефактами ALM Team Foundation Server, что позволяет создавать разнообразнейшие отчеты по всему жизненному циклу. Например, очень просто создать отчет, показывающий, сколько дефектов ассоциировано с каждым файлом кода. При сортировке по убыванию этот отчет демонстрирует, какие файлы кода содержат наибольшее количество ошибок, и выявляет кандидатов на реструктуризацию. Этот отчет легко сформировать, потому что OLAP-куб обеспечивает глубокую связь данных между кодом и рабочими элементами. При создании интеграции для инструмента ALM можно также создать глубокую связь между данными предприятия и имеющимися данными TFS.

Примером интеграции, которая выиграет от глубокой интеграции отчетности, является система учета времени и выставления счетов. В идеале, такая система устанавливает связь между временем, проведенным за работой над определенными рабочими элементами, целевым клиентом, который выполнял работу, и кодом или другими артефактами, зарегистрированными в системе управления версиями. Распространяя эти данные в TFS для обеспечения поддержки одного или более из этих новых отношений, инструменты могут обеспечить составление более детальных отчетов в ходе жизненного цикла, чем это было бы возможным, если бы отчеты формировались только на основании данных системы учета времени и выставления счетов. Например, специалист, выполняющий интеграцию, должен был бы просто показать связь между затраченным временем и регистрацией изменения в системе управления версиями, и связи с рабочими элементами и идентификатором разработчика, работающего над этим кодом, были бы установлены автоматически. Теперь без каких-либо доработок система учета времени и выставления счетов могла бы предоставлять отчеты для сравнения качества кода, написанного на разных этапах процесса разработки. Например, код, написанный в конце процесса разработки проекта, когда сроки уже поджимают, может быть более низкого качества, и создание всего одной интеграции с TFS позволит получать эти данные. Одним словом, всего одна интеграция с хранилищем данных Team Foundation Server открывает доступ к огромному массиву данных, поскольку эта одна связь подключается к множеству уже существующих связей.

Расширение хранилища данных и OLAP-куба выполняется через создание пользовательского адаптера хранилища данных. Team Foundation Server не требует от специалиста, выполняющего интеграцию, глубоких знаний об организации хранения данных и OLAP-кубах, а просто позволяет встраивать адаптеры хранилища данных в код. Для создания адаптера хранилища данных необходимо использовать API хранилища данных, реализовать пару ключевых интерфейсов и изменить некоторые конфигурационные XML-файлы. Готовый адаптер вместе с конфигурационными файлами развертывается в Team Foundation Server. После того как адаптер собран и установлен, обновления происходят автоматически по графику, определенному для Team Foundation Server.

Рекомендуется применять

- Для создания глубокой интеграции данных между инструментом ALM и Team Foundation Server
- В случае необходимости обеспечить сквозную прослеживаемость для артефакта, который в настоящий момент не прослеживается
- Для интеграции данных или артефакта ALM с инфраструктурой отчетов Team Foundation Server

Не рекомендуется применять

- Если интеграции отчетности не требуется

Заключение

Создание пользовательского адаптера хранилища данных является очень мощным механизмом интеграции пользовательских данных с инфраструктурой отчетов Team Foundation Server. Серьезно подумайте о таком уровне интеграции отчетности для своего решения, если хотите, чтобы ваши данные или артефакты были включены в сквозную прослеживаемость жизненного цикла, обеспечиваемую Team Foundation Server.

ВЫВОДЫ

Для поддержки гетерогенной среды ALM на уровне предприятия полнофункциональные интеграции являются обязательным условием. Подробные данные обеспечивают глубокий анализ, и ключом к нему является возможность формирования отчетов. Team Foundation Server 2010 поддерживает полнофункциональную интеграцию, а также мощную, расширяемую и простую в использовании инфраструктуру отчетов, которая обеспечивает формирование отчетов на протяжении всего жизненного цикла ALM.

Кроме того, использование платформы ALM с самого начала означает меньше работы по интеграции и более быстрый запуск проекта. В Team Foundation Server 2010 встроена основная функциональность ALM, используемая большинством компаний. Также эта функциональность доступна на большинстве операционных систем и платформах, что существенно сокращает необходимость в пользовательских интеграциях или интеграциях сторонних производителей и позволяет компаниям сосредоточиться на своих основных задачах.

Если вашей организации необходим инструмент сквозного управления жизненным циклом приложения, который должен поддерживать гетерогенные среды, стоит обратить свое внимание на Team Foundation Server 2010 и платформу Visual Studio 2010.

Гетерогенный процесс разработки

с Visual Studio 2010

Многие организации разрабатывают и поддерживают приложения, написанные на Java, .NET и других платформах. Сегодня эти несопоставимые платформы привели к разделению на окружения разработки, лишая организации возможности сквозного визуального контроля процесса разработки. Это разделение также означает, что любые улучшения процесса, реализованные одной группой разработки, не имеют положительного эффекта для другой группы. С появлением Visual Studio 2010 это болезненное разделение уходит в прошлое.

Get the free mobile app for your phone
http://gettag.mobi

КАК ВОЗНИКЛО РАЗДЕЛЕНИЕ

Долгое время работа с инструментами разработки Майкрософт означала работу под управлением Microsoft Windows и, вероятнее всего, над кодом, который предполагалось выполнять на этих платформах. Организации выбирали инструменты, исходя из используемых технологий, потому что инструменты и технологии сопутствовали друг другу. Например, если создавалось приложение на Visual C++, для этого использовался инструмент, работающий под управлением Windows.

Если использовался Java, скорее всего, разработка велась в интегрированной среде разработки (integrated development environment, IDE) производства компаний Sun, IBM или в одной из сред с открытым исходным кодом. Разработка могла вестись на компьютере, работающем под управлением Windows, но развертывание часто выполнялось в одной из разновидностей Unix/Linux. Выбор дополнительной платформы обычно предлагал совершенно другой набор технологий. Это было благодатной почвой для неразберихи и догм. Достижение эффективности за счет роста объемов производства при таком неоднородном пейзаже практически всегда было иллюзорным.

Подчас это было нечто большее, чем профессиональные предпочтения, но организации часто разделяли группы соответственно используемым ими инструментам. Из-за устаревших систем и необходимости выполнения процедур объединения и слияния организации просто были вынуждены использовать множество платформ и часто также поддерживать избыточный процесс, группы и инструментарий разработки. Стало нормой то, что инструменты, группы разработки и языки программирования были в значительной степени подобными, но очень редко соприкасались. Ты принимал ту или иную сторону, и часто это определяло твою карьеру.

Таков был обычный порядок вещей. Крупным консалтинговым фирмам, большим ИТ-группам и разработчикам продуктов в мире функциональной совместимости в конце концов потребовалось обширный, как правило, изолированный инструментарий. Существовали специальные люди и экспертные группы для многих и разнообразных платформ. Окружения разработки укрепились, и большинство привыкло к изоляции; люди научились сосуществовать, но соприкасались редко.

Инструментарий управления процессом обычно тоже был элементом окружения разработки. При наличии общего процесса он часто был отделен от написания кода; компании имели отдельный штат по управлению процессом и группы разработки продукта для разных наборов технологий. Создание и развертывание для мэйнфрема и других разнородных систем для внутреннего ИТ-отдела и взаимодействующих с внешним миром центров обработки данных породило большое число специалистов и искусственно возводимый стен. Таким образом, шансы объединить процесс и инструментарий в большой мере не осуществлялись. Инструменты для проектирования архитектуры и инструменты для других областей разработки представлялись единичными решениями; целостное визуальное представление процесса и возможность обмена мнениями и целями оставались нереализованными. Мир не спешил поддерживать единение.

НОВЫЕ ПАРАДИГМЫ

С выходом Team Foundation Server 2005 в игру были введены некоторые интересные концепции. Сочетание инструментов управления процессом и IDE, а также совместное использование общего представления действий группы проекта было фундаментальным изменением. Сведение всех участников групп в одно целостное представление, описывающее единый общий процесс посредством множества данных, было чем-то новым и необычным. Появился цельный процесс управления жизненным циклом приложения с тесной интеграцией технических групп.

Теперь Visual Studio 2010 идет еще дальше. Идеи о том, что процесс и инструментарий должны дублироваться для разных используемых технологий, и что разрозненность – это нормально, теперь под вопросом. Подключаемый модуль Team Foundation Server для Eclipse (рис. 1) теперь объединяет практиков, которые так долго оставались в изоляции друг от друга, в одном унифицированном процессе независимо от используемой технологии.

Рисунок 1 - Редактирование кода в Eclipse с помощью подключаемого модуля Team Foundation Server.

Подключаемый модуль Team Foundation Server для Eclipse переносит основную массу возможностей редактирования и функциональности ALM Team Explorer в интегрированную среду разработки Eclipse. В течение многих лет Eclipse была самой популярной интегрированной средой разработки в мире Java; она включает инструменты и расширения для ряда технологий. Разработка на таких языках, как Java/J2EE, C++, Python, Perl, обычно ведется в Eclipse.

Разработка в Eclipse теперь может означать разработку с управлением процессом, обеспечиваемым Visual Studio Team Foundation Server 2010. Кроме управления версиями в своей привычной среде разработки, пользователи Eclipse теперь получили доступ к тем же рабочим элементам и шаблонам процесса, тому же порталу, документам и отчетам проекта и тем же коллекциям проектов, что и разработчики на платформе .NET. И все это не покидая выбранной среды разработки! Код или Веб-инфраструктура, объединяющая в себе .NET, PHP, Java/JSP, Python, Perl, Django или другие технологии, может управляться и обслуживаться с помощью единого набора инструментов ALM. Разработчики могут продолжать использовать привычные инструменты и в то же время подключаться и взаимодействовать друг с другом через общую инфраструктуру ALM Visual Studio 2010. Группы, работающие с разными технологиями, теперь могут действовать заодно, поддерживая общее видение и цели и взаимодействуя в совместно используемом инструменте.

Если вы работаете преимущественно в Eclipse и ранее не сталкивались с Visual Studio, тесная связь между инструментарием управления процессом и разработки кода для вас может быть непривычной. В зависимости от используемых инструментов вы имели сложности либо с получением соответствующих показателей, либо испытывали недостаток в общей среде для координации совместной деятельности. Вне всякого сомнения, имелся набор различных инструментов и масса сложностей с получением общей картины. Использование возможности сопоставления наборов изменений в коде с функциями или дефектами в инструментарии отслеживания и управления требованиями, вероятно, означало дополнительные расходы на дорогостоящее лицензирование. Но благодаря Team Foundation Server 2010 мощные инструменты координации совместной деятельности доступны прямо из интегрированных сред разработки на базе Eclipse.

Рисунок 2 - Подключаемый модуль The Team Foundation Server для Eclipse обеспечивает привычные элементы для работы с ресурсами.

УПРАВЛЕНИЕ ЖИЗНЕННЫМ ЦИКЛОМ ПРИЛОЖЕНИЙ ДЛЯ ВСЕХ И ВЕЗДЕ!

Visual Studio Team Foundation Server 2010 лежит в основе предложения ALM Visual Studio, поэтому возможность просматривать информацию и статистические данные проекта одинаково доступна на многих платформах. Теперь организации, использующие гетерогенные технологии, получили реальную возможность управления жизненным циклом приложения со всей глубиной визуального контроля и понимания процесса, которые обеспечиваются хранилищем данных и службами аналитики SQL Server. Это ALM и совместно используемые инструменты для всех!

Для разработчиков, кто фактически живет в мире смешанной разработки, число инструментов сократилось. Те, кто регулярно работает над проектами, в которых используется несколько языков программирования, теперь может пользоваться преимуществами по-настоящему общей базы кода. Это фундаментальный принцип для разработки с применением гибких методик. Имея в своем распоряжении прочный инструментарий координации совместной деятельности и простой доступ к показателям, вы получаете твердое понимание своих проектов независимо от сочетания используемых для их реализации технологий.

Кроме того, сокращаются издержки на администрирование и поддержку процесса. Использование в рамках организации одного интегрированного набора инструментов для ИТ-отдела и разработки, а не разных инструментов и технологий управления процессом для разных групп, означает избавление от обязательного дублирования процесса в разных областях; также это означает сокращение перечня требуемых навыков для обслуживания процесса. Специалисты, обеспечивающие сопровождение, могут сосредоточиться на одной инфраструктуре управления процессом и обеспечивать улучшение процесса для более широкой аудитории.

С подключаемым модулем TFS для Eclipse гетерогенная разработка перестала быть процессом «мы против них». Совместная работа групп упрощена, и изолированность окружений разработки уходит в прошлое. Общий процесс и отчетность, единый набор инструментов и код теперь делают возможным переход на новый уровень продуктивности и эффективности для ранее изолированных и противостоящих друг другу групп. Поскольку теперь бизнес-требования могут без труда пересекать существовавшие ранее границы между технологиями, обеспечиваются намного большие возможности визуального контроля, чем когда либо до этого.

С Visual Studio 2010 прозрачность и совместная деятельность уже не пустой звук. Майкрософт разрушает старые границы и обеспечивает новые способы работы и новые уровни продуктивности.

Тестирование производительности и нагрузочное тестирование

на протяжении всего жизненного цикла приложения

Низкая производительность приложений стоит компаниям миллионы долларов каждый год и репутации. Простая задача – создать ПО, которое будет вести себя предсказуемо, согласованно и отзывчиво, по-прежнему вызывает трудности. Приложения для использования в компании, имеющие низкую производительность, вводят еще один фактор затрат из-за потери продуктивности и невыполнения сроков. Microsoft Visual Studio 2010 обеспечивает инструменты для измерения, улучшения и проверки производительности приложения в самых напряженных условиях, что позволяет гарантировать предсказуемость поведения ПО независимо от ситуации.

Get the free mobile app for your phone
http://gettag.mobi

Низкая производительность приложений стоит компаниям миллионы долларов каждый год и репутации. Почему это происходит, предельно ясно – пользователи не могут попасть на необходимый им Веб-сайт, или не могут делать покупки, потому что приложение либо отвечает медленно, либо не отвечает вообще. И это только Веб-приложения. Приложения для использования внутри компаний, которые страдают низкой производительностью, убивают время, а время дорого, как по причине потери продуктивности, так и несоблюдения сроков заказов. В прошлом низкая производительность, возможно, не имела такого пагубного влияния, но сегодня она оборачивается огромными потерями. Ситуацию надо менять. Microsoft Visual Studio 2010 обеспечивает инструменты для измерения, улучшения и проверки производительности приложений в самых напряженных условиях, что позволяет гарантировать предсказуемость поведения ПО независимо от ситуации. Во многих случаях тестирование производительности вообще не проводится. И основная причина этому не недостаток времени, а стоимость и сложность инструментальных средств, необходимых для такого тестирования.

ПРИВЫЧНАЯ СРЕДА

Одно из ключевых преимуществ использования предлагаемых Visual Studio 2010 интегрированных инструментов для тестирования производительности в том, что они выполняются в привычной для разработчиков среде. Visual Studio обеспечивает интуитивно понятный интерфейс для создания Веб-тестов, модульных тестов и связанных с ними нагрузочных тестов. И все тесты базируются на одной и той же среде тестирования, которая за последние пять лет эволюционировала и развилась в надежную и хорошо поддерживаемую инфраструктуру.

Еще одним преимуществом, помимо интеграции с Visual Studio, является то что для настройки тестов не потребуется изучать новые языки программирования, они полностью поддерживаются Microsoft .NET Framework. Для работы со многими другими средствами тестирования, являющимися отраслевыми стандартами, необходимо изучить новый язык сценариев или язык программирования и новую интегрированную среду разработки. Использование инструментов тестирования Visual Studio позволяет приступить к работе в очень короткий период времени с минимальной необходимостью обучения.

Раннее выявление проблем с производительностью

Причины проблем с производительностью приложения разнообразны. Чаще всего это ошибочный выбор архитектуры. Неправильная архитектура приводит к образованию узких мест в коде, что может замедлить выполнение всего приложения. Даже введение более производительного оборудования не решает проблемы в этом случае. Исправлять проблем, истоки которых кроются в архитектуре приложения, может быть очень сложно и дорого. Выявление такой проблемы на ранних этапах процесса разработки сбережет существенное количество времени и средств.

Моделирование поведения пользователей с помощью тестов производительности

Тесты Веб-приложений являются одним из ключевых компонентов тестирования производительности в Visual Studio 2010. Эта возможность была введена в Visual Studio 2005. Майкрософт взяли уже готовый инструмент и доработали его, упростив работу с ним и расширив функциональность.

