создать дополнительные таблицы и хранимые процедуры, а также создать несколько новых ssis пакетов. Таким образом, внесением изменений только на уровне данных, обеспечивается возможность управления логистическими единицами. В докладе рассматривается организация программного средства для загрузки медицинских логистических данных, позволяющая снизить накладные расходы по ведению документооборота, устранить поддержку ненужных документов, автоматизировать импорт новых логистических единиц. Рассматриваемое программное средство за счет использования протокола SFTP обеспчивает безопасную транспортировку данных, за счет наличия усилинной технологии валидации обеспечивает контроль дубликатов или некорректных данных.

Список использованных источников:

1. «Ассоциация автоматической идентификации «ЮНИСКАН / ГС1 РУС». Логистика для медицинской промышленности. [Электронный ресурс] — Режим доступа. — URL: http://www.gs1ru.org/files/2721/StandartyGS1variant.pdf.
2. «Мировой рынок систем электронного документооборота» [Электронный ресурс] — Режим доступа. — URL: http://citforum.ru/consulting/docflow/market/article1.8.200222.html(Дата обращения 16.11.2012).

МОДЕЛИ РЕАЛИЗАЦИИ ПРОЦЕССА НЕПРЕРЫВНОЙ ИНТЕГРАЦИИ РАЗРАБАТЫВАЕМОГО ПРОГРАММНОГО СРЕДСТВА

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Агекян В.К.

Бахтизин В. В. – канд. техн. наук, доцент

Непрерывная интеграция — это практика разработки программных средств, которая заключается в автоматической и регулярной сборке и тестировании программных продуктов. Разрабатываемая модель позволяет оптимизировать время процесса интеграции для наиболее раннего выявления и устранения ошибок и противоречий.

В процессе разработки комплексных программных средств разработчики часто объединяют результаты своей работы друг с другом. Каждая интеграция проверяется автоматической сборкой и тестированием для скорейшего определения интеграционных проблем. Данный подход существенно повышает эффективность процесса разработки и качество программного средства. Усовершенствование процессов интеграции и тестирования позволяет своевременно выявлять и устранять дефекты уже в процессе разработки, а не эксплуатации или сопровождения.

Основной целью непрерывной интеграции является обнаружение проблем настолько быстро, насколько это возможно для скорейшего выявления ошибок. Если сборка системы и выполнение тестов занимают много времени, теряются основные преимущества в использовании данного подхода. Для комплексных программных средств наибольшее время при интеграции занимает, как правило, этап тестирования.

Для оптимизации времени выполнения тестов предлагается распределенная модель реализации системы непрерывной интеграции. Используется архитектура ведущих/ведомых серверов. Для работы системы непрерывной интеграции программного продукта выделяется кластер. Для синхронизации работы кластера необходим выделенный сервер, который является единым центром управления.

В задачи распределенной системы входят:

- получение исходного кода текущей версии продукта из системы контроля версии;
- сборка текущей версии продукта;
- выполнение тестов;
- формирование отчетов.

Сборку текущей версии продукта необходимо выполнить на каждом сервере из кластера. Далее исходный набор тестов делится на группы. Группы тестов добавляются в глобальную очередь выполнения. Тесты распределяются по группам таким образом, чтобы суммарное время выполнения набора тестов в каждой группе максимально совпадало. За счет этого достигается оптимальная загруженность выделенных серверов. Оценка времени выполнения каждого теста формируется как экспоненциально взвешенное скользящее среднее на основе времени предыдущих выполнений данного теста:

$$EMA_n = a * p_n + (1-a) * EMA_{n-1},$$

где p_n – время последнего успешного выполнения теста, a – сглаживающая константа.

Таким образом, необходимо хранить статистику о времени выполнения тестов. Каждый сервер в кластере после завершения выполнения набора тестов запрашивает центральный сервер и выполняет следующий набор тестов из очереди. После выполнения всех тестов статистика о результатах тестирования отправляется центральному серверу для формирования результирующего отчета.

Для реализации серверной логики используется язык программирования Python. Графический

интерфейс пользователя реализуется при помощи языка программирования JavaScript.

