(слайд 1)

Экстремальное программирование (XP) и унифицированный процесс разработки представляют собой два основополагающих подхода в методологии создания программного обеспечения. Экстремальное программирование и унифицированный процесс выстраивают фундаментальные принципы, которые определяют ход процесса создания программных продуктов, однако, их фокус и стратегии существенно отличаются.

(слайд 2)

Экстремальное программирование — это подход разработки программного обеспечения, которая фокусирует усилия обеих сторон (программистов и бизнесменов) на общих, достижимых целях.

Данный подход ставит перед собой задачу обеспечить быструю разработку, акцентируя внимание на постоянном взаимодействии с заказчиком и оперативных реакциях на изменения требований.

Этот подход способствует созданию гибких, адаптивных продуктов в условиях динамичной рыночной среды.

Основная идея заключается в том, чтобы улучшить качество разработки программного обеспечения и повысить удовлетворенность заказчика через более гибкие и адаптивные практики.

(Слайд 3)

От других методик экстремальное программирование можно отличить по таким признакам как:

1. Использование чрезвычайно коротких циклов разработки
2. Планирование по нарастающей
3. Гибкий график реализации функциональности
4. Обмен информацией
5. Тесное взаимодействие программистов

(слайд 4)

Методика экстремального програмирования предназначена для работы над проектами, над которыми может работать группа программистов, которые не зажаты в жесткие рамки существующего компьютерного окружения и в которых вся необходимая работа, связанная с тестированием, может быть выполнена в течение одного дня. Можно выделить несколько главных принципов и методик экстремального программирования:

1. Простота
2. Изменчивые требования
3. Программирование в парах
4. Общее владение кодом
5. Тестирование
6. 40-часовая рабочая неделя
7. Заказчик на рабочей площадке

Эти принципы и методики взаимодействуют друг с другом, создавая гибкую и адаптивную среду для разработки программного обеспечения, способствуя улучшению качества продукта и ускорению процесса разработки. Однако важно отметить, что использование конкретных практик может изменяться в зависимости от конкретных условий проекта и команды.

(слайд 5)

К достоинствам экстремального программирования (если его удается применить) можно отнести:

1. Гибкость
2. Быстрое развертывание
3. Качество кода:
4. Участие заказчика
5. Уклонение от излишних формальностей

(слайд 6)

К недостаткам экстремального программирования можно отнести следующие пункты:

1. Недостаток структуры.
2. Сложности в масштабируемости.
3. Возможное недостаточное внимание к архитектуре.
4. Не все проекты подходят для экстремального программирования.
5. Невозможность долгосрочного планирования
6. Отсутствие предварительных исследований.

Как и любая методология, экстремальное программирование имеет как свои преимущества, так и недостатки, и применение метода на конкретном проекте должно осуществляться с учетом конкретных особенностей и контекста проекта.

(слайд 7)

Унифицированный процесс разработки — методология для построения процессов разработки программного обеспечения, позволяющий команде разработки преобразовывать требования заказчика в работоспособный продукт.

В зависимости от требований и доступных ресурсов, процесс разработки может быть адаптирован путём включения или исключения определённых проектных активностей.

Данная методика основана на объектно-ориентированном подходе к процессу разработки программного обеспечения, который обеспечивает методику управления жизненным циклом разработки ПО.

Унифицированный процесс стремится к созданию структурированной и комплексной системы управления проектом. Он ориентирован на документацию, строгие процессы и формализацию этапов разработки, что позволяет создавать крупные и сложные программные продукты с учетом широкого спектра требований и ограничений.

(слайд 8)

Унифицированный процесс обладает такими характеристиками как:

1. Итеративность
2. Инкрементальность
3. Артефакты
4. Архитектурное управление
5. Управление рисками
6. Поддержка различных типов процессов
7. Управление изменениями

Эти характеристики делают унифицированный процесс эффективным инструментом для разработки программного обеспечения, который позволяет управлять сложными проектами и обеспечивать высокое качество конечных продуктов.

(Слайд 9)

Каждый цикл разработки, при унифицированном процессе, состоит из четырёх фаз, представляющих собой промежуток времени между важными этапами проекта, позволяющими руководителям принять важные решения относительно продолжения процесса разработки, объёма работ, бюджета и расписания. Эти фазы включают:

1. Начальная фаза: необходима для выявления общей цели проекта, оценка его технической и экономической осуществимости, выявление основных рисков. Результатом является создание начального бизнес-плана и определение основной архитектуры системы.
2. Разрабатывающая фаза: необходима для проектирования архитектуры, уточнение требований, определение основных компонентов системы. Результатом является уточненная архитектура, план разработки, определенные риски и способы их устранения.

(Слайд 10)

1. Строительная фаза: включает в себя фактическую реализация и тестирование системы, разработка пошаговых инкрементов. Под конец будет рабочая версия системы.
2. Переходная фаза: необходима для развертывания системы, обучения пользователей, подготовки документации, поддержки и обслуживания. Результатом выполнения данной фазы является успешное развертывание системы, обеспечение поддержки и обновлений.