Тесты Веб-приложений выполняют запись трафика браузера на уровне протокола HTTP, которая впоследствии может быть воспроизведена при нагрузочном тестировании для моделирования поведения пользователя. Эти тесты легковесны и очень эффективны для формирования большой нагрузки.

Тесты Веб-приложений обладают насыщенной функциональностью. Ajax и ресурсы страницы обрабатываются автоматически; без всякого написания кода в тесты можно добавлять условия и циклы. Кроме того, динамические параметры, файлы cookie и аутентификация также будут обработаны за вас. Возможность привязки тестов к данным позволяет создавать гибкие управляемые данными тесты. В дополнение к этому тесты Веб-приложений являются расширяемыми. Например, если встроенные правила не обеспечивают требуемой сложной проверки, можно самостоятельно добавить собственные правила валидации. Необходимо выполнять специальную обработку динамических параметров для тестируемого приложения? Создание подключаемого модуля потребует минимального написания кода. Имеется множество точек расширения, которые позволяют настраивать любой аспект тестирования Веб-приложения. На рис. 1 представлены результаты тестирования Веб-приложения.

Рисунок 1 - Результаты тестирования Веб-приложения

Использование модульных тестов для управления нагрузкой

Одно из замечательных преимуществ инструментов тестирования производительности и нагрузочного тестирования в VS 2010 – возможность работать с модульными тестами. То есть для тестирования производительности приложения не требуется, чтобы был полностью реализован пользовательский интерфейс или готовы все части приложения. Кроме того, если есть подозрение на образование узкого места производительности, тестирование может проводиться на более ранних этапах процесса разработки, чем обычно. Также имеется еще одно преимущество при работе с Веб-службами: по мере роста числа приложений, использующих Веб-службы для поддержки пользовательских интерфейсов, производительность этих Веб-служб может тестироваться независимо.

Создание нагрузочных тестов

Как осуществляется тестирование производительности в Visual Studio 2010? Во-первых, могут создаваться либо модульные тесты, либо тесты Веб-приложений, либо сочетание обоих типов тестов. Далее с помощью простого мастера можно разработать сложные нагрузочные тесты, задавая разные значения пропускной способности сети, схемы нагружения, сочетания тестов и браузеры. Кроме того, можно задать время выхода на рабочие показатели нагрузки, продолжительность выполнения теста, применение задержки на обработку и определить компьютеры, показания счетчиков производительности которых будут регистрироваться. Интерфейс выбора всех этих опций прост, а их сочетание может использоваться для создания практически любого профиля производительности. Хотите протестировать производительность приложения для нескольких пользователей, использующих сеть со стандартной полосой пропускания 10 МБ/с, тогда как все остальные применяют коммутируемое соединение? Требуется собрать подробные сведения о производительности не только тестируемой системы промежуточного уровня, но и уровня доступа к данным, прокси-сервера, балансировщика сетевой нагрузки и других устройств, образующих канал связи? Все это можно настроить с помощью мастера. Мастер нагрузочного тестирования представлен на рис. 2.

Рисунок 2 - Задание сочетания тестов, соответственно предполагаемому использованию.

С помощью наборов счетчиков можно без труда собирать показания счетчиков производительности тестируемой системы. Эти счетчики поставляются с заданными пороговыми значениями нагрузки, что позволяет выявлять чрезмерно загруженные ресурсы (рис. 3).

Можно также расширить структуру тестирования, создав пользовательские сборщики данных для сбора необходимых сведений. Применение этих возможностей позволяет регистрировать специальные регистрационные журналы приложения, данные о загруженности сети, обращения к SQL Server или практически любые необходимые данные.

Рисунок 3 – Выбор наборов счетчиков производительности.

В дополнение ко всем предлагаемым опциям – их слишком много, чтобы перечислять здесь – есть возможность масштабировать, масштабировать и масштабировать. Контроллеры и агенты нагрузочного тестирования масштабируются, обеспечивая поддержку сценариев предприятия, а это десятки тысяч пользователей. И все это по цене, намного меньшей, чем когда-либо до этого. Самая хорошая новость – простота лицензирования: база пользователей расширяется посредством приобретения пакетов пользователей. Нет дифференциации типов пользователей, дополнительных сборов за дополнительные контроллеры или каких-либо скрытых сюрпризов. Это обеспечивает гибкость и прогнозируемость затрат на тестирование производительности.

Анализ результатов тестирования

Нагрузочное тестирование – это замечательно, но если нет возможности анализировать результаты тестирования, сами тесты не представляют особой ценности. Графики анализатора тестовой нагрузки Visual Studio позволяют соотнести падения производительности с активностью и условиями на сервере, такими как количество ошибок, зарегистрированных в журнале событий, или чрезмерная загруженность ресурсов (рис. 4). Пороговые значения счетчиков производительности для нагрузочного теста настраиваются автоматически и позволяют быстро выявлять ресурсы сервера, используемые особенно интенсивно.

Рисунок 4 - Сводка результатов нагрузочного тестирования.

С помощью диаграммы активности виртуальных пользователей можно определить, какие действия виртуальных пользователей привели к падению производительности (рис. 5).

Рисунок 5 – Диаграмма активности виртуальных пользователей.

Устранение узких мест

Как только проблема установлена, она должна быть устранена. Для этого Visual Studio 2010 использует сборщик данных Microsoft ASP.NET Profiler. Сборщик данных позволяет использовать выборки или инструментировать код, и даже код в .NET Framework, для анализа на наличие узких мест вплоть до уровня методов. Типовой сценарий может быть таким: в ходе выполнения нагрузочного теста выявляется сценарий, элемент заказа, который выполняется медленнее, чем должен, и требуется выяснить причину этого. Когда проблемный сценарий выявлен, в ASP.NET Profiler можно получить подробные сведения по каждому методу, вызываемому в ходе сеанса (рис. 6). Это позволяет определить критический путь в коде, демонстрирующий самую низкую производительность.

Что такое ASP.NET Profiler? Этот сборщик данных профилирует процесс служб IIS (Internet Information Services). Он отслеживает общую производительность IIS для определенного приложения, записывает счетчики производительности и отслеживает код вплоть до уровня методов. Можно также собрать сведения об обращениях к СУБД, чтобы определить, какое влияние оказывают эти обращения на общую производительность приложения.

Рисунок 6 - Представление детальных сведений о функции

Выявление проблем с производительностью для Веб-пользователей до развертывания

ASP.NET Profiler – отличный инструмент для выявления проблем с производительностью приложений. Вам когда-либо приходилось сталкиваться с ситуацией, когда приложение локально работает прекрасно, но демонстрирует низкую производительность для пользователей, осуществляющих доступ к нему по глобальной сети? Многие проблемы с производительностью создаются «болтливыми» клиентами или клиентами, передающими и принимающими большие объемы данных. Такие приложения хорошо работают в локальных сетях, но демонстрируют низкую производительность при доступе к ним через глобальную сеть. С VS 2010 можно выполнять приложения, имитируя работу с ним по глобальной сети, что позволяет выявлять этот тип проблем с производительностью на ранних этапах.

Истинная эмуляция сети Microsoft Visual Studio 2010 использует программную истинную эмуляцию сети для всех типов тестирования. Эта эмуляция моделирует состояние сети, напрямую работая с сетевыми пакетами. Это делает возможным моделирование проводных или беспроводных сетей и фильтрацию на уровне пакетов.

Возможность формирования больших наборов тестовых данных

Аспект, который часто остается без внимания – объем данных, хранящихся в СУБД. Чем больше там данных, тем больше вероятность того, что запросы будут выполняться медленно, провоцируя взаимные блокировки. Visual Studio Premium и Ultimate обеспечивают возможность формирования тестовых данных. Это позволяет создавать и тестировать большие объемы реалистичных данных, прилагая для этого очень незначительные усилия (рис. 7).

Figure 7 - Автоматическое формирование тестовых данных

Составление отчетов

Visual Studio 2010 представляет новый набор отчетов, которые помогают анализировать отдельные прогоны тестов и публиковать результаты для заинтересованных сторон. Аналитическая мощь Microsoft Excel позволяет быстро и без труда сравнивать прогоны тестов, как показано на рис. 8. В данном представлении можно видеть время ответа страницы для заданной операции – и в момент, когда производительность приложения падает, и когда она повышается.

Scenario	Test Case	Request	Baseline	Comparison Run	Response Goal	% Change from Baseline	% from Goal
Customers	ShoppingCustomer	http://ibuyspyweb/IBuySpy/ShoppingCart.aspx (POST)	0.03	0.04	---	-53%	---
Customers	ShoppingCustomer	http://ibuyspyweb/IBuySpy/Checkout.aspx (POST)	0.01	0.01	---	-1%	---
Customers	BrowsingCustomer	http://ibuyspyweb/ibuyspy	0.02	0.02	---	0%	---
Customers	ShoppingCustomer	http://ibuyspyweb/IBuySpy/AddToCart.aspx	0.02	0.02	---	3%	---
Customers	BrowsingCustomer	http://ibuyspyweb/ibuyspy/ProductDetails.aspx	0.02	0.01	---	12%	---
Customers	BrowsingCustomer	http://ibuyspyweb/ibuyspy/productslist.aspx	0.01	0.01	---	12%	---
Customers	ShoppingCustomer	http://ibuyspyweb/ibuyspy/Login.aspx (GET)	0.02	0.01	---	15%	---
Customers	ShoppingCustomer	http://ibuyspyweb/ibuyspy/Login.aspx (POST)	0.03	0.03	---	18%	---
Customers	ShoppingCustomer	http://ibuyspyweb/IBuySpy/productslist.aspx	0.03	0.02	---	28%	---
Customers	ShoppingCustomer	http://ibuyspyweb/ibuyspy	0.03	0.02	---	33%	---
Customers	ShoppingCustomer	http://ibuyspyweb/IBuySpy/ProductDetails.aspx	0.03	0.02	---	49%	---

Рисунок 8 - Сравнение выполнения страницы

На рис. 9 представлено сравнение транзакций, т.е. продолжительности выполнения набора шагов, образующих пользовательский сценарий. На данном рисунке этими сценариями являются вход в систему, заказ элемента и выход из системы.

Отчеты о времени выполнения счетчика производительности, ошибках, общих результатах тестирования и другие – все это также стандартные отчеты. Их фильтрация обеспечивает необходимый уровень детализации. Выявив узкие места, можно перейти в код и внести необходимые изменения для повышения производительности.

Scenario	Test Case	Transaction	Baseline	Comparison Run	% Change from Baseline
Customers	ShoppingCustomer	UpdateCart	0.03	0.06	-114%
Customers	ShoppingCustomer	AddToCart	0.02	0.02	4%
Services	TestInstantOrderService	Create Client	0.03	0.03	7%
Customers	ShoppingCustomer	Checkout	0.04	0.03	14%
Customers	ShoppingCustomer	Login	0.03	0.03	18%
Customers	ShoppingCustomer	Browse	0.06	0.04	40%
Services	TestInstantOrderService	Order Item	1.12	0.34	69%

Рисунок 9 - Продолжительности транзакций

ТЕСТИРОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ДЛЯ ВСЕХ

Инструменты тестирования производительности Visual Studio обеспечивают зрелую, стабильную, расширяемую платформу для выполнения тестирования, анализа и использования результатов тестирования. Тесная интеграция со средой разработки позволяет дать практическое применение результатам тестирования, они не остаются просто красивым отчетом. Группы могут анализировать данные, переходить к коду, являющемуся причиной возникновения узких мест производительности, корректировать его и повторно выполнять тест для проверки эффекта от внесенных изменений. Независимо от того рассчитано ли приложение на большое или ограниченное число пользователей, является ли оно ответственным или нет, эти инструменты помогают повысить производительность ПО, не прибегая к дорогостоящим инструментам и не требуя особой подготовки для работы с ними.

Устранение ошибки, которую нельзя проследить

с IntelliTrace™

На отладку кода уходит неимоверное количество времени, даже самые простые ошибки требуют внимания. Новые инструменты отладки в Visual Studio 2010 Ultimate обеспечивают успех любого сеанса отладки. Но, более того, теперь разработчики могут делать это намного быстрее, чем раньше. Неуловимые невоспроизводимые и неустойчивые ошибки могут устраняться быстро, что способствует экономии времени и денег и повышению удовлетворенности пользователей.

Морис Вилкес, разработчик и создатель первого компьютера с сохраняемой программой EDSAC, однажды сказал: «...ко мне пришло полное осознание того, что добрую часть отведенной мне жизни я потрачу на поиск ошибок в своих собственных программах».[1]

Я уверен, что для мистера Вилкеса это было довольно грустное открытие, как и для большинства программистов. В конце концов, большинство из нас приходят в этот бизнес, чтобы создавать, а не отлаживать. Однако все мы знаем, что частью процесса приведения в жизнь нового ПО является устранение всех ошибок или, по крайней мере, рисков. Также нам известно, что долгие периоды отладки могут воспрепятствовать созданию шедевра.

По мере того, как инструменты разработки ПО с годами становились все более развитыми, самые существенные улучшения происходили в сфере отладки и отладчиков. Если вам приходилось выполнять отладку, вставляя в свой код выражения print, чтобы иметь возможность видеть значения переменных, тогда вы понимаете, что отладка по нажатию клавиши F5, окна просмотра, стеки вызовов и прочее – это шаг на сотни световых лет вперед от тех дней. Однако одна фундаментальная черта отладки осталась неизменной: отладка – это линейный процесс, и в нем можно продвигаться только вперед.

В Visual Studio обычный сеанс отладки подразумевает использование отладчика для пошагового выполнения кода, во время которого разработчик отслеживает состояние переменных и поток выполнения в попытке определить причину ошибки. Для разработчика это означает задание точек останова в коде выше того места, где возникает ошибка.

Фокус в том, чтобы задать точку останова достаточно близко к ошибке и не тратить часы на пошаговое выполнение кода, не имеющего никакого отношения к проблеме, надеясь при этом, что точка установлена достаточно рано, чтобы выявить причину ошибки, и процесс отладки не придется запускать заново. Общеизвестно, что во многом это метод проб и ошибок, и даже если точка останова задана так, что ошибка воспроизводится в ходе пошагового выполнения, корень проблемы может находиться за много шагов до точки останова. Это приводит к безрезультатным играм с отладчиком: после каждого неправильного шага приходится приводить все в исходное состояние и начинать заново. И даже если вам повезло найти местоположение причины ошибки, извлечь из кода в месте возникновения ошибки достаточно сведений, чтобы разобраться, что пошло не так, может быть очень сложно.

БОЛЕЕ МОЩНЫЕ ВОЗМОЖНОСТИ ОТЛАДКИ

А что если можно было бы выполнять отладку так же, как мы смотрим фильм или телевизионную передачу в цифровой записи? Если попадается какой-то неинтересный момент, есть возможность просто перемотать его; если что-то пропустили, можно вернуться назад. И что если появилась бы кнопка «More Info» (Подробнее), по нажатию которой можно получить более подробные сведения о том, что происходило в момент, когда возникла ошибка? Visual Studio 2010 Ultimate обеспечивает эти возможности.

IntelliTrace – новый инструмент Visual Studio 2010 Ultimate, позволяющий записывать выполнение кода и затем взаимодействовать с записью различными способами, чтобы упростить выявление и устранение ошибок. Для записи сведений о ходе выполнения приложения IntelliTrace использует один из адаптеров диагностических данных (diagnostic data adapters, DDA). Степень детальности сбора данных настраивается посредством конфигурации DDA, поэтому можно получить коллекцию довольно дискретных или намного более детальных данных. Подробнее DDA рассматриваются в этом разделе далее.

Чтобы использовать функциональность IntelliTrace, ее необходимо включить и убедиться, что она обеспечивает сбор необходимых данных. Для этого перейдите в меню Tools—Options (Инструменты—Опции) и под узлом IntelliTrace выберите категорию General (Общие). Окно общих настроек позволяет включать или выключать IntelliTrace и выбирать то, какие данные будут фиксироваться. Чем больше сведений отслеживается, тем больше вероятность снижения производительности приложения в ходе отладки, но часто эти неудобства стоят того, чтобы собрать все необходимые для выявления ошибки данные.

СБОР ДАННЫХ

Когда IntelliTrace включен, он обеспечивает сбор данных при каждом проходе отладки. Взаимодействие с результатами, предоставляемыми IntelliTrace, происходит несколькими способами. Если ошибка возникает в ходе выполнения теста, окно редактора кода будет открыто в соответствующей точке. В окне IntelliTrace будут выведены все данные, полученные от DDA в ходе выполнения вплоть до момента ошибки. Вы сможете перемещаться по журналу выполнения кода, как будто по цифровой видеозаписи: переходить вперед, назад, приостановить выполнение, остановить, перемотать и т.д.

