*Контрольная работа № 2*

*Синюта А.А.*

**Гибкие методы разработки**

Цель работы: знакомство с различными подходами в области проектного управления и разработки программных продуктов, отвечающими понятию «гибкие методы разработки». При выполнении необходимо самостоятельно изучить теоретический материал по указанным темам и составить сообщение объемом 8-10 стр., стараясь придерживаться примерно следующей ***структуры:***

- Название, общая идея;

- Появление (год, автор или группа авторов, организация, где методика была опробована, если такую информацию удалось найти);

- Отличительные особенности реализации, ограничения при применении;

- Достоинства и недостатки;

- Наличие программных средств, поддерживающих данный метод (если есть) или подходящих для него на взгляд автора;

- Уровень популярности, интереса со стороны пользователей в мире и/или в России;

- Перспективы развития (или указать метод, в который произошла трансформация).

**Метод Feature-Driven Development (FDD)**

**Введение**

В современном мире информационных технологий и постоянно меняющихся требований к продуктам разработка программного обеспечения становится всё более сложной задачей. Традиционные методы разработки, основанные на каскадной модели, часто оказываются неэффективными в условиях быстро меняющихся требований и неопределённости. В связи с этим возникает необходимость в использовании гибких методов разработки, которые позволяют быстро реагировать на изменения и обеспечивать высокое качество продукта.

Одним из таких методов является Feature-Driven Development (FDD), который представляет собой подход к разработке программного обеспечения, основанный на управлении функциональными возможностями (фичами) продукта. В данном докладе мы рассмотрим основные принципы и особенности метода FDD, его преимущества и недостатки, а также примеры его использования в реальных проектах.

**Название. Общая идея**

**Feature-Driven Development**переводится как «разработка, управляемая функциями».

Это один из гибких методов разработки программного обеспечения, при котором проект разбивается на отдельные функции, которые реализуются независимо друг от друга.

**Feature-Driven Development (FDD)** — это итеративно-инкрементальный метод разработки программного обеспечения, который фокусируется на создании отдельных функций (features) продукта.

Основная идея метода FDD заключается в том, чтобы разбить проект на отдельные функции, которые могут быть реализованы независимо друг от друга. Это позволяет более точно определить требования к каждой функции, а также упрощает процесс разработки и тестирования.

Метод FDD включает в себя несколько ключевых этапов:

* **Определение функций продукта.** На этом этапе проводится анализ требований к продукту и определяются основные функции, которые должны быть реализованы.
* **Планирование разработки.** На этом этапе разрабатывается план реализации каждой функции, включая определение необходимых ресурсов и сроков выполнения.
* **Разработка функций.** На этом этапе происходит непосредственная реализация функций продукта.
* **Тестирование функций.** На этом этапе проводится тестирование каждой функции, чтобы убедиться в её работоспособности и соответствии требованиям.
* **Интеграция функций.** На этом этапе происходит объединение всех реализованных функций в единый продукт.

Метод FDD позволяет обеспечить более чёткое управление проектом, а также повысить качество продукта за счёт более детального анализа требований и поэтапной реализации функций.

**Появление**

Feature-Driven Development (FDD) — это методология разработки программного обеспечения, которая была создана в 1997 году Джеффом Де Люкой (Jeff De Luca), Питером Коу (Peter Coad) и Дэном Томасом (Dan Tomayko).

Джефф Де Люка работал в компании Hewlett-Packard, где он столкнулся с необходимостью создания более эффективной методологии разработки. В результате он разработал FDD, основываясь на своём опыте работы с объектно-ориентированным программированием и другими методологиями разработки.

Питер Коу и Дэн Томашко также внесли свой вклад в развитие FDD, предоставив свои знания и опыт в области разработки программного обеспечения.

FDD изначально была опробована в компании Hewlett-Packard, а затем стала широко использоваться другими компаниями и организациями.

С тех пор FDD продолжает развиваться и совершенствоваться, оставаясь одним из популярных методов разработки программного обеспечения.

**Отличительные особенности реализации FDD:**

1. **Итеративный подход.** FDD предполагает разделение проекта на отдельные функции, которые реализуются итеративно. Это позволяет постепенно улучшать продукт, учитывая обратную связь от пользователей.
2. **Фокус на функциях.** В FDD основное внимание уделяется разработке отдельных функций продукта. Это позволяет более точно определить требования к каждой функции и упростить процесс разработки и тестирования.
3. **Планирование функций.** Перед началом разработки каждой функции проводится детальное планирование, включая определение необходимых ресурсов и сроков выполнения.
4. **Использование шаблонов проектирования.** В FDD используются шаблоны проектирования, которые помогают обеспечить повторное использование кода и улучшить качество продукта.
5. **Тестирование функций.** После разработки каждой функции проводится тестирование, чтобы убедиться в её работоспособности и соответствии требованиям.

