Особенности разработки ПО в коллективе, контроль версий, подготовка документации

- При организации совместной разработки программного обеспечения и информационных систем в частности, существует множество стандартов как на государственном уровне, так и на уровне корпораций, предприятий и т. д.

При совместной работе следует решить для себя следующие вопросы:

- как одновременно несколько разработчиков могут создавать и корректировать одно и тоже множество файлов (программные модули, техническая документация и т. д.)?
- каким образом разработчик должен узнать о проблемах, возникших в созданном им программном обеспечении и как остальные должны узнать о том, что выявленные ими ошибки исправлены?
- как составить техническую документацию, отчеты о работе, справочную систему и т. д.?
- как добиться полной совместимости компьютера разработчика, на котором установлено огромное количество программного обеспечения, с компьютером будущего пользователя, у которого может быть другая операционная система и ограниченный набор программ, достаточный для выполнения функциональных обязанностей?
- как обеспечить достаточный уровень безопасности при взаимодействии разработчиков через локальные или глобальные сети?
- как одним действием можно установить сразу на пользовательский компьютер разработанное программное обеспечение?

Объекты, участвующие в разработке

- Заказчик (физическое или юридическое лицо, которое желает получить программный продукт и с которым заключены договорные отношения).
- Администрация (руководящее звено заказчика, уполномоченное решать финансовые и юридические вопросы).
- Рабочая группа по разработке системы (часть общей рабочей группы со стороны заказчика).
- Руководитель (лицо, отвечающее за ход разработки и внедрения со стороны заказчика).
- Специалисты по предметной области (представители тех подразделений заказчика, в которых будет внедряться новая система).
- Сотрудники IT подразделений (специалисты в области вычислительной техники со стороны заказчика: отдел ИТ, системные, сетевые и прочие администраторы).

Объекты, участвующие в разработке

- Исполнитель (также физическое или юридическое лицо, обязующееся на договорной основе выполнить определенный объем работ для заказчика)
- Администрация (то же, что и у Заказчика)
- Рабочая группа по разработке
- Руководитель проекта (координатор, главная задача которого заключается в организации работы коллектива)
- Системные аналитики (исследуют предметную область на выявление семантических зависимостей и бизнес-правил)
- Программисты (ведут конкретную работу по созданию программного обеспечения, его настройке, модификации, модернизации и исправлению ошибок)
- Тестировщики (со стороны исполнителя проводят проверку работоспособности функционала и качества пользовательского интерфейса)
- Техническая поддержка при внедрении (обучение персонала заказчика, дублирование его работы, решение вопросов, не требующих изменения ПО)

Субъекты разработки

- Финансово-правовая документация.
- Техническая документация.
- Справочная документация.
- База данных.
- Серверное программное обеспечение.
- Клиентское программное обеспечение.

Схема взаимодействия субъектов

- Права на доступ к информации обязательно должны регламентироваться, и это задача руководителя проекта. При этом в основном задействуются средства операционной системы и системы контроля версий.

Виды программного обеспечения, необходимые для организации работы
Системы контроля версий

- (Version Control Systems, VCS). Назначение систем контроля версий заключается в предоставлении нескольким работникам возможности вносить изменения в одни и те же файлы. Заметим сразу, что речь идет именно о текстовых файлах. Не важно – это простой текстовый файл или структурированный текстовый файл. Остальные файлы относятся к бинарным.
- Со стороны сервера VCS находится хранилище (репозиторий), в котором содержится информация о состоянии всех контролируемых файлов и истории их изменения. Со стороны клиента находится рабочая копия контролируемых файлов, изменения в которых могут фиксироваться в хранилище, и которая может получать последние (как правило) изменения из хранилища.

Виды программного обеспечения, необходимые для организации работы
Системы контроля версий

- (Системы контроля версий позволяют хранить, с одной стороны, историю всех изменений, внесенных участниками проекта (часто приводят пример «машины времени», когда имеется возможность вернуться к состоянию проекта в определенный момент времени), и учитывать все изменения, вносимые во все файлы.
- Практика показала, что любые попытки организации совместной работы без систем контроля версий приводит к постоянной потере вносимых изменений либо к ситуации, когда кто-нибудь из программистов не получает доступа к редактированию нужного файла.

Основные характеристики систем контроля версий

- При выборе конкретной системы контроля версий следует учитывать следующие свойства, которыми могут обладать те или иные версии. Характеристик достаточно много, но многие их них не оказывают реальной пользы при работе, поэтому можно ограничиться следующими.
- Система является коммерческой или бесплатной. Если вы платите за программу, то это еще не значит, что вы получили лучшее.

Основные характеристики систем контроля версий

- Atomic Commits – возможность атомарных фиксаций изменений, сделанных в рабочей копиив хранилище. В этом случае все изменения, во всех файлах рабочей копии, внесенные после последней фиксации в хранилище, будут зафиксированы после одной команды фиксации (commit).
- Организация хранилища. Централизованное хранилище поддерживает всех участников проекта и требует постоянно доступного сервера. Распределенное – подразумевает наличие копии хранилища у каждого участника наряду с рабочей копией.
- Разделение файлов. В модели «Блокирование–Изменение–Разблокирование» участник, редактирующий файл, блокирует доступ к нему до фиксации изменений в хранилище.