1 Maurice Wilkes, Memoirs of a Computer Pioneer, MIT Press

Как и цифровую видеозапись, мы не можем изменить записанный ход выполнения кода – в конце концов, это уже случившийся факт, и мы не можем менять прошлое – но теперь стало намного проще проверять, как выполнялся код. Если что-то пропустили, просто «перемотайте» к необходимому событию и пересмотрите, что произошло. Это позволяет определять причины ошибки быстрее и проще.

На рис. 1 представлен пример выполнения IntelliTrace и результаты, которые может ожидать увидеть разработчик. Из этого конкретного примера видно, что просто просматривая журнал событий, можно определить момент загрузки пользовательского элемента управления и то, что он выполнял чтение данных cookie-файла, чего в данном случае быть не должно.

Рисунок 1 - Выполнение и результаты IntelliTrace

Другой способ работы с журналом IntelliTrace – вручную загрузить его в отладчик. При выполнении IntelliTrace всегда сохраняет данные журнала как файл. Кто угодно, у кого есть IntelliTrace, можете подключить отладчик Visual Studio к этому файлу и воспроизвести весь ход выполнения приложения со всеми возможностями навигации IntelliTrace, даже если находится на другом компьютере. Это исключительно замечательная возможность, благодаря ей результаты IntelliTrace могут передаваться на рассмотрение другим специалистам. Это особенно полезно, если нет возможности воспользоваться отладчиком. Видеоролик «Introducing IntelliTrace» (Введение в IntelliTrace), размещенный по адресу http://youtu.be/fLBpZNXs-Lw демонстрирует, как IntelliTrace может помочь в выявлении и устранении дефектов.

IntelliTrace – замечательный инструмент для индивидуальных разработчиков, которым необходимо устранить ошибки, чтобы можно было продолжить написание кода. Но не только это. Большая часть ПО сегодня разрабатывается группами. Группы могут принимать различные формы и размеры, но обычная группа разработки ПО как минимум включает разработчиков и тестировщиков. Разделение ответственности за устранение ошибок между несколькими ролями может усложнить вынесение резолюций по ошибкам, потому что означает выявление большего числа ошибок. потому что упор делается не только исключительно на ошибки, а это случается только на компьютере разработки.

УСТРАНЕНИЕ БОЛЬШЕГО КОЛИЧЕСТВА ОШИБОК

Как участник группы разработки ПО, скажите, какую резолюцию по ошибке вам приходится слышать чаще всего? «Не получилось воспроизвести», «не ошибка» или «не могу исправить»? Если расставить все резолюции по ошибкам в порядке по частоте их использования, где окажется резолюция «устранена»? А если бы можно было передвинуть «устранена» в начало списка? Как бы изменился процесс разработки ПО, если бы резолюции «не ошибка» и «не могу исправить» стали большой редкостью, а «не могу воспроизвести» и «невоспроизводимая» вообще вышли из употребления?

Чтобы переместить «исправлена» в начало списка используемых резолюций, надо взглянуть на то, как выполняется тестирование ПО и информирование об ошибках. В большинстве сред сегодня группа тестирования, или контроля качества, использует набор инструментов, который позволяет им выполнять их работу, но эти инструменты редко интегрируются с инструментами, используемыми группой разработки. Это первый разлом в системе и вклад в дорогостоящий пинг-понг ошибками (постоянная передача ошибок туда и обратно, когда тестировщик находит ошибку, разработчик не может воспроизвести ее, тестировщик находит ее снова и т.д.). В идеальном мире группа разработки и группа тестирования использовали бы интегрированные инструменты, обеспечивающие сплошной поток данных между этими двумя группами.

Основная масса выполняемого сегодня тестирования является функциональным. Функциональное тестирование подразумевает тестирование определенного действия или функции кода. Как правило, функциональный тест создается, чтобы определить, выполняется ли то или иное требование, реализована ли та или иная возможность, может ли пользовательская история быть выполнена без ошибки. В настоящее время около 70% функционального тестирования выполняется вручную. То есть тестировщик ПО пишет сценарий, чтобы выполнить ряд шагов для проведения проверки. Это означает, что мы доверяем отдельному тестировщику сбор сведений об ошибке, которые позволят разработчику ее устранить.

Visual Studio 2010 представляет абсолютно новый продукт, специально созданный, чтобы помощь тестировщикам в решении этих проблем. Microsoft Test Manager 2010 для тестировщиков – то же самое, что Visual Studio для разработчиков. То есть Visual Studio является IDE (интегрированная среда разработки), а Test Manager – ITE

Рисунок 2 - Интерфейс центра тестирования

(интегрированная тестовая среда). На рис. 2 показан интерфейс центра тестирования, который тестировщики используют для создания тестовых случаев, организации планов тестирования, отслеживания результатов тестирования и формирования отчетов об ошибках при выявлении дефектов. Test Manager, интегрированный с Team Foundation Server, создан с целью повысить продуктивность тестировщиков.

КЛЮЧ – ДИАГНОСТИКА

Ядром возможности Test Manager по сбору сведений об ошибках являются адаптеры диагностических данных (DDA – см. врезку «Адаптеры диагностических данных»). Ранее было рассмотрено, как IntelliTrace работает с одним из этих DDA. Test Manager использует этот и многие другие DDA для выполнения большинства тестовых случаев и получения практически применимых отчетов об ошибках в случае выявления ошибки в ходе тестирования, в формировании которых требуется очень незначительное участие со стороны тестировщика. Чтобы дать определение практически применимому отчету об ошибке, рассмотрим типовой отчет об ошибке, предоставляемый разработчиком. Важно помнить, что во многих случаях тестировщик, заполняющий отчет об ошибке, не является техническим специалистом. Тестировщик записывает сведения, по его мнению важные или полезные для того, кто будет устранять эту ошибку. Проблемы начинаются с этого момента. Можно подумать, что разработчики прибыли с Марса, а тестировщики – с Венеры. Роли этих индивидуумов, что важно для них, и языки, на которых они говорят, настолько различны, что они вполне могли бы быть обитателями разных планет. Как можно ожидать, что тестировщик предоставит именно те данные, которые необходимы разработчику для выполнения его работы? В результате, обычно в отчете об ошибке не хватает деталей, которые необходимы разработчику для понимания ошибки, выявления ее причины и устранения.

С другой стороны, практически применимый отчет об ошибке – это отчет, полный данных, которые помогут разработчику понять ошибку и позволят ему предпринять немедленные меры по ее устранению. Практически применимый отчет об ошибке – это результат выполнения тестового случая в Microsoft Test Manager 2010, выявления дефекта и применения инструмента для создания и формирования нового отчета об ошибке. Для начала тестировщик запускает Test Manager и выбирает тестовый случай для выполнения. Тестовый случай описывает последовательности шагов, которые должны быть выполнены, и проверки, проводимые в ходе одного или более из этих шагов. Когда тестовый случай выполняется, интерфейс Test Manager меняется

Адаптеры диагностических данных

Visual Studio Agents 2010 – это набор технологий в составе Visual Studio 2010, который позволяет выполнять задачи от лица одного или более пользователей. В контексте тестирования и отладки агентом, имеющим наибольшую важность, является агент тестирования, который включает набор адаптеров диагностических данных (DDA).

Существует несколько DDA, каждый из которых служит разным целям. В ходе выполнения теста DDA могут использоваться для:

- захвата видеоизображения с компьютера тестировщика.
- записи шагов, осуществляемых в ходе выполнения теста.
- сбора сведений о системе с любого или всех компьютеров, участвующих в тестировании.
- записи действий – регистрации нажатий клавиш и щелчков мыши, осуществляемых тестировщиком.
- записи файла IntelliTrace, который впоследствии может использоваться для воспроизведения событий, происходящих в хоте выполнения теста.

Это не полный список DDA и их возможностей. Есть DDA, имеющие самую большую ценность при заполнении отчета об ошибке; есть DDA, обеспечивающие автоматизацию сбора данных, которые превращают обычный отчет в развернутый практически применимый отчет об ошибке.

Рисунок 3 - Работа Test Runner

для поддержки работы Test Runner (модуль для выполнения тестов) (рис. 3). Этот инструмент превращается в закрепляемый интерфейс в виде боковой панели. С его помощью тестировщик без труда реализует пошаговое выполнение тестового случая, имея перед собой и Test Runner, и тестируемое приложение.

ВЫПОЛНЕНИЕ РУЧНОГО ТЕСТА

В начале записи тестового случая предлагается опция записи действий. Эта запись осуществляется DDA и регистрирует все щелчки мыши и нажатия клавиш, осуществляемые тестировщиком. При последующих выполнениях этого тестового случая запись действий может использоваться для пропуска части этапов тестового случая.

Когда выполнение теста начинается, Test Runner выводит на экран последовательность шагов, которые должны быть выполнены. Каждый шаг по его завершении отмечается как Passed (Пройденный) или Failed (Непройденный). По мере выполнения шагов тестировщик может указывать результат каждого из них, обеспечивая подробные сведения о ходе выполнения теста.

Если тестировщик обозначает шаг как непройденный, инструмент предлагает добавить примечание о сбое, т.е. указать подробнее, что заставило указать этот шаг как таковой. Кроме того, Test Manager обеспечивает возможность сделать снимок экрана или его части, чтобы показать, что пошло не так в ходе данного шага. Это замечательная возможность для сохранения сообщений об ошибке и диалоговых окон.

После того как дефект выявлен, и шаг обозначен как «непройденный», необходимо создать новый отчет об ошибке. Это также может быть сделано прямо посредством интерфейса Test Manager. Когда новый отчет об ошибке создается, к нему автоматически подключаются все артефакты данных, собранные DDA. Сюда входят шаги, выполненные в ходе тестового случая, системная информация тестовой среды и также, возможно, видеозапись экрана компьютера тестировщика, все снимки экрана, сделанные в ходе выполнения теста, и файл IntelliTrace, с помощью которого разработчик сможет увидеть, что именно происходило в ходе выполнения теста. Все что остается сделать тестировщику – это ввести название нового отчета об ошибке и щелкнуть Save (Сохранить) и Close (Закрыть), все остальное уже сделано за него. Посмотрите демонстрацию «Creating Actionable Bugs with Microsoft Visual Studio Test Professional 2010» (Создание практически применимых отчетов об ошибках в Microsoft Visual Studio Test Professional 2010) по адресу http://youtu.be/P4meKiOojw8.

ПОЛЕЗНЫЙ ОТЧЕТ ОБ ОШИБКЕ

Практически применимый отчет об ошибке исключительно полезен. Вам больше не надо беспокоиться о том, как тестировщик задокументирует шаги для воспроизведения ошибки, или о сведениях о системе – эти данные собираются и добавляются в отчет об ошибке автоматически. Вам больше не надо идти в другую комнату и просить тестировщика выполнить тестовый случай еще раз, чтобы вы могли было посмотреть и увидеть собственными глазами, что происходит – снимок экрана и видео делают это за вас. И с журналом IntelliTrace, прикрепленным к отчету, можено воспроизводить все в точности, как происходило, и выполнять код по шагам так, как будто вы присутствовали в ходе тестирования. Посмотрите демонстрацию «Consuming an Actionable Bug» (Использование практически применимого отчета об ошибке) по адресу http://youtu.be/4CKGji6eks4.

Итак, мы можем самостоятельно выявлять ошибки и располагаем достаточным объемом сведений, чтобы быстро находить их причины. У нас могут быть тестировщики, которые выявляют ошибки и снабжают нас подробными сведениями, необходимыми для их воспроизведения и устранения. Это все? Продолжая добавлять код, мы знаем о том, что ошибки, которые мы нашли и устранили, по-прежнему устранены? Единственный способ быть уверенным в этом – постоянно выполнять тесты, проверяющие состояние ошибки.

Тестирование, обеспечивающее проверку и подтверждение того, что исправленные ошибки не возникли вновь, называют регрессивным тестированием. Регрессивный тест – это тест, позволяющий определить, не начал ли давать сбой код, в который уже были внесены исправления. Регрессивный тест (и другие функциональные тесты) многократно тестирует код, работоспособность которого была подтверждена в какой-то момент времени. Как и функциональные, регрессивные тесты также в основном выполняются вручную. Это массивный удар по ресурсам: тестировщики проводят несчетное число часов, тестируя работоспособную функциональность, исключительно для того, чтобы убедиться, что все пока еще работает. Представьте, насколько более продуктивными были бы тестировщики, если бы могли сконцентрировать свои усилия на создании и выполнении новых тестовых случаев, охватывающих еще непротестированные части системы, а не тратить свое время на тестирование одного и того же кода снова и снова.

КОДИРОВАННЫЕ ТЕСТЫ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Visual Studio 2010 представляет новую возможность – кодированные тесты пользовательского интерфейса, которые позволяют автоматизировать тестирование пользовательского интерфейса приложения. Кодированные тесты пользовательского интерфейса работают аналогично модульным тестам в том смысле, что пишутся на языке программирования, таком как Visual Basic или C#, и осуществляют некоторые действия или последовательности действий, проверяют одно или более утверждений и легко повторяемы с предсказуемыми результатами. Отличие от модульных тестов в том, что тогда как модульные тесты предназначены для тестирования отдельного модуля кода, такого как вызов одного метода, кодированный тест пользовательского интерфейса автоматизирует функциональный тест, включая проверку результатов тестирования.

Visual Studio упрощает для вас создание кодированных тестов пользовательского интерфейса. Для этого можно использовать либо средство записи действий, которое позволяет фиксировать действия тестировщика, нажатия клавиш, щелчки кнопки мыши и т.д., либо можно импортировать существующую запись действий, созданную в Test Manager 2010 в ходе выполнения тестового случая. В любом случае, записанные действия преобразовываются в код на Visual Basic или C#. Сделав запись, можно перейти к коду и добавить в тест утверждения с помощью построителя кодированных тестов пользовательского интерфейса. В этом очень простом интерфейсе есть возможность выбирать области пользовательского интерфейса, которые необходимо проверить. Обозначив область, вы получаете список свойств для этого элемента пользовательского интерфейса, из которого можно выбрать свойства для проверки. Демонстрационное видео «Creating a Coded UI Test» (Создание кодированного теста пользовательского интерфейса) можно найти по адресу http://youtu.be/KYyj5Dfp8iE.

Автоматизация функциональных тестов, включая регрессивные, позволяет высвободить тестовые ресурсы и направить их на создание и выполнение новых тестов, обеспечивая полное покрытия кода тестами. Это как введение автоматизированного тестировщика в группу: он никогда не опаздывает, никогда не жалуется, работает с неимоверной скоростью и берет на себя дополнительную работу по тестированию со скоростью создания новых кодированных тестов пользовательского интерфейса.

ПОБЕДА В БОРЬБЕ ПРОТИВ ДЕФЕКТОВ

Дефекты одолевают нас с того самого первого момента, когда люди начали использовать компьютеры. На заре информационных технологий их было легко выявлять, потому что они имели физическую природу и были материальными (например, мухами). Но по мере того как вычислительные системы становились все более сложными, дефекты становилось все сложнее выявлять и устранять. Visual Studio 2010 делает большой шаг вперед, предоставляя инструменты и данные, необходимые не только для выявления ошибок, но также для сотрудничества с другими участниками группы для обеспечения быстрого и эффективного выявления и устранения дефектов. После того как ошибки исправлены, кодированные тесты пользовательского интерфейса позволят подтверждать, что они не возникли вновь, а если это случится, об этом сразу же станет известно.

Понимание своих систем

с Visual Studio 2010

Большинство проектов начинается с уже существующего кода: либо выполняется доработка существующего приложения, либо приложение переписывается с целью модернизации. В любом случае, чаще всего документации по существующей системе нет, она не отвечает текущему состоянию либо просто ошибочна. Поскольку связь с создателями системы утрачена, новой группе приходится разбираться в ней, не имея никакой отправной точки. Visual Studio 2010 обеспечивает эту отправную точку и позволяет группам выйти на необходимый уровень производительности в кратчайшие сроки.

Get the free mobile app for your phone
http://gettag.mobi

Visual Studio 2010 Ultimate представляет новые инструменты разработки архитектуры, которые помогут повысить окупаемость инвестиций в разработку систем и существенно сократить затраты на сопровождение и обновление существующих систем. Задача инструментов разработки архитектуры – обеспечение возможности визуализации приложений и понимания потока данных участниками группы. Это обусловливает значительное снижение затрат и усилий на введение новых участников в группу и понимание приложений. При использовании инструментов разработки архитектуры группы смогут уделять больше внимания новым возможностям и тратить меньше времени на устранение существующих проблем. Теперь существующие приложения могут приносить дополнительные выгоды при меньших затратах, поскольку будут функционировать дольше, и обслуживать их будет проще.