**Ограничения при применении FDD:**

* **Сложность определения функций.** Определение функций может быть сложным процессом, особенно если требования к продукту нечёткие или постоянно меняются.
* **Зависимость от экспертов.** Для успешной реализации FDD требуется наличие экспертов, которые могут определить функции продукта и разработать план их реализации.
* **Необходимость планирования.** Планирование каждой функции требует времени и усилий, что может замедлить процесс разработки.
* **Риск потери фокуса.** Если не уделять достаточно внимания планированию функций, можно потерять фокус и потратить ресурсы на ненужные функции.

В целом, FDD является эффективным методом разработки, который позволяет обеспечить чёткое управление проектом и повысить качество продукта. Однако для его успешной реализации необходимо уделить внимание планированию и определению функций продукта.

Начало формы

**Преимущества FDD.**

**Наглядность.**

План работ составляется с использованием диаграмм, что позволяет визуализировать и систематизировать информацию. Таким образом, и команда, и заказчик будут иметь представление о том, каким будет продукт, когда будут реализованы функции и в какой последовательности. В процессе работы менеджер проекта имеет возможность следить за прогрессом и оптимизировать процессы, чтобы добиться наибольшей эффективности и не выходить за рамки установленных дедлайнов.

**Параллельная работа команд.**

В отличие от каскадного метода, FDD позволяет командам заниматься своими задачами одновременно. То есть тестировщики не будут сидеть без дела, пока разработчики пишут код для очередной функции внутри функции. Это, в свою очередь, позволяет сократить время разработки. Кроме того, таким образом можно своевременно обнаружить ошибки кода, которые требуется исправить, чтобы улучшить продукт.

**Постоянное взаимодействие внутри команды.**

Преимущество, которое присуще всем гибким методологиям. Когда каждый участник процесса вовлечен в разработку на каждом этапе, от составления плана до введения в эксплуатацию, то он выполняет свои обязанности более эффективно. Это происходит потому, что он знает весь проект от начала до конца и заинтересован не только в достижении собственных целей и задач, но и общих.

Кроме того, специалисты никогда не остаются наедине с проблемами, возникающими в процессе разработки. У них есть целая группа наставников в лице менеджера проекта, главного архитектора, менеджера разработки и главного программиста. Каждый из них может предоставить консультацию и помочь разрешить конфликты.

**Масштабизация.**

Данный метод позволяет масштабировать процессы разработки вместе с проектами и командой. То есть, если в начале мы имели средний по сложности и размерам продукт, то в процессе, когда потребуется его развитие, то FDD позволит это легко сделать.

**Гибкость.**

Каждая функция делится на ещё меньшие функции, что позволяет упростить процесс разработки. Также это дает возможность внедрять изменения в продукт, исправлять ошибки и улучшать его качество. Это, в свою очередь, повышает уровень удовлетворенности клиента.

**Рост и развитие команды.**

Опыт разработчиков и тестировщиков постоянно растет и меняется, так как они постоянно имеют дело с новыми проектами. Даже если они похожи между собой, то все равно обладают собственной спецификой. Более опытные разработчики могут делиться своим опытом и знаниями с другими членами команды, которые в этом нуждаются. Получается, что специалисты компании постоянно развиваются в полевых условиях.

**Недостатки FDD.**

**Метод рассчитан только на средние и большие проекты.**

В случае, если команда занимается разработкой простых продуктов, то данный подход не будет эффективным. Также он не подойдет в том случае, если при работе над проектом используется каскадный подход.

**Зависимость от главного программиста и ведущих разработчиков.**

Качество продукта и эффективность процесса разработки во многом зависит от уровня hard skill и soft skill главного программиста и ведущих разработчиков. То есть они должны обладать достаточно высоким уровнем компетенций и обширным опытом, чтобы иметь возможность решать задачи разной степени сложности. Если компания по какой–то причине потеряет одну из ключевых позиций, это может негативно отразиться на эффективности разработки.

**Не существует стандартной процедуры итерации.**

Это во многом зависит от специфики проекта, какими функциями он обладает, их количество и сложность. Поэтому команде каждый раз приходится начинать все с чистого листа, на это тратится немало времени и трудозатрат.