- Остальные участники при этом полностью лишены возможности редактирования этого файла. В модели «Копирование–Изменение–Слияние» один и тот же файл могут редактировать одновременно несколько участников. Если при этом они работают с различными частями файла (еще раз – это именно текстовый файл), то потом изменения сливаются. Если редактируется одна часть файла, то запрашивается, какое из изменений надо принять, и окончательное решение принимается обязательно в ручном режиме (это вопрос принципиальный, и подробнее о нем будет рассказано в следующих статьях цикла).
- Возможность ветвления. В определенной точке разработки проект может развиваться по двум различным направлениям (ветка, branch) совершенно независимо. Это может потребоваться для проверки нескольких вариантов его развития.

- перемещение и переименование файлов и директорий, копирование файлов и директорий, репликация, создание удаленной копии репозитория, разграничение доступа к различным частям репозитория, объединение группы связанных изменений в разных файлах в один логический блок, подробная история построчных изменений, возможность получения отдельной директории из репозитория, контроль изменений в рабочей копии до фиксации в репозиторий, задание отдельного текста комментария для отдельного файла при фиксации, наличие Web-интерфейсов пользователя, наличие GUI-интерфейсов пользователя, встроенная поддержка во многих IDE.

Подход к выбору системы контроля версий

- Однозначного подхода до сих пор не существует, однако можно следовать некоторым рекомендациям:
- сразу отбросить коммерческие продукты, они не настолько превосходят бесплатные, чтобы на это тратить деньги;
- следует отдавать предпочтение системам с атомарными фиксациями изменений, иначе вы обязательно что-нибудь пропустите при фиксации;
- очень неудобно работать с моделью, основанной на блокировке редактируемого файла(ваш партнер уехал в командировку и не разблокировал файл, который остался для всех остальных недоступным для редактирования);
- если у вас идет интенсивная работа над проектом, то удобнее иметь хранилище на общедоступном в любое время сервере.

- Всегда следует помнить, что главное в совместной работе – это четкая организация работы и исполнительская дисциплина всех до одного участников проекта.

документации
Почему не подходят стандартные офисные пакеты

- Процессу подготовки технической документации должно уделяться самое пристальное внимание. Кроме стандартных средств, к которым можно отнести текстовые процессоры (OpenOfficeWriter, MS Word и т. д.), широкий ряд продуктов Adobe (Adobe FrameMaker, AdobeRoboHelp, Adobe Captivate, Adobe Acrobat), на рынке присутствует ряд специализированных программ, предназначенных для автоматического формирования документации на основе программного кода (Javadoc, phpDocumentor и т. д.).
- Последняя группа слишком сильно ориентирована на программный код, что препятствует разработке таких документов, как «Руководство оператора», «Техническое задание» ипрочие, не основанные на программном коде.

- Первая группа более или менее подходит для небольших проектов с малым объемомтехнической документации, когда один (в лучшем случае) сотрудник (технический писательили техрайтер) полностью занимается всей документацией.
- Как только документация разрастается, различные разделы достаются разным сотрудникам,в связи с чем мгновенно возникают проблемы, связанные с необходимостью собиратьдокумент из кусков, подгонять форматирование под единые требования, пытаться вноситьизменения в занятые другими сотрудниками документы и т. д. Испытав на себе несколькораз такое, поневоле понимаешь – это ложный путь.

- Тут на помощь приходят специализированные средства (почему такое странное название, станет понятно позже), которые предоставляют нам следующие возможности:

- реальную многопользовательскую работу;

- отдельное хранение стилей оформления от собственно документации;

- возможность форматирования выходных документов в заданных форматах;

- легкость версионирования документации.

Основы DocBook

На самом деле DocBook вовсе не является системой для подготовки документов, а представляет собой набор правил для создания структурированного текстового файла. Этот файл состоит из набора тегов, и в этом отношении очень похож на широко известный HTML. DocBook поддерживает форматы XML и SGML. За счет этого он легко поддерживается системами контроля версий и, следовательно, с документами в этом формате могут свободно работать сотни пользователей одновременно.

Редактирование файлов возможно и в универсальном текстовом редакторе, но это требует самостоятельного контроля синтаксиса. Существует достаточно большое количество специализированных редакторов, позволяющих формировать документы в этом формате.

Стили форматирования хранятся отдельно и, следовательно, нет необходимости в согласовании формата каждого из документов.

Специальные утилиты позволяют формировать из готового файла документы в различных выходных форматах (например, PDF).

Однако не следует использовать DocBook для небольших документов, которые действительно проще подготовить в обычном текстовом редакторе.

Регистрация ошибок, замечаний и пожеланий в процессе разработки и эксплуатации

Необходимость документируемой (в электронном виде) системы мгновенного обмена информацией между субъектами, участвующими в процессе разработки ПО, вызвана разделением функций между участниками проекта.

Найденные ошибки, недостатки, неточности чаще всего не могут быть исправлены тестировщиками, группой внедрения, пользователями.

Взаимодействие участников в поиске и исправлении ошибок должно основываться на наличии технической возможности:

- зафиксировать свои замечания в базе данных;
- определить ответственного за решение проблемы;
- указать важность проблемы и сроки ее решения;
- показать степень ее решения.

Такие системы предназначены для отслеживания ошибок и позволяют организовать процесс коллективной отладки программного обеспечения.

В настоящее время широко распространены следующие программы.

Свободно распространяемые:

- Bugzilla
- BUGS – the Bug Genie
- eTraxis
- Mantis bug tracking system
- Trac
- EmForge

СРАВНЕНИЕ СИСТЕМ КОНТРОЛЯ ВЕРСИЙ

Признаки	GIT	MERCURUL	CVS

CVS	Достоинства	Недостатки
Git		
Mercural		
SVN		

- Repository
- Commit
- Branch
- Master
- Merge
- Push
- Pull
- Fetch