СОКРАЩЕНИЕ СТОИМОСТИ СОПРОВОЖДЕНИЯ СУЩЕСТВУЮЩИХ СИСТЕМ

Сопровождать существующие системы дорого. На их поддержку уходит больше денег и усилий, чем на разработку новых. По мере роста срока службы системы, возрастает и стоимость ее сопровождения. Даже обслуживание недавно выпущенной системы может требовать больших вложений в случае передачи ее новым разработчикам, которым требуется время на понимание приложения. В этом и заключается обслуживание: понимание системы и потенциального влияния на нее любых изменений. Visual Studio 2010 снижает расходы, связанные с сопровождением, потому что сокращает количество времени, необходимого на понимание системы.

Традиционные для любой системы проблемы усиливаются при обслуживании уже существующей системы. Многие из этих проблем связаны с документацией, а точнее с ее отсутствием. Комментариев в коде нет, они непонятны или неточны. Документация системы обновляется редко, поэтому исходная архитектура редко соответствует готовому продукту. В ходе работ по сопровождению документация не оформляется вообще. Иногда намного большую проблему представляет «спагетти»-код: неупорядоченный код, который пронизывает многие части системы. Из-за этих проблем разработчики могут проводить дни или даже недели в попытках согласовать документацию с приложением. Microsoft Visual Studio 2010 Ultimate предоставляет две возможности для сокращения времени, уходящего на эти упражнения: обозреватель архитектуры и диаграмму последовательности действий.

Обозреватель архитектуры

Проблема с внесением изменений в существующие системы состоит в том, что часто мы даже не представляем, какие негативные последствия это может иметь, до тех пор пока не становится слишком поздно. Это очень затратная по времени и средствам ситуация, которая может приводить к необходимости многочисленных исправлений сразу после выпуска версии. Для решения данной проблемы в Visual Studio 2010 Ultimate включен обозреватель архитектуры. Обозреватель архитектуры обеспечивает ряд способов быстро и просто представить структуру приложения любого уровня (метода, класса, пространства имен, сборки или решения). Также в нем имеется представление зависимостей, с помощью которого можно увидеть, какие элементы взаимосвязаны с элементом, в который вносятся изменения.

С обозревателем архитектуры разработчик получает возможность начинать с малого и затем постепенно расширять представление для получения полного понимания приложения или для более детального рассмотрения отдельных его частей. Используя это мощное решение для визуализации, разработчики также могут сообщать о рисках и потенциальных последствиях в графическом формате. Этот процесс представлен на рис. 1-4 на следующих страницах.

Рисунок 1 - Высокоуровневое представление приложения.

На рис. 1 изображено высокоуровневое представление приложения в момент, когда разработчик приступает к рассмотрению BlogEngine.Core.dll. В данном случае разработчик желает сосредоточиться на изменении, в котором нуждается пространство имен BlogEngine.Core. Переходя на более глубокие уровни, они получают представление, показанное на рис. 2.

Рисунок 2 - Покомпонентное представление пространства имен BlogEngine.Core.

Рисунок 3 - Метод Comment и связанные с ним классы.

Теперь разработчик может видеть классы, образующие это пространство имен, и переходить глубже на уровень класса или классов, которые предполагается изменять, для анализа потенциальных последствий изменений. При переходе на все более глубокие уровни, разработчику открывается картина, показанная на рис. 3.

На данном этапе разработчик может переходить далее на более глубокие уровни детализации или переключаться к диаграмме зависимостей, как показано на рис. 4. С помощью диаграммы зависимостей разработчик может формировать визуальное представление схемы зависимостей для любого кода, визуализированного в обозревателе архитектуры. По сути, эта диаграмма показывает, как классы (или методы, сборки, пространства имен и т.д.) связаны с другими классами, что позволяет разработчикам видеть отношения и потенциальные последствия изменений до их внесения. Ранее такой возможности не было. Это позволит сократить риски, обусловливаемые каждым отдельно взятым изменением.

Рисунок 4 - Представление зависимостей для методов, обращающихся к методу Comment.

Диаграммы последовательности действий

Как только разработчик определяется с тем, какие изменения должны быть внесены, ему необходимо понять последовательность обращений к методу, чтобы произвести соответствующие изменения и здесь. До выхода Visual Studio 2010 для этого приходилось организовывать пошаговое выполнение кода, расставляя точки останова соответствующим образом. Это не только отнимало массу времени, но и, в некоторых случаях, было сложно реализовать. Диаграмма последовательности действий позволяет, щелкнув метод правой кнопкой мыши, сформировать визуальное представление полной последовательности действий с указанием всех классов, с которыми взаимодействует этот метод (рис. 5).

Рисунок 5 - Диаграмма последовательности действий для метода RewritePage (фрагмент).

Моделирование на UML

Как видим, без документации или комментариев к коду разработчик может разобраться и понять имеющееся приложение в считанные мгновения. Новые инструменты анализа архитектуры помогают, прежде всего, максимально сократить неудобства, связанные с отсутствием документации по системе. Обычно процессы документирования и реализации разделены, и объединить их очень сложно. Из-за этого разделения велики шансы их рассинхронизации, что приводит к ситуациям, возникающим со многими старыми системами. Сочетание Visual Studio 2010 и Team Foundation Server 2010 устраняет эту проблему и обеспечивает возможность упреждающего проектирования, которое напрямую связывается с документацией. Это позволяет визуально переходить от проекта архитектуры к документации и к коду. При таком уровне прослеживаемости инвестиции в документирование не пропадают зря и окупаются в ходе работ по сопровождению, помогая сократить общую стоимость системы и сроки возврата инвестиций.

Visual Studio 2010 предоставляет возможность моделирования на UML 2.1 с некоторыми функциями, которые доступны только благодаря их интеграции в Visual Studio. Отправной точкой в разработке многих приложений являются диаграммы вариантов использования, потому что они обеспечивают визуальное руководство для пользователей по их взаимодействию с системой и функциям, обеспечиваемым системой. Одно из преимуществ, которое модели UML получают благодаря тесной интеграции с Team Foundation Server – возможность напрямую подключаться к одному или более рабочим элементам или формировать рабочий элемент из артефакта модели. Это превращает модели в «живые» ресурсы, которые непрерывно используются и приносят пользу, а не просто пылятся на полке.

На рис. 6 представлена диаграмма вариантов использования. На ней обведен вариант использования Add Post (Добавить публикацию), чтобы показать связанные с ним рабочие элементы. Это позволяет разработчикам перейти от графической перспективы к детальным требованиям, которые они будут реализовывать в коде, и, в конечном счете, обеспечивает плавный переход от требований к коду и обратно. Это очень важно и при сопровождении системы, потому что разработчик имеет возможность взглянуть на код и понять не только, почему та или иная возможность была добавлена в систему, но и то, как она связана с другими возможностями системы с точки зрения пользователя.

Рисунок 6 - Диаграмма вариантов использования со связанными рабочими элементами.

Также можно добавлять ссылки на другие диаграммы, такие как диаграмма компонентов, представленная на рис. 7.

Рисунок 7 - Установление связей между вариантами использования и требованиями и дизайнами.

Это простой способ объединения диаграмм с требованиями и кодом, что обеспечивает интегрированное решение, позволяющее прослеживать модели архитектуры и документы. Благодаря этому не приходится создавать общие файлы в сети, которые быстро устаревают, или беспокоится о том, что кто-то не может найти необходимый документ или модель, чтобы внести обновления. Все материалы размещаются централизованно. Кроме того, если есть желание подключить к модели документы, а не другие модели, это можно сделать, просто предоставив ссылку на документ, хранящийся в Веб или на файловом сервере. Таким образом, все создаваемые и необходимые для работы артефакты будут доступны в том контексте, в котором они были созданы. Это сокращает время на понимание приложений, благодаря чему участники группы в процессе разработки или обслуживания могут приступить к эффективной работе намного быстрее и с меньшими шансами на возникновение регрессивных ошибок.

НАЧНИТЕ ПОНИМАТЬ СВОИ СИСТЕМЫ УЖЕ СЕГОДНЯ

Visual Studio 2010 Ultimate обеспечивает группам и организациям возможность сразу же повысить ROI существующих систем благодаря сокращению времени, необходимого на понимание и обслуживание этих систем. Имея в своем распоряжении инструменты разработки архитектуры, разработчики получают возможность уделять больше внимания написанию кода и меньше времени тратить на выяснение, какие последствия любое отдельно взятое изменение будет оказывать. Также новые разработчики и группы, принимающие под свою ответственность унаследованный код, могут быстрее и без особых усилий разобраться с этим приложением и его контекстом. Большую часть времени при введении новых участников группы в курс дела или работе с существующими .NET-приложениями разработчики тратят на то, чтобы понять приложение на техническом уровне. С этими инструментами данная проблема во многом теряет свою актуальность, а производительность разработчиков значительно увеличивается.

Упрощение процесса разработки

с Visual Studio 2010

Многие организации попадают в ловушку оптимизации части используемого процесса разработки, забывая о процессе в целом. Хотя такой поверхностный подход иногда и может приводить к положительным изменениям, но чаще всего он упускает большие преимущества, которые могли бы быть достигнуты в случае концентрации на процессе в целом. Visual Studio 2010 и Team Foundation Server 2010 обеспечивают огромные возможности визуального контроля и прослеживаемости в ходе всего процесса, реально меняя положение дел в этой области, что затрагивает организацию в целом, а не просто отдельную группу.

Get the free mobile app for your phone
http://gettag.mobi

КАКОЕ ОТНОШЕНИЕ ЭТО ИМЕЕТ КО МНЕ?

Термин «процесс» по каким-то причинам стал синонимом невыполнения поставленных задач. Он символизирует бесконечные документы и рутинные операции, по большому счету, означая постоянную занятость с нулевой продуктивностью. Это может быть следствием плохо организованных процессов, которые не ориентированы на решение проблем. Однако правильно построенный процесс с соответствующим инструментальным обеспечением может сократить рабочую нагрузку, повысить эффективность, качество и возможности визуального контроля и, в итоге, повысить удовлетворенность заказчика. Visual Studio 2010 предоставляет гибкие инструменты, которые работают на, а не против вас, помогая выполнять проекты в установленные сроки с выполнением бюджета и обеспечением высокого качества. Visual Studio 2010 – это платформа для улучшений, которые имеют положительное влияние на процесс разработки ПО в целом. По сути, она упрощает процесс разработки, устраняя изолированность групп и делая их взаимодействие более эффективным и продуктивным.

ИНТЕГРИРОВАННОЕ РЕШЕНИЕ

Один из факторов, приводящий к снижению производительности и повышению издержек – интеграция инструментов. Группы используют один инструмент для управления версиями, другой – для отслеживания дефектов, третий – для управления проектом, для разработки архитектуры, для тестирования и т.д. Список инструментов, используемых группами, бесконечен; масса времени и средств уходит на обслуживание всех этих разрозненных инструментов. Кроме того, это усложняет введение в курс дел новых разработчиков, потому что им приходится осваивать все это многообразие инструментальных средств. Во многих случаях инструменты плохо сочетаются, и для этого для каждого из них предусмотрен набор временных решений. И даже если разные инструменты мирно сосуществуют, обеспечение совместимости разных версий – головная боль руководства. Решаться эта проблема может посредством непрерывного потока обновлений. Team Foundation Server 2010 (TFS) и Microsoft Visual Studio 2010 одним махом устраняют все эти проблемы, обеспечивая набор интегрированных инструментов.

Рисунок 1 - Team Foundation Server 2010 и Visual Studio 2010 – сердце интегрированного решения.

Team Foundation Server обеспечивает прослеживание задач, отслеживание дефектов, управление изменениями, тестовыми случаями и проектом в целом, как показано на рис. 1. В качестве серверной части он использует SQL Server 2008 и интегрируется с Visual Studio, Microsoft Office SharePoint Server, Microsoft Test Manager и Microsoft Expression Blend. С появлением подключаемого модуля для Eclipse даже группы, работающие в этой интегрированной среде разработки, могут пользоваться теми же преимуществами и мощью, что и разработчики в Visual Studio. И даже если эти инструменты не отвечают всем вашим требованиям, предусмотрен программный интерфейс приложения (application programming interface, API), который позволяет создавать собственные надстройки для беспрепятственной работы с Visual Studio и Team Foundation Server.

Интегрированная возможность отслеживания задач

Система отслеживания рабочих элементов в TFS без труда решает обычно предъявляемое требование – прослеживаемость. Рабочий элемент – основной элемент отслеживания. Он может представлять требование, задачу, дефект, запрос на внесение изменений или любой другой тип артефакта, интересующий организацию. Выполняя распределение задач с помощью системы отслеживания рабочих элементов, разработчики могут ассоциировать код непосредственно с назначенными им задачами. Многие группы разработки подчиняются одному типу директивы по соблюдению регулятивных норм – HIPAA или SOX, например – они требуют предоставления свидетельств рабочих потоков для аудита. Как правило, на сбор таких свидетельств уходит много времени и усилий, но все эти процессы встроены в Team Foundation Server.

Такая интеграция обеспечивает дополнительные преимущества. Например, разработчики и тестировщики могут работать в одной системе, вместо использования множества различных систем. Благодаря этой интеграции TFS имеет возможность формировать разнообразнейшие отчеты по любому заданному срезу данных. К примеру, группы могут увидеть требования, определить, какая часть работы выполнена и что осталось сделать, сколько дефектов зарегистрировано для отдельно взятого требования, сколько тестов было для него выполнено. Этот тип сведений представляет стандартный отчет Обзор описаний функциональности, показан на рис. 2.

Такой тип отчетов доступен только в полностью интегрированной системе.

Рисунок 2 - Обзор описаний функциональности.

Устранение изолированности групп

Группы – это люди, работающие вместе для достижения общей цели. Когда организации упрощают один процесс, они иногда оставляют без внимания эффект, оказываемый этим на другие части группы, или никогда не реализуют все преимущества улучшения. Возьмем, к примеру, архитектуру. Существует множество стандартов и способов документирования архитектуры, но сколько есть путей сохранить ее соответствие на протяжении всего жизненного цикла приложения? Зачем создавать замечательные архитектурные диаграммы, если никто из группы никогда ими не пользуется, и они просто «пылятся на полке»? Одно из основных преимуществ интегрированной системы в том, что данные могут использоваться в соответствующее время соответствующими людьми, и то, что это имеет значение.

Team Foundation Server – это интегрированное ядро, которое помогает руководителям проекта, архитекторам, разработчикам и тестировщикам эффективно сотрудничать и обмениваться данными. Архитекторы могут подключать создаваемые ими модели к рабочим элементам, тогда как разработчики могут сверяться с архитектурой системы, что помогает им понять, что именно они должны построить. Это обеспечивает не только более высокий уровень соответствия дизайна и реальности, но также создает платформу для обмена сведениями, посредством которой разработчики и архитекторы могут взаимодействовать. Тестировщики получают такой же уровень интеграции с группой разработки. Новые возможности управления тестовыми случаями позволяют им создавать сценарии тестирования, к которым имеют доступ и разработчики. В отчете об ошибке, формируемом тестировщиком, все данные, требуемые разработчику для выявления причины ошибки, включены и прикреплены к тестовому случаю. Нет необходимости экспортировать данные из одной системы и импортировать их в другую или заставлять разработчиков работать с множеством систем.

Использование множества систем также негативно отражается на работе руководства, потому что им приходится собирать данные из всех систем и сводить их вместе в один отчет о состоянии. Это не только забирает массу времени, но и чревато неточностями. С обеспечиваемой TFS возможностью отслеживания рабочих элементов руководители практически в реальном масштабе времени получают данные о том, над чем и когда работает каждый из участников группы. Им известно, кто какую задачу выполнил в заданный день, неделю или месяц, и кто над чем работает в настоящий момент. Больше нет необходимости в электронных письмах с запросами о состоянии, и руководителям проектов не надо каждую неделю часами заниматься сбором сведений для создания отчетов о состоянии.