**Наличие программных средств, поддерживающих данный метод (если есть) или подходящих для него на взгляд автора;**

Для реализации метода Feature-Driven Development (FDD) можно использовать различные инструменты и программные средства. Выбор конкретных инструментов зависит от специфики проекта и предпочтений команды разработчиков.

Вот несколько популярных инструментов, которые могут быть полезны при использовании метода FDD:

1. **Инструменты управления проектами.** Для планирования функций и отслеживания прогресса проекта можно использовать инструменты управления проектами, такие как Jira, Trello или Asana.
2. **Интегрированные среды разработки (IDE).** Для разработки функций можно использовать интегрированные среды разработки, такие как Visual Studio, Eclipse или IntelliJ IDEA.
3. **Инструменты тестирования.** Для тестирования функций можно использовать инструменты тестирования, такие как Selenium, JUnit или TestNG.
4. **Системы управления версиями.** Для контроля версий кода можно использовать системы управления версиями, такие как Git или SVN.
5. **Шаблоны проектирования.** Для обеспечения повторного использования кода и улучшения качества продукта можно использовать шаблоны проектирования, такие как MVC, MVVM или MVP.

Выбор конкретных инструментов зависит от конкретных потребностей проекта и предпочтений разработчиков. Важно выбрать инструменты, которые соответствуют требованиям проекта и обеспечивают эффективное выполнение задач.

Конец формы

**Уровень популярности и интереса к методу Feature-Driven Development (FDD) в мире и в России**

Метод FDD является одним из гибких методов разработки программного обеспечения, который фокусируется на создании отдельных функций продукта. Этот метод может быть эффективным инструментом для управления проектами и повышения качества продукта.

В целом, популярность FDD можно объяснить его эффективностью и гибкостью, а также тем, что он подходит для разработки различных типов проектов.

11% компаний, которые постоянно используют FDD, вероятно, оценили его преимущества и адаптировали его под свои потребности. Они могут использовать его для разработки сложных и масштабных проектов, где требуется высокая степень контроля над процессом разработки.

31% компаний, которые время от времени используют FDD, могут использовать его для разработки отдельных функций или небольших проектов. Они могут выбирать FDD, когда им нужно быстро создать определённые функции продукта.

Таким образом, популярность FDD обусловлена его универсальностью и эффективностью. Он подходит для разработки различных типов проектов и может быть адаптирован под конкретные потребности команды и проекта.

В целом, популярность гибких методов разработки, включая FDD, растёт в последние годы. Это связано с тем, что гибкие методы позволяют быстро реагировать на изменения требований и обеспечивать высокое качество продукта в условиях неопределённости.

**Перспективы развития (или указать метод, в который произошла трансформация).**

**Перспективы развития FDD** связаны с его адаптацией под современные требования к разработке программного обеспечения. В частности, FDD может развиваться в следующих направлениях:

1. **Интеграция с другими методологиями.** FDD может быть интегрирован с другими методологиями разработки, такими как Agile или Scrum, чтобы создать более гибкий и эффективный процесс разработки.
2. **Использование современных инструментов.** FDD может использовать современные инструменты разработки, такие как системы управления проектами, для автоматизации процесса разработки и повышения его эффективности.
3. **Фокус на качестве.** FDD может уделять больше внимания качеству кода и тестированию, чтобы обеспечить высокое качество продукта и снизить риски возникновения ошибок.
4. **Обучение и сертификация.** FDD может развивать систему обучения и сертификации специалистов, чтобы обеспечить высокий уровень квалификации разработчиков и повысить доверие к методологии.
5. **Применение в различных областях.** FDD может применяться в различных областях разработки, таких как веб-разработка, мобильная разработка и разработка игр, чтобы обеспечить гибкость и эффективность процесса разработки.
6. **Развитие функциональной карты.** Функциональная карта может быть усовершенствована, чтобы лучше отражать потребности пользователей и обеспечивать более эффективное планирование разработки.
7. **Улучшение коммуникации.** FDD может улучшить коммуникацию между участниками проекта, чтобы обеспечить более эффективное взаимодействие и принятие решений.
8. **Адаптация под новые технологии.** FDD может адаптироваться под новые технологии, такие как искусственный интеллект и машинное обучение, чтобы обеспечить эффективное использование этих технологий в разработке.

В целом, перспективы развития FDD связаны с его способностью адаптироваться к изменяющимся требованиям и условиям разработки программного обеспечения.

Начало формы

Конец формы