**1 слайд**

Здравствуйте, тема моей презентации спиральная модель ЖЦ: достоинства и недостатки.

**2 слайд**

Спиральная модель была впервые описана Барри Боэм в его статье 1986 года «Спиральная модель разработки и улучшения программного обеспечения».

Главной отличительной чертой спиральной модели является то, что она создает подход к процессу разработки ПО, основанный на рисках, а не на документах или коде.

Каждая фаза спиральной модели в разработке программного обеспечения начинается с определения цели проектирования и заканчивается тем, что заказчик просматривает прогресс. Всего 4 фазы на каждом витке.

**3 слайд**

Каждый веток спирали начинается с определения следующих пунктов:

* цель части разрабатываемого продукта (производительность, функциональность, способность приспосабливаться к изменениям и т. д.);
* альтернативные средства реализации этой части продукта (дизайн A, дизайн B, повторное использование, покупка и т. д.);
* ограничения, налагаемые на применение альтернатив (стоимость, график, интерфейс и т. д.).

**4 слайд**

Следующий шаг — оценка альтернатив относительно целей и ограничений. Часто этот процесс выявляет области неопределенности, которые являются значительными источниками проектного риска. Если это произошло, то следующий шаг должен включать формулирование экономически эффективной стратегии для устранения источников риска. Это включает в себя прототипирование, симуляцию и другие методы разрешения рисков.

Если этот прототип является операционно полезным и достаточно надежным, чтобы служить базой с низким риском для будущих итераций продукта, то последующие шаги, которые тоже созданы с учётом рисков, будут серией эволюционных прототипов.

**5 слайд**

Если предыдущие усилия по прототипированию уже устранили имеющиеся риски, то следующий шаг следует базовому каскадному подходу, а именно концепция работы, требования к программному обеспечению, проверка требований и т. д. В данной фазе идёт активная разработка, в результате которой будет законченный для этого этапа продукт.

Выявленные функции разрабатываются и проверяются посредством тестирования. В конце фазы доступна следующая версия программного обеспечения.

**6 слайд**

В данной фазе заказчики оценивают уже разработанную версию программного обеспечения и начинается планирование следующего этапа.

Планы для последующих фаз могут также включать декомпозицию продукта для последующей разработки или компоненты, которые будут разработаны отдельными организациями или лицами. В этом случае создаётся ряд параллельных спиральных циклов, по одному для каждого компонента. Например, отдельные спирали могут развиваться для отдельных программных компонентов или итераций.

**7 слайд**

Важной особенностью спинальной модели является то, что каждый виток и фаза завершаются обзором с участием основных людей или организаций, заинтересованных в продукте. Этот обзор охватывает все продукты, разработанные в течение предыдущего витка, включая планы на следующие ветки и ресурсы, необходимые для их выполнения. Основная цель обзора - гарантировать, что все заинтересованные стороны взаимно готовы для прохождения следующей фазы.

Сама спираль начинается с гипотезы о том, что конкретная операционная миссия (или набор миссий) может быть улучшена с помощью усилий по разработке программного обеспечения. Затем спиральный процесс включает проверку этой гипотезы: в любое время, если гипотеза не проходит проверку (например, если задержки приводят к тому, что программный продукт не попадает в свое рыночное окно или если появляется на рынке превосходящий коммерческий продукт), спираль прекращается. В ином случае она завершается установкой нового или модифицированного программного обеспечения, и гипотеза проверяется путем наблюдения за эффектом на операционную миссию. Обычно опыт с эксплуатационной миссией приводит к дальнейшим гипотезам об улучшениях ПО, и для проверки гипотезы запускается новая спираль обслуживания. Таким образом, инициирование, завершение и итерация задач и продуктов предыдущих циклов неявно определяются в спиральной модели.

**8 слайд**

Основное преимущество спиральной модели заключается в том, что ее диапазон опций включает в себя хорошие черты других существующих моделей, в то время как ее подход, ориентированный на риск, позволяет избежать многих их трудностей. Спиральная модель имеет ряд дополнительных преимуществ:

* На ранних этапах она фокусирует внимание на вариантах, включающих повторное использование существующего программного обеспечения. Этапы, включающие оценку альтернатив, поощряют эти варианты.
* Она обеспечивает подготовку к эволюции жизненного цикла, росту и изменениям программного продукта.
* Она предоставляет механизм для включения целей качества программного обеспечения в разработку программного продукта. Этот механизм вытекает из акцента на выявлении всех целей и ограничений в течение каждого витка спирали.
* Он фокусируется на устранении ошибок и непривлекательных альтернатив на раннем этапе. Этапы анализа рисков, проверки и принятия обязательств охватывают эти соображения.
* Для каждого из источников проектной деятельности и расходов ресурсов он отвечает на ключевой вопрос: «Сколько достаточно?» Другими словами, «Сколько анализа требований, планирования, управления конфигурацией, обеспечения качества, тестирования, формальной проверки и т. д. должен выполнять проект?». Можно увидеть, что ответ не одинаков для всех проектов и что соответствующий уровень усилий определяется уровнем риска, возникающего из-за недостаточного выполнения.
* Он обеспечивает жизнеспособную основу для разработки интегрированной аппаратно-программной системы. Фокус на управлении рисками и устранении непривлекательных альтернатив раньше и недорого в равной степени применим к аппаратному и программному обеспечению.

**9 слайд**

Модель полной спирали может быть успешно применена во многих ситуациях, но необходимо решить некоторые трудности. Основные проблемы это:

* Соответствие контрактному программному обеспечению. Внутренние разработки программного обеспечения обладают большой гибкостью и свободой для приспособления новых поэтапных обязательств, отсрочки обязательств для конкретных вариантов или приспособления таких практик, как прототипирование или проектирование по стоимости. Миру контрактного приобретения программного обеспечения сложнее достичь этих степеней гибкости и свободы без потери ответственности и контроля, и сложнее определения содержания контрактов, итоговые результаты которых не определены заранее.
* Спиральная модель во многом полагается на способность разработчиков программного обеспечения выявлять и управлять источниками риска проекта. Хорошим примером этого является спецификация спиральной модели, основанная на рисках, которая переносит элементы с высоким риском до большой степени детализации и оставляет элементы с низким риском для разработки на более поздних этапах; к этому времени риск поломки становится меньше. Однако команда неопытных или нерешительных разработчиков может также создать спецификацию с другим шаблоном изменения уровней детализации: большая проработка деталей для хорошо понятых, малорисковых элементов и малая проработка плохо понятых, высокорисковых элементов. Если нет проницательного обзора такой спецификации опытным персоналом по разработке или приобретению, этот тип проекта будет создавать иллюзию прогресса в период, когда он на самом деле движется к катастрофе.
* Еще одна проблема заключается в том, что спецификация, ориентированная на риск, также будет зависеть от людей. Например, проект, разработанный экспертом, может быть реализован неспециалистами. В этом случае эксперт, которому не нужно много подробной документации, должен подготовить достаточно дополнительной документации, чтобы не дать неспециалистам сбиться с пути. Рецензенты спецификации также должны быть чувствительны к этим проблемам.
* В целом, шаги процесса спиральной модели требуют дальнейшей разработки, чтобы гарантировать, что все участники разработки работают в согласованном контексте. Некоторые примеры этого — потребность в более подробных определениях спецификаций и этапов спиральной модели, характера и целей обзоров, методов оценки и синхронизации графиков, а также наличие индикаторов состояния модели и процедур отслеживания затрат и прогресса. Другая потребность — в руководствах и контрольных списках для выявления наиболее вероятных источников риска проекта и наиболее эффективных методов разрешения рисков для каждого источника риска. Опытные люди могут успешно использовать спиральный подход без этих разработок.

**10 слайд**

Основные условия, при которых спиральная модель становится эквивалентной другим основным моделям процессов следующие:

• Если проект имеет низкий риск в таких областях, как получение неправильного пользовательского интерфейса или несоответствие строгим требованиям к производительности, и если он имеет высокий риск в бюджете и предсказуемости и контроле расписания, то эти соображения риска приводят спиральную модель к эквивалентности каскадной модели.

• Если требования к программному продукту очень стабильны (что подразумевает низкий риск дорогостоящего проектирования и поломки кода из-за изменений требований во время разработки), и если наличие ошибок в программном продукте представляет высокий риск для миссии, которой он служит, то эти соображения о рисках приводят спиральную модель к сходству с двухсторонней моделью точной спецификации и формальной дедуктивной разработки программы.

• Если проект имеет низкий риск в таких областях, как потеря предсказуемости и контроля бюджета и графика; возникновение проблем интеграции крупных, и если он имеет высокий риск в таких областях, как получение неправильного пользовательского интерфейса или требований поддержки принятия решений пользователем, то эти соображения о рисках приводят спинальную модель к эквивалентности модели эволюционного развития.

• Если доступны возможности автоматизированной генерации программного обеспечения, то спиральная модель учитывает их либо как варианты быстрого прототипирования, либо как применения модели преобразования, в зависимости от связанных с этих рисков.

**11 слайд**

В презентации использовались следующие источники, в основном статья создателя.

**12 слайд**

Спасибо за внимание.