Каждая из этих фаз имеет свои цели, активности и результаты, и представляет собой важный этап в цикле разработки программного обеспечения с использованием унифицированного процесса.

(Слайд 11)

Унифицированный процесс, также как и другие методики программирования, имеет свои достоинства и недостатки. Можно выделить следующие достоинства унифицированного процесса разработки:

1. Итеративность и инкрементальность.
2. Гибкость и адаптивность.
3. Ориентация на архитектуру.
4. Управление рисками.

(Слайд 12)

Можно выделить следующие недостатки унифицированного процесса разработки:

1. Сложность.
2. Ресурсоемкость.
3. Не всегда подходит для малых проектов.
4. Неудовлетворительно для некоторых видов проектов.

Экстремальное программирование и унифицированный процесс разработки занимают центральное место в современной индустрии разработки программного обеспечения.

(слайд 13)

Экстремальное программирование больше всего подходит для различных проектов, особенно в условиях переменных требований, быстрого изменения среды и необходимости частого взаимодействия с заказчиком. Вот несколько типов проектов, для которых экстремальное программирование может быть особенно подходящим. Это такие проекты как:

1. Проекты с быстро меняющимися требованиями
2. Стартапы и инновационные проекты
3. Малые и средние проекты
4. Проекты с неопределенными или слабо определенными требованиями

(Слайд 14)

Унифицированный процесс разработки представляет собой гибридную методологию управления проектами и разработки программного обеспечения. Он может применяться в различных проектах, но чаще всего находит применение в средних и крупных проектах с достаточно сложными требованиями и потребностями. Вот несколько типов проектов, для которых унифицированный процесс разработки может быть подходящим:

1. Средние и крупные проекты разработки программного обеспечения

2. Проекты с переменными или изменяющимися требованиями

3. Проекты, где важно поддерживать высокое качество

4. Проекты с акцентом на архитектуре

5. Проекты с требованиями к документации

6. Распределенные проекты

Данные методики имеют свои преимущества и недостатки при определенных условиях разработки, но важно отметить, что выбор методологии зависит от множества факторов, включая размер команды, характер проекта, требования заказчика и т.д.

**Ожидаемые вопросы:**

**XP**

**Простота**: разработчики должны следовать принципу максимальной простоты при создании кода. Это включает в себя минимизацию сложности кода, избегания избыточности и лишних деталей, а также выбор наиболее простых и понятных решений. Основной вектор идет на создании наименьшего объема кода, который выполняет необходимые задачи.

**Континуальная поставка** - в экстремальном программировании принято за правило видеть результат своих действий настолько быстро, насколько это вообще возможно. Или, говоря техническим языком, обеспечить максимально быструю интеграцию нового кода в общую кодовую базу. Короткие циклы обратной связи являются ключевым элементом XP. Это означает, что изменения в коде должны тестироваться и интегрироваться в систему как можно быстрее. Это помогает выявлять проблемы, делать корректировки на ранних этапах разработки, уменьшить время на ввод новой функциональности в проекте.

**Обратная связь**: данный метод необходим чтобы получать обратную связь от заказчика и других участников процесса разработки. Это помогает убедиться, что продукт соответствует требованиям и может быстро реагировать на изменения.

**Изменчивые требования**: Требования могут меняться с течением разработки. XP предполагает, что требования могут меняться, и разработка должна быть способной быстро адаптироваться к этим изменениям.

**Сопровождение**: работать необходимо в устойчивом темпе, чтобы избежать перегрузки и усталости команды. Устойчивый темп способствует долгосрочной продуктивности. Работа над проектом должна вестись через короткие временные интервалы, называемые итерациями. Каждая итерация привносит новые функции и улучшения в проект, что позволяет быстро адаптироваться к изменяющимся требованиям.

**Программирование в парах**: разработчики работают в парах, совместно решая задачи. Один пишет код, а другой следит за процессом, предлагает идеи и обеспечивает обратную связь. Это способствует повышению качества кода, снижает вероятность ошибок и обмену знаниями.

**Общение**: В командах, работающих по методу XP, всегда приветствуется общение - самое быстрое средство обмена информацией и опытом. Это очень важно, когда требуется максимальная скорость разработки. Необходимость стимулировать активное и открытое общение между участниками процесса разработки. Эффективная коммуникация считается ключевым элементом успешной разработки.

**Общее владение кодом**: каждый разработчик может вносить изменения в любую часть кодовой базы. Это способствует распределению знаний и повышению гибкости команды.

**Тестирование**: тестирование играет ключевую роль в XP. Тесты пишутся до написания кода, и разработчики регулярно запускают все тесты, чтобы удостовериться в работоспособности системы.

**Рефакторинг**: разработчики регулярно улучшают структуру кода, делая его более понятным и поддерживаемым, без изменения его функциональности. Рефакторинг проводится в тесном взаимодействии с тестированием.