Обеспечение возможности визуального контроля

Во многих организациях есть проблемы с эффективным взаимодействием групп разработки и заказчиков. Они регулярно обсуждают проект, но все равно часто упускают ценную информацию. И подчас группа проводит массу времени, отвечая на вопросы заказчика, тогда как необходимо лишь обеспечить простоту доступа заказчика к этим сведениям. Благодаря интеграции со службами отчетов SQL Server и сервером SharePoint Microsoft Office TFS обеспечивает механизм ответа на многие вопросы пользователей без участия группы разработки. На рис. 3 представлена стандартная панель мониторинга в MOSS 2007. На основании сведений этой панели мониторинга группы могут сделать выводы о том, будут ли выполнены ими сроки выпуска версии. Диаграммы невыполненных задач и хода выполнения задач обеспечивают два взгляда на ход работы группы: по часам и по объему работ, который необходимо выполнить. Диаграмма хода реализации пользовательских историй показывает, сколько требований реализовано (т.е. готовы к поставке заказчику). Диаграмма тенденций возникновения проблем за неделю показывает для группы объем Активных задач (тех, над которыми ведется работа), Выполненных (завершенных, но не отмеченных в системе как выполненные) и закрытых (полностью завершенных).

Такие представления обеспечивают точные и своевременные сведения, что дает ответы на вопросы обо всех аспектах процесса разработки. Группам больше не приходится каждую неделю тратить часы на создание отчетов. Все они могут быть настроены, когда возникает необходимость отчитаться по любым данным, фиксируемым TFS. Всеобъемлющие и детальные отчеты также позволяют выявлять узкие места процесса в целом.

Рисунок 3 - Панели мониторинга SharePoint

Выявление и устранение узких мест

Вы укладываетесь в установленные сроки разработки? Если нет, в чем причина отставания? Требуется больше разработчиков или, возможно, тестировщиков? Во многих ситуациях на эти вопросы ответить сложно, но TFS делает эти вопросы неактуальными, поскольку все ответы прямо перед глазами. На рис. 4 представлен типовой отчет о выполнении работ с обозначением идеального и фактического хода выполнения.

Глядя на этот отчет, группы могут понять общие тенденции, начиная с самого начала проекта, и то когда вероятнее всего он будет завершен. В данном случае фактическая кривая хода выполнения работ более крутая, чем идеальная. Это означает, что группа выполнит работы в срок, и это можно спрогнозировать уже в первые несколько дней работы над проектом. Другие отчеты демонстрируют состояние каждого отдельно взятого требования, на основании чего группы могут принимать решения о вводе дополнительных разработчиков или тестировщиков. Благодаря наглядному представлению информации группы имеют возможность предпринимать действия для устранения проблем на самых ранних этапах процесса.

Рисунок 4 - Отчет о выполнении работ

Интеграция тестировщиков

Очень часто в организациях тестировщики и разработчики разнесены в разные группы, что препятствует их эффективному сотрудничеству. В результате этого разработчики часто не знают, чем занимаются тестировщики (им не известны ожидания тестировщиков относительно создаваемого ими кода), и тестировщики не представляют, над чем работают разработчики, до тех пор пока не получат код на тестирование. Но так не должно быть.

Новые возможности управления тестовыми случаями, обеспечиваемые TFS и Microsoft Test Manager (включен в пакет VS Ultimate и VS Test Professional), играют важную роль в интеграции групп. Теперь разработчики могут видеть тестовые случаи, создаваемые тестировщиками. Когда тестировщик формирует отчет об ошибке, ассоциированной с тестовым случаем, он прямо к нему подключает детальные журналы трассировки. Это полностью меняет правила игры взаимодействия разработчиков и тестировщиков. С выходом VS 2010 разработчики и тестировщики получили возможность работать сообща, как группа, эффективно и продуктивно выявляя и устраняя дефекты.

СКВОЗНОЙ ПРОЦЕСС ВЫГОДЕН ВСЕМ

Эффективное сотрудничество и взаимодействие групп оказывает положительный эффект во всем. Единый процесс реально сокращает объем работ, который группам приходится выполнять, и повышает продуктивность. Уход от разрозненных инструментов означает устранение избыточности и повышение эффективности благодаря тому, что теперь все можно найти в одном инструменте. Оптимизация выходит далеко за рамки простого сокращения номенклатуры используемых инструментов – мы являемся свидетелями интеграции инструментов, которая обеспечивает эту мощь. Инструменты должны быть целенаправленными, но также они должны поддерживаться той же платформой, что и результаты, доступные всем, кто в них нуждается. Team Foundation Server 2010 и Microsoft Visual Studio 2010 поставляют платформы для улучшения процесса.

Создание качественных приложений

с Visual Studio 2010

Вопросы качества имеют первостепенное значение в деле производства ПО. Уже давно стало очевидным, что чем позже дефект выявляется, тем дороже его устранять. Например, для сообщества, использующего гибкие подходы к разработке, «ранний сбой» – это своего рода боевой клич и, следовательно, непрерывная интеграция и тестирование рассматриваются как фундаментальные практики. Visual Studio 2010 обеспечивает агрессивный набор нововведений в области тестирования и обеспечения качества, благодаря которым группы разработки ПО получают возможность поставлять превосходные результаты.

Get the free mobile app for your phone
http://gettag.mobi

Согласно отчетам Национального института стандартов и технологий (2002)[1] убытки экономики США, обусловленные некачественным ПО, составляют около $59 млрд. ежегодно. Эти деньги теряются из-за низкой производительности и неэффективного расходования ресурсов. Исследование показало, что одной трети этих потерь можно было бы избежать за счет улучшения инфраструктуры тестирования. Как показывает отчет Chaos Report группы аналитиков Standish Group (2009)[2] (рис. 1), количество неудачных или проблемных проектов опять выросло в прошлом году. Панацеи от этого нет, но организации могут помочь себе, используя инструменты и процессы, обеспечивающие наилучшие шансы успешного выполнения проектов. В конце концов, единственным настоящим мерилом качества является то, нравится окончательный продукт заказчику или нет. Проще говоря, обычно это означает, что продукт делает то, что хочет заказчик (обеспечивает большую продуктивность и эффективность ежедневных рутинных операций) и не имеет дефектов, негативно влияющих на его эффективность. Visual Studio 2010 обеспечивает группам разработки максимально возможные шансы создания высококачественного ПО.

Рисунок 1 - Отчеты Chaos Reports не демонстрируют существенных улучшений в показателях успешности проектов.

ВЫПОЛНЕНИЕ ТРЕБОВАНИЙ ЗАКАЗЧИКА

Независимо от используемого инструмента собрать полностью все требования заказчика сложно, поскольку основную проблему представляют непредсказуемые взаимодействия пользователей с ПО. Но при наличии надежной прослеживаемости от требований к тестовым случаям, коду, дефектам и результатам тестирования стабильность может быть достигнута, а также обеспечен мониторинг и формирование отчетов с количественными показателями выполнения требований заказчика. Из полного отчета с результатами прослеживания группы могут видеть, какие требования являются наиболее приоритетными, ход выполнения реализации требований (количество выполненных и пока невыполненных задач) и качество выполнения требований. Если с требованием ассоциировано множество дефектов, то завершена ли его реализация, не имеет значения, поскольку согласно общему определению качества это означает, что требования заказчика не реализованы. TFS обеспечивает необходимую степень прослеживаемости и простоту анализа данных посредством полностью настраиваемых стандартных отчетов.

На первый взгляд, представленный на рис. 2 отчет довольно прост, но он объединяет в себе множество данных, таких как количество часов, затраченных на реализацию требования, количество тестовых случаев, реализованных для проверки требований, результаты тестирования для каждого требования и количество активных и устраненных дефектов для каждого требования. Больше не надо угадывать, насколько хорошо реализованы требования пользователя – у нас есть вполне определенный ответ.

Stories Overview

Helps you track how far each user story has been implemented. Shows each story's actual number of hours of work remaining and completed, its acceptance test results, and the number of bugs that are linked to each story.

Related Reports
- Bug Status
- Status on All Iterations
- Stories Progress
- Test Case Readiness
- Test Plan Progress

Completed
Remaining
Passed
Failed
Not Run
Active
Resolved

Title	Work Progress: % Hours Completed	Work Progress: Hours Remaining	Test Status: Tests	Test Status: Test Results	Test Status: Bugs
As a new customer, I want to order a meal.	79 %	12	3	22 % 62 %	1 2
As a new customer, I want to get an idea of what DinnerNow offers with a brief glance at the web site.	56 %	11	0		6
As a customer who has completed an order, I want DinnerNow to keep track of my meal preferences so that I can order the meal that I want more quickly in the future.	100 %	0	2	31 % 69 %	2 1
As a returning customer, I want to order one of the meals that I've recently ordered.	81 %	7	2	31 % 69 %	
As a new customer, I want to choose a meal from a specific provider.	60 %	2	0		
As a meal provider, I want to display photos and information about each of my meals so that I can distinguish them from other meal providers.		10	0		

Рисунок 2 - Обзор описаний функциональности.

1 Национальный институт стандартов и технологии. (2002). Отчет о планировании 02-3, Экономические последствия неадекватной инфраструктуры тестирования программного обеспечения. Министерство торговли США.

2 Standish Group, (2009). Chaos Report.

3 ШАГА К ДОСТИЖЕНИЮ КАЧЕСТВА

Создание качественных приложений включает очень много этапов, но весь процесс можно упростить до трех основных шагов: проектирование, разработка и тестирование.

Качество начинается с проектирования...

Сегодня многие инструменты разработки архитектуры отделены от инструментов разработки, сбора требований и тестирования. При таких условиях создавать архитектуру, которая не будет проигнорирована с самых первых этапов написания кода, чрезвычайно сложно. Разработчикам приходится постоянно сверяться с другими документами или инструментами и в конце концов им это надоедает. В результате дизайн архитектуры и код рассинхронизируются, утрата прослеживаемости происходит на техническом уровне. Это имеет негативное влияние и на сопровождение существующих систем. Без понимания системы обслуживать ее очень сложно. Visual Studio 2010 Ultimate включает новый набор инструментов разработки архитектуры, построенных на базе UML. Благодаря этому группы не только создают архитектуры решений прямо в среде разработки, но также подключают эти дизайны к требованиям таким образом, что позволяет без труда сверяться с ними и сохранять их актуальность.

На рис. 3 представлены примеры UML-диаграмм вариантов использования и деятельности вместе с новым связанным с качеством инструментом разработки архитектуры, диаграммой слоев.

Рисунок 3 - Диаграмма вариантов использования, деятельности и слоев.

Диаграмма слоев не только определяет архитектуру слоев (слоем может быть все, что угодно: метод, класс, пространство имен, сборка или произвольная группа элементов) и пути обмена данными, но может использоваться для их соблюдения. Если создан дизайн для достижения определенной цели и этот дизайн не соблюдается, могут быть предприняты соответствующие шаги для изменения либо дизайна, либо кода.

обеспечивается разработчиками...

Visual Studio 2010 включает многие инструменты, с помощью которых разработчики обеспечивают выполнение требований заказчика. Такие возможности как статический анализ кода (для .NET-кода и кода базы данных) помогают находить потенциальные вопросы с производительностью, стабильностью, возможностью обслуживания, масштабируемостью и безопасностью до того, как они вырастут в настоящую проблему. С помощью метрик кода разработчики могут определять самые запутанные (т.е. наименее удобные в сопровождении) участки кода и вносить изменения до того, как код превращается в «спагетти»-код. Разработчики также могут проводить полное тестирование производительности своего кода, как WPF-приложений, так и ASP.NET-приложений, обеспечивая соответствие приложения требованиям масштабируемости, предъявляемым заказчиком. На рис. 4 представлено нагрузочное тестирование. Предоставляются детальные сведения по любому аспекту любой системы, вовлеченной в тестирование. Это обеспечивает сбор достаточного объема сведений для выявления проблем с производительностью. Также можно сравнить разные прогоны нагрузочного теста для определения величины изменения характеристик производительности со временем.

Рисунок 4 - Выполнение нагрузочного теста.

Если в ходе нагрузочного тестирования выявляются проблемы с производительностью, анализ подробных отчетов по его завершении поможет точно определить, какие части приложения включают проблемный код, чтобы можно было их исправить. На рис. 5 представлен анализ пользовательской нагрузки на приложение в ходе выполнения теста. Рисунок показывает не только пользовательскую нагрузку, но и выполняемые страницы, и все исключения или другие сбои, имевшие место.

Чтобы дефекты не попали в конечный продукт, разработчики должны иметь возможность оперативно выявлять и устранять ошибки в ходе сеанса отладки. Самое последнее нововведение Майкрософт, IntelliTrace™, обеспечивает это и многое другое. IntelliTrace™ позволяет разработчикам выполнять приложение, приостанавливать выполнение в любой момент и затем перемещаться вперед и назад по временной шкале сеанса отладки для проверки состояния приложения в заданный момент времени. Записывается журнал трассировки стека, по которому можно перемещаться, что избавляет разработчиков от необходимости расставлять точки останова в надежде, что они правильно угадали место ошибки. И не только это: когда формируется исключение, разработчики могут немедленно переходить к точке возникновения исключения, определять причину его формирования и устранять ее. Сверх этого, при тестировании компилированного приложения тестировщики могут записывать журналы IntelliTrace™, благодаря которым при возникновении ошибки разработчик имеет полную картину происходящего в ходе выполнения теста. Таким образом, разработчики могут находить и устранять дефекты более эффективно, чем когда-либо до этого.

Рисунок 5 - Отчетность нагрузочного тестирования.

Разработчики также должны выполнять некоторые тесты до передачи кода тестировщикам или даже регистрировать его в системе управления версиями. С помощью модульных тестов разработчики проводят детальную проверку правильности поведения кода. Для групп, интенсивно использующих модульное тестирование (таких как группы, применяющие подход разработки через тестирование), предлагается новая возможность анализа влияния изменений в коде на тестирование. Этот анализ позволяет сократить количество выполняемых тестов с сохранением высокой степени уверенности выполнения обязательных тестов. В ходе анализа влияния изменений в коде на тестирование изменения в коде рассматриваются и соотносятся с прогонами модульных тестов, что позволяет определить, когда изменение в коде может повлиять на результаты ранее выполненного теста. Таким образом, разработчики проводят выборочную проверку своего кода перед его отправкой тестировщикам или группе контроля качества.

VS 2010 также предоставляет разработчикам возможность автоматизировать тестирование UI: кодированные тесты UI. Кодированные тесты автоматизируют тестирование элементов пользовательского интерфейса приложений. VS 2010 автоматически формирует код теста на основании прогона теста, либо осуществляемого тестировщиком, либо записанного самими разработчиками. Этот кодированный тест UI может выполняться вручную или как часть процесса сборки для автоматизации регрессивного тестирования UI. Инструмент управления лабораториями тестирования VS обеспечивает виртуализированную тестовую среду, которая может использоваться разработчиками для выполнения кода в чистых установках, что намного более точно соответствует поддерживаемым средам выпуска. Это позволит разработчикам экспериментировать и тестировать в безопасных легко повторяемых средах.

Автоматизированные сборки были сутью TFS, и новые возможности TFS 2010 как никогда упростили процесс выявления дефектов в коде до того, как они перекочуют в основной код приложения. TFS 2010 построен на базе Windows Workflow 4.0, что обеспечивает невероятные возможности по управлению рабочим процессом, ограниченные лишь вашим воображением. Одна из самых передовых из них – регистрация изменений при достижении порога качества. Это еще один шаг на пути улучшения непрерывной интеграции. Когда в ходе непрерывной интеграции сборка дает сбой, уже очень поздно: код в системе управления версиями уже поврежден, и группе требуется некоторое время на его исправление. Принцип регистрации изменений при достижении порога качества следующий: сборка и все необходимые тесты выполняются до регистрации кода в системе управления версиями. Таким образом, если происходит сбой сборки или теста, данный код просто не регистрируется в системе управления версиями. Разработчик, написавший этот код, получает возможность исправить его, а вся остальная группа разработки просто продолжает работу в обычном режиме. Реализовать это позволяет уникальная особенность TFS, так называемая возможность отложить регистрацию изменения. Если в ходе регистрации изменений возникает сбой, разработчик может попросить других разработчиков помочь в устранении дефекта в коде. Благодаря всем этим мероприятиям дефектный код гарантированно никогда не будет зарегистрирован в системе управления версиями.

и проверяется тестировщиками...

Обычно тестировщики не входили в состав группы разработки, и тестирование выполнялось после завершения разработки. Осознание того, что тестировщики должны быть неотъемлемой частью группы разработки, снизошло на организации лишь сейчас. Причиной этому послужили цифры, приведенные в начале данного раздела. Visual Studio 2010 обеспечивает полную интеграцию тестировщиков в любой проект, предоставляя возможность управления тестовыми случаями и профессиональный инструмент выполнения и управления тестированием Microsoft Test Manager (MTM), показанный на рис. 6.