Применение подобного подхода существенно ускоряет процесс непрерывной интеграции посредством использования распределенных вычислений при выполнении интеграционного тестирования, что позволяет наиболее эффективно использовать преимущества практики непрерывной интеграции. Предлагаемый подход может быть внедрен в процесс разработки программного средства. Наибольшая эффективность достигается при внедрении в процесс разработки комплексного программного средства со строгими требованиями к качеству и большим набором интеграционных тестов.

Список использованных источников:

1. Дюваль, П. М.Непрерывная интеграция. Улучшение качества программного обеспечения и снижение риска /
2. П. М. Дюваль. – М. : Вильямс, 2008. – 240 с.
3. Хамбл, Д. Непрерывное развертывание программного обеспечения / Д. Хамбл. – М. : Вильямс,, 2011. – 432 с.
4. Криспин, Л. Scrum: гибкая разработка программного обеспечения / Л. Криспин. – М. : Вильямс,, 2011. – 432 с.

ПРИНЦИПЫ ПОВЫШЕНИЯ КАЧЕСТВА СПЕЦИФИКАЦИЙ ТРЕБОВАНИЙ WEB-ПРИЛОЖЕНИЙ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Чиркова А. Ю.

Бахтизин В. В. – к. т. н., доцент

В настоящее время создается огромное количество web-приложений, однако, большая их часть являются неудовлетворительными для конечного пользователя или заказчика или даже заканчиваются провалом. Во многих случаях причиной этому служит плохо составленная спецификация требований.

Разработке спецификации требований к web-приложениям зачастую уделяется недостаточно внимания в связи с тем, что web-приложения стереотипно не рассматриваются как сложные программные средства. Однако, вне зависимости от сложности и шаблонности разрабатываемого web-приложения, спецификация должна состоять из качественных требований. Качественные требования – это требования, которые выражают то, что основные заинтересованные лица действительно хотят получить как результат работы разрабатываемого программного продукта [1].

В спецификациях требований к web-приложениям нужно описать рамки проекта, это позволит избежать использования лишней информации. Рамки проекта пишутся в виде сценариев работы пользователей с web-приложением и описывают общую функциональность и интеракции с интерфейсом.

Ключевой частью спецификации является описание информационной архитектуры и интерфейсов. Информационная архитектура определят то, как будет выглядеть и работать web-приложение с пользователями.

Информационная архитектура состоит из структуры web-приложения (так называемые высокоуровневые прототипы), шаблонов страниц (низкоуровневые прототипы, описывающие непосредственно интерфейс), описания контента (описание содержания каждой страницы web-приложения).

Кроме того, с целью сокращения временных затрат на приемочные квалификационные испытания рекомендуется включать в спецификацию требований к web-приложениям описание верификации готового приложения.

При создании спецификации требований для web-приложения также следует руководствоваться ключевыми принципами, перечисленными в [2]:

Принцип выявления наиболее значимых целей и задач продукта. В спецификации требований важно описать, какую выгоду, ценность или результат поставляет проект.

Принцип применения различных техник при сборе требований. В настоящее время существует несколько техник сбора требований, но все они имеют свои достоинства и недостатки. Совокупность применения нескольких техник позволяет оптимизировать процесс сбора требований.

Принцип квантификации требований, или определения количественных мер в требовании. Использование чисел в требованиях является базовым и мощным методом определения качества требования.

Принцип разделения задач и решений. Требование должно быть составлено и абстрагировано таким образом, чтобы оно четко и однозначно определяло задачу, а не конкретное решение, которое может не описывать все ожидаемые результаты, быть недостаточно эффективным и т.д.

Принцип валидации требований. Все требования должны быть понятны всем участникам проекта. Все заинтересованные лица должны подтвердить, что они понимают все описанные требования, и что сформулированные требования соответствуют поставленным задачам.

Принцип использования методов управления качеством требований. Для снижения влияния субъективного мнения на оценку качества требований к программным средствам предлагается внедрять в процесс разработки спецификации метод управления качеством требований, базирующийся на наборе методологий совершенствования процессов CMMI[3].