**Качество**: необходимо поддерживать высокое качество кода через практики тестирования, рефакторинг и другие техники, чтобы минимизировать количество ошибок и упростить поддержку кода.

**Планирование**: планы разработки строятся на основе актуальных требований и возможностей команды. Планирование в XP более гибкое и адаптивное, чем в традиционных методологиях.

**40-часовая рабочая неделя**: Сверхурочная работа рассматривается как признак больших проблем в проекте. Не допускается сверхурочная работа 2 недели подряд — это истощает программистов и делает их работу значительно менее продуктивной.

**Заказчик на рабочей площадке**: основной проблемой разработки программного обеспечения является недостаток знаний программистов в разрабатываемой предметной области. Экстремальное программирование предполагает, что заказчик должен принимать участие в процессе разработки.

**Достойнства**

**Гибкость**: XP позволяет быстро реагировать на изменения в требованиях и потребностях клиентов, возможность быстро и аккуратно вносить изменения в ПО в ответ на изменившиеся требования.

2. **Быстрое развертывание**: XP ставит основной упор на функциональность и быстроту выпуска программного обеспечения.

3. **Качество кода**: XP сосредотачивается на качестве и улучшении кода благодаря тестированию и интеграции.

4. **Участие заказчика**: методология включает заказчика в процесс разработки, что обеспечивает более точное соответствие продукта потребностям и ожиданиям.

5. **Уклонение от излишних формальностей**: вместо документации XP основывается на разработке и быстром обмене информацией в команде.

Недостатки

1. **Недостаток структуры**: некоторые разработчики могут столкнуться с нехваткой формализации, что приводит к хаосу в разработке.
2. **Сложности в масштабируемости**: XP может быть сложно масштабировать на крупные проекты и большие команды.
3. **Возможное недостаточное внимание к архитектуре**: из-за уклонения от излишней документации, архитектурный дизайн может страдать.
4. **Не все проекты подходят для XP**: некоторые проекты, особенно те, где требуются строгие процессы и документация, могут не подходить для XP.
5. **Невозможность долгосрочного планирования**: невозможность запланировать сроки и трудоемкость проекта на достаточно долгую перспективу и четко предсказать результаты длительного проекта в терминах соотношения качества результата и затрат времени и ресурсов
6. **Отсутствие предварительных исследований**: XP неприспособленна для проектов в которых возможные решения не находятся сразу на основе ранее полученного опыта, а требуют проведения предварительных исследований.

**Унифицированный процесс разработки**

1. **Итеративность**: УП поддерживает разработку через серию небольших итераций, в рамках каждой из которых происходит выполнение определенных задач. Этот подход позволяет быстрее получать обратную связь от заказчика и быстрее реагировать на изменения.
2. **Инкрементальность**: разработка в УП осуществляется путем добавления новой функциональности к системе постепенно и последовательно. Каждая итерация или инкремент добавляет новые возможности или улучшения к системе, что позволяет не просто создавать большие объемы кода, а давать конечный продукт клиентам на более ранних этапах разработки.
3. **Артефакты**: в рамках УП используются стандартные артефакты, такие как модели UML, документация требований, диаграммы, спецификации и т.д. Эти артефакты помогают визуализировать, документировать и управлять различными аспектами процесса разработки.
4. **Архитектурное управление**: УП уделяет особое внимание архитектуре системы, включая проектирование, документирование и поддержание соответствия архитектуры на протяжении всего процесса разработки. Акцент делается на создание стабильной архитектуры системы как базы для последующего развития.
5. **Управление рисками**: Методология УП включает в себя управление рисками, что позволяет выявлять и управлять рисками на ранних этапах разработки, что в свою очередь снижает вероятность возникновения проблем в будущем.
6. **Поддержка различных типов процессов**: УП гибко подходит для различных типов проектов.
7. **Управление изменениями**: УП предусматривает механизмы контроля изменений в требованиях и архитектуре системы.

**Главные отличия между экстремальным программированием и унифицированным процессом разработки**

1. Гибкость и размер проекта: XP обычно используется для малых и гибких проектов, где требуется быстрая реакция на изменения требований. UP, с другой стороны, чаще всего применяется для более крупных проектов с упором на формальность и структурированный подход.
2. Итеративность: Обе методологии применяют итеративный подход к разработке, однако в XP итерации обычно короче и более частые, чем в UP.
3. Основа процесса: XP базируется на четырнадцати основных практиках, таких как парное программирование и тестирование, в то время как UP включает в себя четыре базовые дисциплины: Бизнес-моделирование, Техническое проектирование, Реализацию и Тестирование.
4. Риск и изменения: XP сосредоточено на минимизации рисков и поддержании гибкости в проекте, в то время как UP уделяет больше внимания формализации процессов и управлению изменениями.
5. Роль команды: XP подчеркивает коллективную ответственность команды, в то время как UP имеет более жестко определенные роли и ответственности каждого участника процесса.