Рисунок 6 - Microsoft Test Manager

В начале проекта группа контроля качества может напрямую связать тестовые случаи с требованиями. Благодаря этому разработчики видят, какие тестовые случаи будут выполняться. Позднее тестировщики тестируют код, проверяя его соответствие требованиям заказчика. Существует много типов тестирования для обеспечения качества приложения. Ручное или общее тестирование составляет около 70% всех мероприятий по тестированию, проводимых организациями в наши дни, и также доступны другие типы тестов. Диапазон этих других типов тестов очень широк, начиная от стандартных модульных тестов и тестов производительности, заканчивая нагрузочными и автоматизированными тестами. В ходе процесса тестирования тестировщики могут собирать детальные диагностические данные и записывать видео сеанса тестирования. Благодаря этому формируемые отчеты об ошибках становятся практически применимыми, т.е. разработчики могут на самом деле устранить выявленные дефекты. На рис. 7 показан раздел воспроизведения теста отчета об ошибке, сформированного MTM. Все шаги подробно описаны, включая сведения о том, какой шаг был пройден успешно, какой дал сбой, комментарии или вложения. Временная шкала видео проиндексирована, что позволяет разработчикам переходить именно в тот момент видеозаписи, который фиксирует возникновение ошибки. Также в отчет об ошибке включен журнал IntelliTrace™, в котором записаны подробные данные отладки, помогающие выявить дефект в ходе сеанса тестирования.

3/16/2010 4:16:14 PM Bug filed on "Logged on users information is pre-filled in the comment fields"

Step no.	Result	Title	Video links
1	Passed	**Navigate to http://tfs2010:8001**	Video:00:01:09
		Expected result: Blog engine welcome page is displayed	
2	Passed	**Click the Log In link**	Video:00:01:12
		Expected result: Log in page is displayed	
3	Passed	**Enter 'Jeff' for the username**	Video:00:01:14
4	Passed	**Enter 'P@ssw0rd' for the password and click Log In**	Video:00:01:16
		Expected result: Welcome page is displayed and you are logged on	
5	Failed	**Click the first blog post**	Video:00:01:05
		Expected result: Post details page is displayed and the name = 'Jeff' and the e-mail address = 'Jeff.Jones@nowhere.com'	
		Attachments: Screenshot1(TC143Iteration1Step5).png	

Test configuration: Windows 7 and IE 8

Diagnostic Data Adapter	Log / Output
Actions	TC143_ActionLog.1.txt
Actions	TC143_ActionLog.1.html
IntelliTrace	w3wp_100316_161610918_7180.1.iTrace
System Information	SystemInformation.1.xml
Video Recorder	TC143Video.1.wmv

Рисунок 7 - Этапы воспроизведения ошибки.

Анализ влияния изменений в коде на тестирование – новая возможность, благодаря которой тестировщики получают сведения, необходимые для выбора наиболее приоритетных тестов на основании изменений в коде, вносимых разработчиком. При внесении изменений в код, который уже прошел тестирование, тестировщики получают предупреждение и могут повторно выполнить эти тесты для устранения или сокращения регрессивных ошибок. Управление лабораториями тестирования обеспечивает виртуализацию инфраструктуры тестирования, что еще более совершенствует этот процесс. При возникновении ошибки делается мгновенный снимок виртуальной среды. Затем на основании этого снимка разработчик может без труда воссоздать ошибку в среде, в которой она была выявлена.

... и руководство имеет возможность его визуального контроля

Visual Studio 2010 и TFS 2010 обеспечивают качество, начиная с самих ранних этапов, а руководство желает знать о состоянии проекта. Руководство должно отчитываться перед заказчиками о ходе выполнения и качестве ПО, и если проблемы обнаруживаются в последний момент, уже очень сложно изменить курс. TFS включает систему детальных отчетов на базе служб аналитики SQL Server. В сочетании с сервером Microsoft Office SharePoint Server и службами Excel они обеспечивают постоянный доступ к актуальной и точной информации, не просто данным. На рис. 8 представлена одна из предлагаемых панель мониторинга.

Отчет о ходе выполнения плана тестирования показывает, сколько всего запланировано тестов, сколько тестов было выполнено, и сколько из них было завершено успешно. Отчет о состоянии сборки представляет число автоматизированных сборок и успешность их выполнения с течением времени: в нем данные об успешном/неудачном выполнении сборки сочетаются с результатами автоматизированного тестирования, которое выполняется как часть сборки. Это обеспечивает ежедневный снимок качества приложения. Отчеты о ходе исправления ошибок и реактивациях ошибок дают детальные сведения о состоянии всех выявленных в приложении ошибок, об объеме работ, которые пришлось выполнить из-за несоответствующего исправления ошибок.

Рисунок 8 - Показатели качества

Имея в распоряжении эти сведения, руководство получает возможность корректировать курс на ранних этапах. Если формируется большое число отчетов об ошибках, предпринимаются действия по сокращению их количества. Если тесты не выполняются, привлекается большее число тестировщиков. Таким образом, обеспечивать своевременную поставку качественного продукта, удовлетворяющего требованиям заказчика, становится проще.

ВОПРОСЫ ОБЕСПЕЧЕНИЯ КАЧЕСТВА

Цель всех проектов по разработке ПО одна – создание качественного приложения. Но эту цель сложно реализовать без соответствующих инструментов. От проектирования архитектуры приложения до разработки, тестирования и обслуживания – на всех этапах Visual Studio 2010 обеспечивает группы разработки необходимыми инструментами, помогая гарантировать наивысшее качество приложений. Эти инструменты охватывают все роли процесса разработки и созданы с учетом их нужд с целью упростить задачу реализации требований заказчиков.

С новым решением для разработки географически распределенный разработчик ПО ожидает удвоить производительность

IКомпания ICONICS, сертифицированный золотой партнер Майкрософт, поставляет промышленное ПО для автоматизации и визуализации в организации в более чем 60 странах. Группы разработки этой компании рассредоточены по трем странам. Такая географическая разбросанность обусловила трудности с координацией совместной деятельности. Разработчикам не хватало интегрированного набора инструментов, который упростил бы управление проектами, доступ к данным, обеспечил бы повышение темпов разработки и автоматизацию процессов тестирования. В 2009 году ICONICS модернизировали свою среду разработки, внедрив Microsoft Visual Studio 2010 Ultimate, Visual Studio Team Foundation Server 2010 и Visual Studio Lab Management 2010. Полностью интегрированное решение обеспечило географически распределенным группам унифицированные инструменты, что упростило процессы управления проектом и жизненным циклом продукта. В результате, ICONICS получили возможность сократить расходы и улучшить качество продукта и ожидают повысить производительность на 100%.

“Visual Studio 2010 предлагает поистине революционные новые возможности тестирования. Я вижу, что как только больше наших групп внедрят эти инструменты, мы удвоим нашу производительность. Теперь мы можем выполнять автоматизированное тестирование по ночам абсолютно без всякого участия человека. В будущем мы также ожидаем, что такие возможности, как IntelliTrace, обеспечат существенное сокращение времени отладки приложений.”

- Крис Элсбри, главный архитектор ПО, ICONICS

“Все преимущества применения Visual Studio 2010 сводятся к производительности и качеству. Такие возможности как прослеживание иерархии рабочих элементов, визуализация ветвления и IntelliTrace – и возможность создавать ПО для предыдущих версий с использованием новой системы – все это обеспечивает общее повышение нашей производительности. А это в итоге приводит к созданию более качественного продукта.”

- Расс Агруса, президент и исполнительный директор, ICONICS

Устранение дефектов

с Visual Studio 2010

На отладку кода уходит неимоверное количество времени, даже самые простые ошибки требуют внимания. Новые инструменты отладки в Visual Studio 2010 Ultimate обеспечивают успех любого сеанса отладки. Но, более того, теперь разработчики могут делать это намного быстрее, чем раньше. Неуловимые невоспроизводимые и неустойчивые ошибки могут устраняться быстро, что способствует экономии времени и денег и повышению удовлетворенности пользователей.

Get the free mobile app for your phone
http://gettag.mobi

Вопросы качества имеют первостепенное значение в деле производства ПО. Уже давно стало очевидным, что чем позже дефект выявляется, тем дороже его устранять. Например, для сообщества, использующего гибкие подходы к разработке, «ранний сбой» – это своего рода боевой клич и, следовательно, непрерывная интеграция и тестирование рассматриваются как фундаментальные практики. Visual Studio 2010 обеспечивает беспрецедентный набор нововведений в области тестирования и обеспечения качества [смотрите раздел «Создание качественных приложений»], благодаря которым группы разработки ПО получают возможность поставлять превосходные результаты.

Лейтмотивом выпущенной версии Visual Studio 2010 стала концепция «больше нет невоспроизводимых ошибок». То есть все дефекты, выявляемые тестировщиками, всегда могут быть воспроизведены людьми, занимающимися их устранением. Почему невозможность воспроизвести дефект является такой большой проблемой? Потому что устранение ошибок стоит времени и денег. Или, при самом худшем сценарии, дефекты просто остаются неисправленными, потому что разработчики не могут воссоздать их в управляемой среде. Большая часть этой проблемы обусловлена тем, как составляются отчеты о выявленных ошибках. Например, разработчика могут проинформировать об ошибке просто по телефону, сопровождая сообщение туманным рассказом о том, что делал пользователь в момент ее возникновения. Или ошибка может возникать на одном компьютере, но не повторяться на другом – печально известный синдром «на моем компьютере оно работает». С Visual Studio 2010Майкрософт сделали огромный шаг по направлению к полному устранению этой проблемы.

ПРАКТИЧЕСКИ ПРИМЕНИМЫЙ ОТЧЕТ ОБ ОШИБКЕ

Как решить эту проблему? Существует ряд подходов. Первый – обеспечить воспроизведение выявленных в системе дефектов в любых доступных средствах. Разработчик должен получать точные и практически применимые сведения. Смутных толкований причин возникновения ошибки недостаточно. С появлением Microsoft Test Manager в Microsoft Visual Studio Test Professional 2010 (также доступен в Visual Studio 2010 Ultimate) это видение стало реальностью. Будь то ручное тестирование с применением Microsoft Test Runner, либо автоматизированное или функциональное тестирование, тестировщики могут вести видеозапись хода тестирования. В случае если тест не пройден, эта видеозапись предоставляется разработчику, чтобы он мог увидеть, что именно привело к ошибке, и что пользователь делал в тот момент. Эти сведения имеют исключительную важность для устранения любого дефекта, их сбор обеспечивают сборщики данных.

Благодаря сборщикам данных инфраструктура тестирования может получать от тестируемой системы любые необходимые разработчикам данные. Существует несколько встроенных сборщиков данных, такие как средство записи видео и сборщик данных анализа влияния изменений в коде на тестирование, а также более комплексные инструменты, такие как сборщик сведений о системе и IntelliTrace™.

Встроенные инструменты не только могут предоставить разработчикам данные, к которым в настоящее время они не имеют доступа, сборщики также являются расширяемыми инструментами. Индивидуальные разработчики или группы разработки могут создавать собственные сборщики данных, которые будут собирать всю необходимую им информацию. Это обеспечивает неограниченные возможности при сборе сведений для понимания того, что случилось. Сборщики данных также могут использоваться при тестировании из Visual Studio 2010, например, в ходе продолжительного теста производительности или теста, запущенного из Team Build. Сборщики данных могут использоваться не только в диспетчере тестов.

Рисунок 1 – Журнал IntelliTrace™

Выявление дефекта

Как только разработчик понял, что происходит, он должен выполнить отладку проблемы и найти конкретное место в коде, куда необходимо внести исправления. Отладка кода может быть сложной. Еще сложнее выполнять отладку незнакомого унаследованного кода. Все это делает отладку также процессом, требующим много времени. Слишком часто разработчики не могут воспроизвести ошибки, о которых им сообщают, т.е. эти дефекты сохраняются до тех пор, пока не начнут проявляться достаточно часто, чтобы их можно было отследить. IntelliTrace™ является новым инструментом для разработчиков, с помощью которого они получают возможность «воспроизводить» сеансы отладки. Этим сеансом может быть предыдущий сеанс отладки или результат выполнения тестовых случаев. То есть разработчики могут возвращаться назад во времени и видеть, что происходило с приложением в любой момент сеанса. Разработчики даже смогут выяснять детали сбоя, не открывая исходный код.

Повышение производительности разработчиков неимоверное. Им больше не приходится выполнять приложение много раз, чтобы добиться повторения ошибки. Они просто воспроизводят сеанс, в котором возник сбой. Даже еще лучше, они могут запустить выполнение нового экземпляра приложения, подключить к выполняющемуся приложению журнал IntelliTrace™ и провести поэтапную отладку, сравнивая выполнение с записанным сеансом. На рис. 1 представлена страница сводных сведений журнала IntelliTrace™, которая дает представление о том, сколько данных собирается и передается на рассмотрение разработчику.

Журнал IntelliTrace™ показывает все выполнявшиеся потоки, что позволяет разработчику выбирать необходимый поток. Если отчет об ошибке был сформирован в ходе сеанса ручного тестирования с использование диспетчера тестов, разработчик может перейти прямо к разделу кода, который выполнялся, и проверить только его, а не проводить пошаговое выполнение всего приложения. IntelliTrace™ сокращает не только время выявления дефектов, но и негативные эмоции с этим связанные.

Устранение дефекта

Возможность IntelliTrace™в Visual Studio 2010 Ultimate обеспечивает быстрое получение развернутых сведений о дефектах, ускоряя и координируя действия разработчика-тестировщика, направленные на их устранение. У разработчика даже может сложиться ощущение того, что он работает в паре с экспертом-тестировщиком. На рис. 2 показан сеанс отладки с IntelliTrace™.

Возможность видеть все происходящее в ходе сеанса тестирования или отладки означает, что разработчикам не приходится выставлять по 50 точек останова и постоянно жать F5 (Выполнить), проверять данные в одной точке останова и переходить к следующей. Этот процесс особенно болезненный, если разработчик упускает точку в коде, где необходимо было бы выполнить остановку, поскольку в этом случае весь процесс приходится начинать заново!

Любопытный побочный эффект от получения в свое распоряжение всех этих сведений в том, что они позволяют находить дефекты, которые в противном случае остались бы невыявленными. На рис. 2 можно видеть исключения, которые были сформированы в ходе обычного выполнения приложения. Таким образом, разработчик не пытался выявить эту конкретную проблему, но она все равно была найдена! В ходе обычного сеанса отладки невозможно увидеть, что происходит до точки останова. Наличие такого представления означает, что появилась возможность выявлять ошибки, которые не были бы выявлены при использовании традиционных методов отладки.

Рисунок 2 - Сеанс отладки с IntelliTrace™

С помощью IntelliTrace™ разработчики могут видеть, какие методы или классы выполнялись в ходе теста, или просто находить, где возникла проблема, просматривая журналы IntelliTrace™. На рис. 3 разработчик пытается понять, почему в ходе выполнения пользовательского элемента управления загружаются неверные данные.

На рис. 3 видно, что просто просматривая журнал, можно определить, где загружается пользовательский элемент управления, и увидеть, что он выполняет чтение данных cookie-файла, чего не должно быть.

Рисунок 3 - Отладка с IntelliTrace™

Здесь разработчик может войти в метод и увидеть, что загружается и почему, а также получить все значения всех переменных в ходе сеанса отладки.

INTELLITRACE™ ЭКОНОМИТ ВРЕМЯ

На отладку кода уходит неимоверное количество времени, даже самые простые ошибки требуют внимания. Новые инструменты отладки в Visual Studio 2010 Ultimate обеспечивают успех любого сеанса отладки. Но, более того, теперь разработчики могут делать это намного быстрее, чем раньше. Неуловимые невоспроизводимые и неустойчивые ошибки могут устраняться быстро, экономя время тестировщиков и разработчиков. Имея IntelliTrace™ в составе используемого комплекта инструментов, руководители групп могут быть уверенны, что специалисты быстро возвратятся к работе над следующей возможностью или выпускаемой версией.

Minitab®

Интегрированный комплект инструментов сокращает затраты и упрощает процесс разработки ПО

Компании Minitab, занимающейся разработкой ПО для статистики, требовалось упростить и ускорить процесс разработки. Они внедрили интегрированное решение, включающее Microsoft Visual Studio 2010 Ultimate, Visual Studio Team Foundation Server 2010 и Visual Studio Team Lab Management 2010. Теперь Minitab, используя такие возможности, как IntelliTrace и Microsoft Visual F#, могут повысить продуктивность, снизить затраты на разработку и лицензирование и сократить время выхода новых предложений на рынок.

“IntelliTrace будет очень важным инструментом отладки для нас, потому с его помощью мы можем записывать рабочий процесс тестирования и затем прикреплять эту запись к отчету об ошибке. Кроме того, IntelliSense в Visual Studio 2010 предлагает новые возможности формирования отчетов об ошибках, синтаксического анализа и обработки в фоновом режиме, а это, несомненно, ускорит разработку. Мы также можем использовать Visual Studio Team Foundation Server 2010 для управления регистрацией изменений в коде в системе управления версиями и публикации тестовых рабочих потоков в файлах журналов IntelliTrace.”

- Мэтт Ковальски, руководитель отдела разработки баз данных, Minitab

Wintellect®
Know how.

За счет применения нового инструмента консалтинговая фирма ожидает сократить время отладки, по крайней мере, на 25%

Консалтинговой фирме по разработке ПО Wintellect требовались более инновационные процессы отладки, что могло бы способствовать сокращению времени на устранение дефектов и повышению качества кода. Компания нашла то, в чем нуждалась, в системе разработки Microsoft Visual Studio 2010 Ultimate, которая включает новый инструмент отладки IntelliTrace. Теперь у Wintellect есть необходимые инструменты для улучшения координации совместной деятельности, снижения затрат и отладки приложений на 25% быстрее.

“Насколько мне известно, аналогов инструменту IntelliTrace в Visual Studio 2010 Ultimate не существует.”

- Джон Роббинс, соучредитель, Wintellect

Повышение производительности и эффективности тестирования

с Visual Studio Test Professional 2010

Любой специалист по контролю качества и тестированию имеет две основные цели: проверить качество приложения и улучшить общее качество ПО. До сих пор время, отведенное на тестирование, часто непродуктивно расходуется на вынужденные повторения, теряется из-за нечеткой приоритизации тестов и необходимости устранения ошибок, обусловленных человеческим фактором.

Visual Studio Test Professional 2010 обеспечивает продуктивную и эффективную реализацию этих двух целей с применением соответствующих инструментов тестирования и диагностики, что позволяет создавать превосходные решения и взаимодействия. Также он исключает потери на протяжении всего жизненного цикла приложения, устраняя изолированность групп и обеспечивая основные возможности интеграции, гарантирующие повышение производительности групп.

Get the free mobile app for your phone
http://gettag.mobi

Любой специалист по контролю качества и тестированию имеет две основные цели: проверить качество приложения и улучшить его качество. Существует множество проблем, которые очень усложняют достижение этих двух простых целей. Время, отведенное на тестирование, часто тратится впустую на вынужденные повторения, теряется из-за нечеткой приоритизации тестов и необходимости устранения ошибок, обусловленных человеческим фактором. Эти распространенные проблемы делают тестирование неэффективным и непродуктивным. Но что такое хороший отчет об ошибке? Почему один отчет является практически применимым, а другой нет? Почему сведений, которых вполне достаточно для решения одной проблемы, совершенно недостаточно для другой? Отчеты об ошибках, не включающие достаточно информации, приводят к ухудшению отношений между разработчиками и тестировщиками, обусловливая напряженность в общении и сотрудничестве. И это всего лишь один аспект процесса тестирования. Тестировщики часто сталкиваются со сложностями при составлении отчетов о качестве приложения из-за отсутствия его целостного представления. Studio Test Professional 2010 призван устранить пробелы, имеющиеся во многих средах разработки ПО, и обеспечить создание ПО наивысшего качества.

СЕГОДНЯ

Одной из основных причин возникновения напряженности между разработчиками и тестировщиками является невозможность устойчивого воспроизведения многих ошибок. Если ошибка не может быть воспроизведена, ее очень сложно устранить. Это приводит к вражде между разработчиками и тестировщиками и пустой трате времени на попытки тестировщиков воспроизвести ошибку и попытки разработчиков найти невоспроизводимую ошибку. Если тестировщики не в состоянии предоставить практически применимые отчеты об ошибках, разработчики не ценят их вклад в общий процесс. В преимущественном большинстве ситуаций Microsoft Test Manager 2010, основной инструмент тестирования в составе Visual Studio Test Professional 2010 (также включен в пакет Visual Studio 2010 Ultimate), гарантирует формирование практически применимых отчетов об ошибках и возможность их отслеживания до исправления ошибки.

Обычной причиной невозможности воспроизведения ошибки является качество отчета об ошибке. На создание должного отчета об ошибке требуется время даже при наличии достаточного объема сведений. Во многих случаях тестировщики не располагают всеми необходимыми данными или не могут вспомнить все выполненные ими действия. Это особенно чувствуется при проведении произвольного тестирования, когда нет четко обозначенной последовательности шагов. Время является критическим компонентом процесса тестирования.

Что такое практически применимый отчет об ошибке?

Практически применимый отчет об ошибке включает достаточно данных, чтобы разработчик мог воспроизвести ошибку и определить, где произошел сбой. Кроме того, в отчете должно быть обозначено желаемое поведение (т.е. что код должен делать). Благодаря этому разработчик сможет понять, что ошибка устранена.

Если ошибка обусловлена определенными настройками среды, воспроизвести ее становится намного сложнее. Если к проблеме приводит какой-то один из параметров среды, шансы на ее воспроизведение без доступа к этой конкретной среде ничтожны. Причин этому множество, но самой очевидной является то, что разработчики ничего не знают о характеристиках компьютера, на котором выполняется тестирование, или не имеют доступа к этому компьютеру.

Один из руководителей группы тестирования заметил относительно тестировщиков: «В то время как от них требуется большая продуктивность, они тратят уйму времени на многократное тестирование одних и тех же возможностей, поскольку вынуждены возвращаться и проверять устраненные ошибки». Имея в своем распоряжении ограниченное количество времени, разработчики должны работать максимально эффективно и выполнять только те тесты, которые обеспечат выявление ошибок с наибольшей вероятностью. Когда разработчики вносят небольшие изменения в код, тестировщикам необходимо иметь средство для отсева тестовых случаев, на которые эти изменения, скорее всего, не оказали никакого влияния. Имея такое огромное число тестовых случаев и такие ограниченные сроки, тестировщики могут просто не добраться до наиболее важных тестовых случаев.

ЗАВТРА

Microsoft Visual Studio Test Professional 2010 обеспечивает инструменты для решения абсолютно всех упомянутых здесь проблем. Это мощный, но при этом простой в использовании инструмент, с помощью которого группы получают возможность управлять планами тестирования, выполнять тесты, формировать практически применимые отчеты об ошибках и проверять, были ли исправлены выявленные дефекты. Также с его помощью можно формировать отчеты по многим разным аспектам тестирования. Поскольку эти возможности тесно интегрированы с Team Foundation Server 2010 (TFS), они используют возможности TFS для обеспечения полной прослеживаемости от требований к тестовым случаям и от дефектов к коду. Кроме того, для составления отчетов им обеспечиваются расширенные возможности хранилища данных и аналитического куба. Но как это обеспечит повышение эффективности работы тестировщиков?

Ядром Microsoft Visual Studio Test Professional 2010 является Microsoft Test Manager (диспетчер тестов). С его помощью группы тестирования могут управлять планами тестирования, пакетами тестов и тестовыми случаями вплоть до получения результатов отдельных прогонов тестов. Он не требует установки Visual Studio, а подключается к Team Foundation Server 2010, который интегрирует тестирование с процессом разработки и обеспечивает беспрепятственное взаимодействие между разработчиками и тестировщиками. Это инструмент, посредством которого тестировщики выполняют тестовые случаи, формируют отчеты об ошибках и проверяют состояние исправленных дефектов. С его помощью тестировщики работают более эффективно и формируют практически применимые отчеты об ошибках.

Кроме того, современный и легковесный интерфейс диспетчера тестов посредством сквозного рабочего процесса планирование-тестирование-отслеживание обеспечивает возможность сосредоточиться на задачах, которые приходится выполнять чаще всего. На рис. 1 представлен диспетчер тестов в режиме **планирования**.

Рисунок 1 – Интерфейс Microsoft Test Manager 2010, реализующий рабочий процесс планирование-

Что делает практически применимый отчет об ошибке? Для разработчика очень важно знать, какие действия тестировщика привели к возникновению ошибки, какие данные были введены в приложение, условия среды, в которой выполнялся тест, и тестирование какой сборки ПО выполнялось. Введение всех этих сведений вручную требует много времени и чревато ошибками, именно поэтому традиционные методы формирования отчетов не годятся. Диспетчер тестов берет всю эту работу на себя. На рис. 2 представлены этапы воспроизведения ошибки, отчет о которой был сформирован диспетчером тестов.

Step no.	Result	Title	Video links
1	Passed	Log on as Jeff (Editor)	
1.1	None	Navigate to http://vs2010beta2:8001 Expected result: Blog Engine welcome page is displayed	
1.2	None	Click the Log In link Expected result: Log in page is displayed	
1.3	None	Enter the username as 'Jeff'	
1.4	None	Enter the password as 'P@ssw0rd' and click Log In Expected result: Welcome page is displayed and you are logged in as an editor. The administration box is displayed and it does not have an entry for 'Users'	
2	Failed	Click the Welcome to Blog Engine .NET 1.5 blog post Expected result: The post details are displayed and the Name = 'Jeff' and the e-mail address = 'Jeff.Levinson@nwcadence.com' Comments: Previous users information was displayed Attachments: TC41Iteration1Step2Screenshot1.png	Video:00:00:31

Test configuration:	Vista and IE 7
Applications being recorded:	All Applications except: winword.exe , outlook.exe , excel.exe , infopath.exe , onenote.exe , msaccess.exe , powerpnt.exe , groove.exe , communicator.exe , leviewer.exe , mstsc.exe
Data Collector	**Log / Output**
Action Recording and Action Log	TC41_ActionLog.1.txt
Action Recording and Action Log	TC41_ActionLog.1.html
IntelliTrace	w3wp_091227_155826018_17232.1.tdlog
System Information	SystemInformation.1.xml
Video Recorder	TC41Video.1.wmv

Рисунки 2 и 3 - Этапы воспроизведения ошибки, отчет о которой был сформирован с помощью Microsoft Test Manager 2010

В данном случае разработчик точно знает, какой тестовый случай выполнялся (к отчету об ошибке подключен тестовый случай), какие этапы были пройдены, и какой(-ие) этап(-ы) дал(-и) сбой. Как часть тестового случая велась видеозапись (средство записи видео – это всего лишь один из сборщиков данных, предоставляемых тестировщикам). Временная шкала этой видеозаписи проиндексирована, благодаря чему разработчик может точно видеть, что и когда пошло не так. Тестировщик также прикрепляет снимок среды в момент сбоя, ассоциированный с конкретным этапом теста.

User Name	tfsSetup
Computer Name	VS2010BETA2
User Domain Name	VS2010BETA2
OS Name	Microsoft® Windows Server® 2008 Standard
OS Version	Microsoft Windows NT 6.0.6002 Service Pack 2
System Directory	C:\Windows\system32
System Locale	English (United States)
User Locale	English (United States)
Total Physical Memory	4094 MB
Available Physical Memory	1129 MB
Memory Load	72%
Total Virtual Memory	2047 MB
Available Virtual Memory	1340 MB
Processor Name	Intel(R) Core(TM)2 Duo CPU T9400 @ 2.53GHz
Processor Family	x64 Family 6 Model 23 Stepping 6
Processor Speed	2527 MHz
Screen Resolution	1024 * 768
Color Quality	16
Internet Explorer Version	8.0.6001.18865

Рисунок 4 - Сведения о системе автоматически фиксируются в отчете об ошибке.

Когда тестировщики формируют отчеты об ошибках, разработчики получают доступ к дополнительным сведениям, любезно собранным в фоновом режиме сборщиками данных (рис. 4).

Если этой информации не достаточно, можно создавать дополнительные сборщики данных для сбора именно тех сведений, которые представляют интерес для конкретной организации или процесса тестирования. Сборщики данных обеспечивают расширенные возможности диагностики, поэтому позволяют собирать на целевом компьютере любые необходимые сведения посредством простого механизма расширяемости.

Диспетчер тестов предлагает новый инструмент, IntelliTrace™, для записи всего сеанса отладки тестируемого приложения. Разработчик получает возможность, буквально, воспроизводить сеанс тестирования и просматривать выполнение кода по шагам, не выполняя приложение. Таким образом, разработчик получает не только конечный результат, каковым является отчет об ошибке: журнал IntelliTrace™ позволяет ему видеть все значения всех переменных в ходе выполнения теста и все сформированные исключения.

Функциональность управления лабораториями тестирования обеспечивает дальнейшее улучшение этого процесса. Для управления лабораториями тестирования используется технология виртуализации, которая позволяет ускорить установку, свертывание и восстановление сложных виртуальных сред в исходное состояние. Проблема невозможности воспроизведения ошибки решается через обеспечение тестировщикам средств для формирования развернутых отчетов об ошибках со ссылками на контрольные точки, которые используются разработчиками для воссоздания ошибок в сложных средах. Одним щелчком мыши разработчик может запустить виртуальную среду, точно соответствующую той, в которой была выявлена ошибка. Наконец, существенно расширяются возможности автоматизации сборки за счет автоматизации подготовки виртуальной машины, интегрированного развертывания сборки и проверки сборки. С таким подходом группы получают большую гибкость к изменениям в этом предъявляющем все более и более высокие требования мире.

Как говорилось ранее, тестировщики тратят чрезмерно много времени на тестирование одного и того же кода снова и снова, потому что должны проверять исправление ошибок или выполнять регрессивное тестирование. Задача диспетчера тестов – сократить время, затрачиваемое на эти операции. Для улучшения производительности и эффективности предлагается две основных возможности: запись автоматизированных тестов для ускоренного прогона ручных тестов и анализ влияния изменений в коде на тестирование. Это позволяет сократить время на проверку того, была ли устранена ошибка, при этом сохраняется повторяемость и обеспечивается возможность проверять намного больше ошибок в заданный период времени.

Рисунок 5 – Итерации теста с привязкой к данным записываются для обеспечения возможности ускоренного прогона ручных тестов.

Возможность пропуска части тестов для ускоренного прогона ручных тестов обеспечивается путем записи журнала действий в ходе первого прогона ручных тестов и позволяет перемещаться по тестовому случаю для продолжения проверки при последующих прогонах. При воспроизведении теста участие пользователя не требуется. Запись автоматизированных тестов будет воспроизводиться со скоростью, соответствующей скорости ответов приложения.

При первом прогоне теста диспетчер тестов записывает управляемые данными итерации с помощью Microsoft Test Runner. Записанные связанные с данными действия используются впоследствии для ускоренного прогона ручных тестов – это мощная возможность для быстрого перемещения по тесту.

А как же ситуации, когда разработчики могут непреднамеренно создать условия для возникновения регрессивных ошибок? Как тестировщики смогут знать, что необходимо тестировать? Именно здесь на помощь приходит интеграция инструментов тестирования и разработки с возможностью анализа влияния изменений в коде на тестирование (Test Impact Analysis, TIA). Вместо того чтобы проводить часы или даже дни, пытаясь определить, какие тесты должны быть выполнены повторно, тестировщик просто берет список рекомендованных в связи с изменениями в коде тестов и сразу же знает, какие дефекты были устранены в данной сборке. Теперь вы можете правильно выбирать момент получения новой сборки для тестирования, значительно сокращая потери времени и усилий.

Анализ влияния изменений в коде на тестирование (TIA) соотносит код, выполняемый для каждого теста со всем тестом. При внесении изменения в исходный код TIA находит тесты (или тестовые случаи), которые ранее выполнялись для этого кода. На основании этого тестировщики проводят анализ рисков для тестовых случаев и определяют их приоритеты для использования в регрессивном тестировании. Эта тактика применяется для ручного, автоматизированного и модульного тестирования.

Составлять отчеты о качестве приложений сложно. По большей части из-за того, что код, результаты тестирования и требования обычно не интегрированы в одну систему. Visual Studio Test Professional 2010 максимально использует преимущества этой интеграции и инструментарий бизнес-аналитики от Майкрософт для предоставления комплексных панелей мониторинга по качеству приложения в целом. И все эти возможности являются встроенными. Мощь SharePoint позволяет публиковать эти отчеты для заинтересованных сторон, кому необходимо получать эти сведения на регулярной основе. Предлагаются стандартные отчеты, такие как соотношение числа непройденных и пройденных тестов и количество сформированных отчетов об ошибках. Всего для описания качества приложения имеется 15 стандартных отчетов. Один из самых интересных – обзор описаний функциональности (рис. 6), который предлагает многогранный подход к консолидации данных.

Рисунок 6 - Пример отчетов о тестировании

В этом отчете состояние процесса тестирования сопоставляется с отдельными требованиями системы, количеством отчетов об ошибках, которые были сформированы для этих требований, их состоянием и ходом выполнения работ по реализации требований. Для каждого отдельно взятого требования пользователи сразу могут видеть число созданных тестовых случаев, сколько из них было выполнено, сколько пройденных и непройденных, число активных, устраненных и закрытых дефектов. В сочетании с отчетами по сборкам и обновлениям кода получается общая картина качества приложения. Благодаря этому тестировщикам проще увидеть, куда должны быть направлены их усилия по тестированию для обеспечения максимального качества приложения.

ЗАКЛЮЧЕНИЕ

С Microsoft Visual Studio Test Professional 2010 тестировщики получают возможность делать больше, прилагая для этого меньше усилий, и инструменты для создания практически применимых отчетов об ошибках. Тестировщики могут сосредоточиться на том, что должно быть протестировано. Сокращается количество невоспроизводимых ошибок, улучшается взаимодействие разработчиков и тестировщиков, что ведет к повышению продуктивности. Значительно сокращается количество мероприятий, на которые обычно уходит много времени в ходе процесса тестирования, такие как написание отчетов об ошибках, проверка состояния исправленных дефектов и определение того, какие тесты должны выполняться в ходе регрессивного тестирования. В результате этого тестировщики получают больше времени на сложные задачи, такие как произвольное тестирование, или на создание большего числа тестовых случаев для обеспечения большего покрытия кода тестами. Microsoft Visual Studio Test Professional 2010 позволяет тестировщикам делать больший вклад в создание качественных приложений.

Интегрированное решение по управлению виртуальными лабораториями тестирования с Visual Studio 2010

Разработка современного ПО требует применения современных практик разработки, которые устраняют пропасть между руководством ИТ-отдела, группой разработки и группой контроля качества. Visual Studio 2010 предлагает инструменты и технологию, необходимые для простой и эффективной разработки и тестирования ПО в средах, которые повторяют среды производственной эксплуатации. Такая имитация стала возможной благодаря функциональности управления лабораториями тестирования. Автоматизированные инструменты позволяют создавать и управлять этими средами без особых усилий. Встроенные рабочие процессы, поддерживающие регрессивное тестирование и интерактивное использование, обеспечивают быстрое автоматизированное развертывание.

ИНТЕГРИРОВАННОЕ РЕШЕНИЕ ПО УПРАВЛЕНИЮ ВИРТУАЛЬНЫМИ ЛАБОРАТОРИЯМИ ТЕСТИРОВАНИЯ

Создание высококачественного ПО предъявляет высокие требования к организации и группам разработки. Номенклатура создаваемого сегодня ПО широка: от простых клиентских приложений для сложных развертываний с множеством уровней, поддерживающих разные операционные системы, и вспомогательных служб, таких как базы данных и Веб-серверы. Создавать эти решения достаточно сложно. А если взять во внимание еще и контроль качества и регрессивное тестирование, задача может показаться вообще неосуществимой.

СЕГОДНЯ

Сегодня создание высококачественного ПО подразумевает наличие эффективного цикла разработки ПО, который включает некоторый формальный процесс контроля качества. Типовой процесс контроля качества охватывает специалистов, процесс и инструменты. Лаборатория тестирования – замечательный инструмент, который нелишним будет иметь. Современные лаборатории тестирования обычно бывают двух типов.

Первый тип – это старые лаборатории тестирования с реальным оборудованием, в качестве которого обычно применяются компьютеры, ранее использовавшиеся для разработки. Созданием виртуальных образов и развертыванием занимается группа инженеров, что может привести к недостаточной гибкости цикла контроля качества в целом, от создания новой сборки до тестирования. Кроме того, работой всех этих специалистов необходимо управлять, и их труд должен оплачиваться. В эпоху, когда технология движется в направлении к «зеленому ИТ», ничто не является таким ярким примером старого мира, как большая лаборатория неэффективных компьютеров, создающих собственную климатическую зону.

Второй тип лаборатории тестирования – современная лаборатория, использующая технологию виртуализации. Несмотря на то, что она знаменует собой гигантский шаг вперед по сравнению с типовой лабораторией, проблемы есть и здесь. ПО для управления лабораториями тестирования в наши дни очень дорогостоящее. Также многие продукты слишком сложны и не укладываются в общий цикл разработки ПО (Software Development Lifecycle, SDLC). Многие из них являются побочным продуктом поставщиков, основным родом деятельности которых является эксплуатация, а не разработка или контроль качества. Интегрировать практику разработки и контроля качества с их рабочим процессом может быть сложно и утомительно.

Независимо от используемого типа лаборатории тестирования, существует болезненная проблема, обусловливаемая типичным циклом устранения дефектов между разработчиками и тестировщиками. В старой лаборатории тестирования тестировщики имеют доступ к оборудованию и ресурсам только для выполнения нескольких строго оговоренных циклов. Это существенно усложняет или делает практически невозможным доступ разработчиков к лаборатории тестирования, который может потребоваться, когда дефект легко выявляется в среде лаборатории тестирования, но не воспроизводится в среде разработки. Обычно среды тестирования и производственной эксплуатации отличаются от среды разработки, и даже наличие высококачественных отчетов об ошибках не помогает в этом случае. Это приводит к «пинг-понгу» проблемами между группами разработки и тестирования.

НЕПРЕРЫВНАЯ ПОСТАВКА ЧЕРЕЗ ИНТЕГРИРОВАННОЕ УПРАВЛЕНИЕ ЛАБОРАТОРИЯМИ ТЕСТИРОВАНИЯ

Решение Microsoft Visual Studio по управлению лабораториями тестирования обеспечивает полный набор инструментов для управления и использования виртуальной лаборатории тестирования. Оно предлагается в комплекте с Visual Studio 2010 Ultimate с подпиской MSDN или Microsoft Visual Studio Test Professional 2010 с подпиской MSDN и помогает устранить проблемы большинства решений лабораторий тестирования, чтобы сократить трения между разработчиками и тестировщиками. Построенное поверх Windows Hyper-V и System Center Virtual Machine Manager 2008 R2 и интегрированное с Team Foundation Server 2010 (TFS) решение по управлению лабораториями тестирования обеспечивает эффективное с экономической точки зрения комплексное решение для тестировщиков и разработчиков. Оно позволяет быстрее выявлять и устранять дефекты и в конечном счете помогает организации быстрее вывести качественное решение на рынок.

УМЕНЬШЕНИЕ ОПЕРАЦИОННЫХ ИЗДЕРЖЕК ЧЕРЕЗ МОДЕЛЬ САМООБСЛУЖИВАНИЯ

Функциональность управления лабораториями тестирования доступна уже сегодня как часть подписки MSDN с Visual Studio 2010 Ultimate или Visual Studio Test Professional 2010. Эта модель лицензирования «на пользователя» упрощает внедрение решения по управлению лабораториями тестирования на индивидуальной основе, несмотря на то что эта функциональность в значительной степени поддерживается Team Foundation Server 2010. Microsoft Test Manager 2010[1] и System Center Virtual Machine Manager 2008 R2[2] будут вашими инструментами конфигурации и управления, обеспечивающими эту встроенную технологию. Сразу же после установки и конфигурации среды домена Team Foundation Server 2010 группа может пользоваться преимуществами модели самообслуживания. Это обеспечивает каждому участнику группы возможность доступа к виртуальным средам, повторяющим среду производственной эксплуатации, по требованию!

Поскольку решение управления лабораториями тестирования построено поверх Hyper-V, оно обеспечивает все преимущества виртуализации. Можно использовать сервер Hyper-V или активировать роль Hyper-V на Windows Server 2008 или Windows Server 2008 R2. Также доступны преимущества «зеленого ИТ»: сокращение числа физических серверов, которые потребляют мощность и генерируют тепло, снижение затрат на распределение оборудования и сокращение потребностей в рабочей силе для управления всеми этими серверами. Hyper-V поддерживает ряд расширенных возможностей, включая кластеризацию и динамическую миграцию, таким образом, лаборатории тестирования тоже имеют доступ к ним всем.

Основная точка ввода в систему управления лабораториями тестирования – диспетчер тестов. Это насыщенное приложение Windows Presentation Foundation (WPF), которое упрощает пользователю доступ и использование виртуальных машин. Центр лабораторий диспетчера тестов обеспечивает группе простой доступ ко всем возможностям лаборатории тестирования для быстрого создания и управления средами виртуальных машин (рис. 1). В центре лабораторий среды создаются, развертываются и клонируются. У каждой среды могут быть разные параметры тестирования, и она может быть образована существующими виртуальными машинами, шаблонами – которые также называют «золотыми образами» – или их сочетанием. Кроме того, можно создать физическую среду из физических компьютеров. Решение управления лабораториями тестирования обеспечивает инструменты управления для совместного использования и перемещения ресурсов туда, где они необходимы.

Рисунок 1 – Центр лабораторий диспетчера тестов обеспечивает доступ к виртуальной лаборатории тестирования.

Диспетчер тестов также обеспечивает доступ к Microsoft Environment Viewer (Обозреватель сред). Этот инструмент позволяет подключаться ко всем виртуальным машинам в среде как к единому модулю. Кроме того, можно создавать мгновенные снимки среды (рис. 2).

1 Поставляется с Microsoft Visual Studio 2010 Ultimate или Microsoft Visual Studio Test Professional 2010.

2 Поставляется с Microsoft Visual Studio 2010 Ultimate с подпиской MSDN или Microsoft Visual Studio Test Professional 2010 с подпиской MSDN.

Рисунок 2 – Microsoft Environment Viewer обеспечивает быстрый доступ к виртуальным машинам и мгновенным снимкам их сред.

Диспетчер тестов делает виртуальные машины доступными. С полной поддержкой ролевого администрирования пользователи могут без труда выполнять доступ к существующим средам, как для разработки, так и для тестирования. Кроме того, диспетчер тестов обеспечивает упрощенный доступ к виртуальным машинам и «золотым образам». Благодаря этому участники группы могут самостоятельно развертывать новые среды, не создавая дополнительных проблем для ИТ-отдела и не ожидая, пока один из сотрудников ИТ-отдела выполнит необходимые настройки.

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ РАБОТЫ ЧЕРЕЗ АВТОМАТИЗИРОВАННЫЕ РАБОЧИЕ ПРОЦЕССЫ

Созданные среды можно без труда сделать основой рабочего процесса сборка-развертывание-тестирование (build-deploy-test, BDT). Используя возможность сборки в Team Foundation Server, просто выбираем сборку из обозревателя проекта и выполняем мастер для описания всего рабочего процесса. Сюда может входить задание среды, используемой в качестве установочной базы, управление и применение мгновенных снимков среды (рис. 3), определение того, какая сборка используется, как она должна развертываться, и принятие решения о том, как должны выполняться автоматизированные тесты.

Мгновенные снимки среды устраняют барьеры в тестировании сред, включающих множество компьютеров. Всего один щелчок мыши – и система управления лабораториями тестирования сохраняет мгновенный снимок всех виртуальных машин среды в соответствующий момент времени. Этот мгновенный снимок может использоваться для перемещения вперед и назад по временной шкале, благодаря чему разработчики и тестировщики получают возможность всегда знать состояние виртуальных машин, с которыми работают.

Рисунок 3 – Задайте, какой мгновенный снимок должен использоваться, перед развертыванием приложения.

Рабочий процесс сборки отличается большой гибкостью и поддерживает расширяемость через действия Windows Workflow 4. Он обеспечивает полный контроль над использованием сред. В частности, он упрощает развертывание в известную конфигурацию (рис. 4). Посредством мгновенных снимков можно задать определенное состояние среды до развертывания. Затем, после развертывания приложения, рабочий процесс может использовать другой мгновенный снимок среды. Это обеспечивает громадную экономию времени для тестировщиков. Они могут без труда настраивать определенную среду, выполнять несколько тестов, возвращать среду в исходное состояние и быть уверенным, что приложение установлено и очищено для следующего цикла тестирования.

Рисунок 4 – Сделайте мгновенный снимок среды после развертывания сборки.

Как говорилось выше, в ходе процесса сборки в среде могут выполняться автоматизированные тесты. Агенты тестирования[1], которые выполняют эти тесты и рабочий процесс, поддерживают сбор развернутых диагностических данных. Среда может использовать общий или собственный набор параметров, определяющий собираемые данные (рис. 5). Эти адаптеры диагностических данных поддерживают ряд опций для настройки стратегии сбора данных для каждой сборки и среды.

1 Поставляется с Visual Studio Agents 2010 – дополнительное ПО для Visual Studio 2010, свободно доступно в центре загрузки Майкрософт.

Рисунок 5 – Задание активных адаптеров для среды.

ИНТЕРАКТИВНОЕ ИСПОЛЬЗОВАНИЕ ЧЕРЕЗ ПОВТОРНОЕ И СОВМЕСТНОЕ ИСПОЛЬЗОВАНИЕ

Использование технологии виртуализации в системе управления лабораториями тестирования ускоряет настройку, свертывание и восстановление сложных виртуальных сред в исходное состояние с обеспечением непрерывной их доставки группам. Если среда оказывается полезной или востребованной, ее можно клонировать и предоставить для доступа другим участникам группы. Это возможно даже для сложных сред, включающих множество виртуальных машин, в том числе контроллеры доменов, серверы базы данных и многое другое. В управлении лабораториями тестирования это обеспечивается возможностью, которую называют изоляция сети (рис. 6). Она позволяет многим участникам группы развернуть одну и ту же среду и использовать ее без создания конфликтов имен компьютеров или DNS в общей сети группы.

Адаптеры диагностических данных (DDA) – это подключаемые модули, используемые агентами тестирования для сбора данных в ходе выполнения тестов, ручных или автоматизированных. Разные комбинации DDA можно объединять в логически группы параметров тестирования. Кроме предоставляемых Майкрософт DDA, можно создавать собственные адаптеры для сбора специальных данных.

Рисунок 6 – Включение изоляции сети для улучшения возможностей повторного и совместного использования среды.

Замечательно, что диагностические возможности, доступные в ходе выполнения сборок и автоматизированных тестов, также доступны тестировщикам при выполнении ручного тестирования. Автоматически собранные данные могут быть добавлены в любое поле отчета об ошибке, что делает этот отчет практически применимым для разработчика. Использование обозревателя сред решения управления лабораториями тестирования обеспечивает мощную возможность включения ссылки на мгновенный снимок среды в отчет об ошибке (рис. 7). Имея в своем распоряжении эти данные, разработчик возвращает тестовую среду в момент времени и условия, при которых была выявлена ошибка. Так решается проблема невозможности воспроизведения: тестировщики имеют возможность предоставлять отчеты об ошибках с ссылками на мгновенные снимки среды, которые используются разработчиками для воссоздания условий возникновения сбоя. Одним щелчком мыши разработчик может запустить виртуальную среду, которая точно соответствует той, в которой была выявлена ошибка.

Рисунок 7 – Включение ссылки на мгновенный снимок среды в новый

ЗАКЛЮЧЕНИЕ

Решение Microsoft Visual Studio 2010 по управлению лабораториями тестирования обеспечивает инструменты, необходимые для упрощенного создания, использования и управления множеством сред виртуальных машин Hyper-V в целях разработки и контроля качества. Благодаря Microsoft Test Manager каждый участник группы имеет в своем распоряжении простой в использовании инструмент для самостоятельного доступа к необходимым средам. Эти реализованные в стиле мастера инструменты упрощают и ускоряют подготовку и клонирование сред. Всесторонняя интеграция с Team Foundation Server и сервером сборки означает, что используемые среды могут синхронизироваться с последними сборками. Кроме того, интеграция рабочего процесса со сборкой обеспечивает развертывание последних версий ПО с полной поддержкой автоматизированного регрессивного тестирования и проверки. Наконец, благодаря общему доступу разработчики могут загружать именно те среды, которые использовались тестировщиками для выявления дефекта, что позволяет быстро выявить и устранить его. С решением Visual Studio по управлению лабораториями тестирования группа может поднять планку качества для всех создаваемых ею приложений.

Приводимые в данном документе сведения, включая URL и другие ссылки на Веб-сайты, могут меняться без уведомления. За исключением случаев, явно оговоренных любым письменным лицензионным соглашением с Майкрософт, предъявление данного документа не обеспечивает права на патенты, торговые марки, авторские права или другую интеллектуальную собственность, являющуюся предметом данного документа.

© Copyright 2011 Корпорация Майкрософт. Все права защищены.