Практическая работа  ***Исследование методологий разработки программного обеспечения***

Выполнили: Галуха Павел, Бражалович Александр, Гасюк Даниил, Барсуков Егор

***Цель***: исследовать методологии разработки ПО

***Задание:***

1. выполните исследование методологий. Результаты оформите в таблицу.

|  |  |  |
| --- | --- | --- |
| *Характеристика* | *Методология 1* | *Методология 2* |
| Полное название | Scrum | Feature driven development |
| Авторы | К. Швабер, Д. Сазерленд | Джефф де Лука |
| Год появления | 1995 | 1999 |
| Основные принципы | ● Итеративность и инкрементальность: Проект разбивается на короткие временные отрезки, называемые спринтами, обычно продолжительностью от одной до четырех недель. Каждый спринт приводит к созданию инкремента продукта, который представляет собой работающую версию продукта с добавленным функционалом.  ● Коллективная самоорганизация: Команда Scrum сама принимает решения о том, как достичь целей спринта. Вместо того, чтобы руководить командой, Scrum Master поддерживает команду и помогает ей устранять преграды.  ● Работа по приоритетам: Заказчик (Product Owner) определяет и приоритизирует требования продукта в Product Backlog. Команда выбирает самые важные элементы для выполнения в каждом спринте и сосредотачивается на достижении приоритетных целей.  ● Взаимодействие и прозрачность: Команда Scrum и заинтересованные стороны регулярно взаимодействуют, чтобы обсудить прогресс, проблемы и планы. Ежедневные стендапы (Daily Stand-ups) используются для обмена информацией о работе, а Sprint Review и Sprint Retrospective проводятся в конце каждого спринта для обсуждения достигнутых результатов и улучшения процесса.  ● Адаптивность и гибкость: Scrum основан на принципе непрерывного улучшения и адаптации. Команда Scrum регулярно оценивает свой прогресс и принимает меры для улучшения своей эффективности. Если требования меняются или возникают новые факторы, команда адаптирует свой план и приоритеты.  ● Доставка ценности для клиента: Scrum нацелен на создание ценности для клиента через регулярную итеративную поставку работающих инкрементов продукта. Ранняя и частая доставка функциональности позволяет клиенту получать обратную связь и вносить коррективы в продукт. | ● Разработка по функциональным возможностям: FDD фокусируется на разработке итеративно и инкрементально по функциональным возможностям (features) продукта. Каждая функциональная возможность разрабатывается отдельно и приводит к созданию работающего компонента или модуля.  ● Короткие итерации: Разработка в FDD осуществляется через короткие итерации, называемые "двухнедельными обзорами" (two-week builds). Каждая итерация фокусируется на реализации одной или нескольких функциональных возможностей и приводит к созданию работающего инкремента продукта.  ● Распределение ответственности: FDD пропагандирует распределение ответственности между разработчиками. Каждая функциональная возможность назначается конкретному разработчику или группе разработчиков, которые становятся ответственными за ее реализацию.  ● Команда разработки: FDD подчеркивает важность сформирования кросс-функциональных команд разработки. Команда включает разработчиков с различными навыками, такими как анализ требований, проектирование, программирование, тестирование и т. д. Это способствует более эффективной разработке функциональных возможностей.  ● Инспекция и адаптация: FDD предусматривает систематическую инспекцию и адаптацию процесса разработки. Регулярные проверки и обзоры помогают выявлять проблемы и вносить коррективы в процесс, чтобы улучшить его эффективность и результативность.  ● Модель прогресса разработки: FDD использует модель прогресса разработки, которая включает пять этапов: формирование общего видения (Develop an Overall Model), разработка списка функциональных возможностей (Build a Features List), планирование по функциональным возможностям (Plan by Feature), разработка по функциональным возможностям (Design by Feature), и сборка по функциональным возможностям (Build by Feature). Эти этапы обеспечивают структурированность и организацию процесса разработки. |
| Специализированное ПО (при наличии) | ● Jira: Jira является одним из самых популярных инструментов управления проектами, который широко используется для управления процессом разработки по Scrum. Он предоставляет возможности для создания и отслеживания задач, управления бэклогом продукта, планирования спринтов, отслеживания прогресса и многое другое.  ● Azure DevOps: Azure DevOps (ранее известный как Visual Studio Team Services) представляет собой набор инструментов, который включает в себя возможности управления проектами Agile, включая Scrum. Он предоставляет функциональность для создания и управления бэклогом продукта, планирования спринтов, отслеживания задач и контроля версий.  ● Trello: Trello - это визуальная доска и инструмент управления проектами, который может быть адаптирован для работы по методологии Scrum. Он позволяет создавать карты задач (карточки), организовывать их в списки, устанавливать сроки, отслеживать прогресс и делиться информацией с командой.  ● Rally (CA Agile Central): Rally является инструментом для управления проектами Agile, который поддерживает работу по Scrum. Он предоставляет возможности для планирования спринтов, управления бэклогом продукта, отслеживания прогресса, создания отчетов и многое другое.  ● Scrumwise: Scrumwise - это онлайн-инструмент, специально разработанный для управления проектами по методологии Scrum. Он предлагает функциональность для создания и управления задачами, планирования спринтов, отслеживания времени, обмена информацией с командой и других связанных функций. | ● Microsoft Project: Microsoft Project - это инструмент управления проектами, который может быть адаптирован для работы по методологии FDD. Он предоставляет функциональность для создания и управления планами проекта, распределения задач, отслеживания прогресса, установки сроков и ресурсного планирования.  ● Jira: Как было упомянуто ранее, Jira является популярным инструментом управления проектами, который может быть настроен для работы по FDD. Он позволяет создавать и отслеживать задачи, управлять бэклогом функциональных возможностей, планировать и отслеживать прогресс, а также взаимодействовать с командой разработки.  ● Trello: Trello также может быть адаптирован для работы по FDD. Он предоставляет возможности для создания карточек задач, организации их в списки, установки сроков, отслеживания прогресса и обмена информацией с командой  ● Azure DevOps: Azure DevOps (ранее известный как Visual Studio Team Services) поддерживает работу по Agile-методологиям разработки и может быть настроен для работы в рамках FDD. Он предоставляет функциональность для планирования и отслеживания задач, управления бэклогом продукта, контроля версий и других связанных функций. |
| Критичность изменения требований к ПС на поздних этапах ЖЦ ПО | ● Влияние на планирование и бюджет: Изменение требований может потребовать пересмотра планов и бюджета проекта. Это может привести к задержкам и увеличению затрат.  ● Влияние на спринт: Если требования меняются в середине спринта, это может привести к нарушению планирования и невозможности выполнения запланированной работы.  ● Риски качества: Изменение требований может повлечь за собой риск неполного или некорректного тестирования новых функций. Это может привести к ошибкам и нежелательным последствиям для продукта.  ● Влияние на команду разработки: Постоянные изменения требований могут оказывать стресс на команду разработки, особенно если они происходят на поздних этапах проекта. Команда может испытывать затруднения в планировании и выполнении работы. | ● Влияние на планирование и бюджет: Изменение требований может потребовать пересмотра планов и бюджета проекта. Это может привести к задержкам и увеличению затрат.  ● Влияние на моделирование и проектирование: В FDD функции являются основой для моделирования и проектирования системы. Изменение требований может потребовать пересмотра моделей и проектной документации.  ● Риски качества: Изменение требований может повлечь за собой риск неполного или некорректного тестирования новых функций. Это может привести к ошибкам и нежелательным последствиям для продукта.  ● Влияние на команду разработки: Постоянные изменения требований могут оказывать стресс на команду разработки, особенно если они происходят на поздних этапах проекта. Команда может испытывать затруднения в планировании и выполнении работы. |
| Нужно ли реализовать основные функции на ранних этапах разработки | На начальных этапах проекта в Scrum формулируются основные требования и создается начальный план разработки. В каждом спринте команда разработки выбирает определенное количество задач или функций, которые они планируют реализовать. Приоритет отдается наиболее важным функциям, которые принесут ценность заказчику. | На начальных этапах проекта в FDD проводится процесс создания общей модели системы и определения основных фич, которые предоставят ценность заказчику. Затем эти фичи декомпозируются на более мелкие задачи для реализации в рамках каждой итерации. |
| Новизна проблем предметной области | ● Масштабирование Scrum: Когда проекты становятся более крупными и команды становятся более крупными или распределенными, возникает необходимость в масштабировании Scrum. Это может включать координацию работы нескольких команд, управление общим бэклогом продукта и обеспечение согласованности процессов.  ● Интеграция с другими методологиями: В некоторых случаях команды разработки должны интегрировать Scrum с другими методологиями разработки, такими как DevOps или Waterfall. Это может представлять трудности в обеспечении согласованности и эффективного взаимодействия между различными подходами.  ● Вовлечение заинтересованных сторон: Scrum акцентирует внимание на вовлечении заинтересованных сторон, включая заказчиков и пользователей продукта. Однако, в реальном мире может быть сложно обеспечить активное участие заинтересованных сторон и получить своевременную обратную связь.  ● Управление зависимостями: В процессе разработки могут возникать зависимости между различными задачами или командами. Управление этими зависимостями может быть сложной задачей и требовать координации и согласования.  ● Управление изменениями: Scrum обладает гибкостью для внесения изменений в требования и функциональности продукта. Однако, управление изменениями может быть сложным процессом, особенно при работе с большим количеством заинтересованных сторон и изменяющимися требованиями. | ● Недостаток понимания концепции фич: Одной из основных проблем может быть недостаточное понимание концепции "фич" и определение их границ. Определение четкого перечня и описания фич требует хорошего понимания требований заказчика и способности разбивать функциональность на более мелкие и понятные элементы.  ● Управление приоритетами: Определение приоритетов фич и управление их доставкой являются важными аспектами FDD. Однако, в реальных проектах может возникать сложность при определении приоритетов и управлении изменяющимися требованиями заказчика. Необходимо разработать эффективные методы приоритизации и обеспечить прозрачность и согласованность в процессе управления фичами.  ● Разделение работы и координация: В FDD фичи разбиваются на более мелкие задачи, которые выполняются параллельно различными командами или разработчиками. Это может привести к необходимости эффективной координации работы и обеспечению согласованности между членами команды. Коммуникация и согласование становятся важными аспектами для успешного выполнения фич.  ● Необходимость непрерывного обновления документации: FDD предполагает наличие детальной документации о каждой фиче, включая ее описание, предусловия и ожидаемые результаты. Поддержание актуальности документации может быть сложной задачей, особенно при изменении требований и появлении новых фич. Необходимо разработать эффективные методы управления документацией и обновления ее в соответствии с изменениями в проекте.  ● Масштабирование FDD: При работе над крупными проектами или в организациях с несколькими командами может возникнуть необходимость в масштабировании FDD. Координация и управление различными командами, управление общим бэклогом фич и обеспечение согласованности процессов между командами могут представлять сложности. |
| Критичность временных рамок реализации проекта | ● Увеличение риска: Если спринты становятся слишком долгими или несистематичными, это увеличивает риск непредсказуемости и неудачного выполнения задач.  ● Потеря прозрачности: Длинные временные рамки могут снизить прозрачность разработочного процесса, замедлив выявление проблем и задержек, что затрудняет получение оперативной обратной связи от заказчика.  ● Снижение эффективности: Неясные временные рамки могут размыть фокус команды и ухудшить её производительность.  ● Рост затрат: Нарушение сроков может повлечь за собой дополнительные расходы на проект, так как неопределенность и задержки могут потребовать дополнительных ресурсов и усилий для достижения поставленных целей. | ● Увеличение риска: Несоблюдение сроков увеличивает риски проекта, вызывая неопределенность и возможные негативные последствия.  ● Ухудшение прозрачности: Несоблюдение временных рамок может сбивать с толку заказчика, вызывая сомнения и ухудшая восприятие процесса разработки.  ● Задержки в доставке: Неаккуратное соблюдение сроков может вызывать задержки в выпуске новых функций, что в конечном итоге может нарушить график всего проекта.  ● Неэффективное использование ресурсов: Несоблюдение временных рамок может заставить команду разработки заниматься неприоритетными задачами, что приводит к неэффективному использованию времени и ресурсов.  ● Низкое качество продукта: Неисполнение сроков может вынудить команду ускорить процесс тестирования и обеспечения качества, что может привести к неполноценному тестированию и появлению ошибок в функциональности, ухудшая общее качество продукта. |
| Возможность увеличения бюджета | ● Увеличение бюджета на конкретный проект или спринт: В Scrum обычно фиксируется бюджет на период спринта или проекта. Однако, могут возникнуть ситуации, требующие дополнительных ресурсов или средств. Если такое потребление возникает, команда и заинтересованные стороны проводят обсуждение и соглашаются о дополнительном финансировании. Это может потребовать обоснования необходимости ресурсов и оценки их влияния на график и цели проекта.  ● Увеличение бюджета для развития и поддержки Scrum-процесса: Scrum предлагает набор практик и ролей, которые иногда требуют дополнительных инвестиций для обеспечения эффективной реализации методологии. Это может включать в себя обучение и сертификацию членов команды, найм специалистов Scrum-мастеров или коучей, использование специализированного программного обеспечения или инструментов для управления проектами. В таких случаях организация может рассмотреть возможность расширения бюджета для поддержки этих дополнительных требований. | ● Увеличение бюджета проекта FDD: Если в ходе разработки FDD-проекта возникает потребность в дополнительных ресурсах или финансировании, команда и заинтересованные стороны могут обсудить возможность увеличения бюджета. Это может включать в себя привлечение дополнительных разработчиков, покупку специализированного оборудования или инструментов, а также финансирование дополнительных этапов тестирования и обеспечения качества. Решение о повышении бюджета принимается на основе оценки необходимости и ожидаемой выгоды от такого решения.  ● Увеличение бюджета для развития и поддержки процесса FDD: Поддержка FDD-процесса может включать не только финансирование самого проекта, но и предоставление необходимых ресурсов и инфраструктуры для успешной реализации методологии. Это может включать в себя обучение и сертификацию команды по FDD, найм опытных разработчиков, использование специализированного программного обеспечения или инструментов, а также внедрение дополнительных процессов и практик для улучшения эффективности разработки. В этом случае компания или организация может рассмотреть возможность увеличения бюджета для поддержки развития и оптимизации процесса FDD. |
| Примеры успешных проектов | ● Spotify: Музыкальная платформа Spotify успешно применяла Scrum для итеративной разработки новых функций и улучшений, обеспечивая гибкую реакцию на потребности пользователей и изменения в индустрии стриминговой музыки.  ● Salesforce: Ведущая компания в области CRM-систем, Salesforce, также использует Scrum, чтобы быстро адаптироваться к требованиям клиентов и регулярно предоставлять новые функции и улучшения.  ● Adobe: Компания Adobe, специализирующаяся на программном обеспечении для креативных профессионалов, успешно внедряет Scrum в своих проектах, что позволяет ей эффективно удовлетворять меняющиеся требования отрасли и быстро выпускать инновационные продукты.  ● Toyota: Даже компании вне сферы программного обеспечения, как Toyota, используют Scrum. Например, Toyota внедрила Scrum в разработке автомобилей, чтобы обеспечить более гибкую и эффективную разработку новых моделей и технологий.  ● Microsoft: Одна из крупнейших IT-компаний, Microsoft, также успешно применяла Scrum в разработке программного обеспечения, такого как операционные системы и офисные приложения, чтобы быстро предоставлять новые функции и обновления пользователям. | ● Singapore Stock Exchange (SGX): Разработка системы торговли и расчетов для Singapore Stock Exchange была успешно завершена с использованием FDD. Это позволило эффективно управлять разработкой, акцентируя внимание на конкретных фичах и обеспечивая быструю поставку ценностей для бизнеса.  ● Boeing 777: Разработка самолета Boeing 777 также прошла с применением FDD, что позволило команде разработчиков эффективно планировать и выполнять работу, сосредотачиваясь на функциональных возможностях самолета и обеспечивая высокое качество и надежность.  ● UBS Bank: UBS Bank, один из крупнейших банков в мире, использовал FDD при разработке своих банковских систем. Это помогло им эффективно управлять разработкой функциональностей, предоставлять новые банковские продукты и услуги, а также обеспечивать высокий уровень безопасности и надежности системы.  ● Telstra: Telstra, крупнейший телекоммуникационный оператор в Австралии, также воспользовался FDD при разработке своих телекоммуникационных систем. Это позволило им эффективно управлять функциональностями, такими как мобильная связь, широкополосный доступ в Интернет и другие услуги, обеспечивая высокую скорость и качество связи.  ● National Australia Bank (NAB): National Australia Bank применил FDD при разработке своих банковских систем и приложений, что успешно позволило управлять проектами, фокусируясь на разработке и поставке конкретных банковских функций, сокращая время разработки и повышая качество продукта. |
| Достоинства методологии | ● Гибкость и адаптивность: Scrum - гибкая методология разработки, позволяющая командам быстро адаптироваться к изменениям требований и среды. Команды Scrum гибко реагируют на новые знания и взаимодействуют с заинтересованными сторонами для определения наиболее ценных фич.  ● Итеративность и инкрементальность: Scrum основан на итеративной и инкрементальной разработке. Проект разбивается на спринты, где команда создает, тестирует и предоставляет инкременты продукта. Это обеспечивает быструю обратную связь и корректировку планов разработки.  ● Коллаборация и командная работа: Scrum стимулирует работу в команде. Все участники активно участвуют в планировании, оценке, разработке и обсуждении проблем. Это улучшает коммуникацию, совместное принятие решений и снижает риски.  ● Прозрачность и видимостьScrum обеспечивает прозрачность и видимость прогресса разработки. Стендапы, ревью и ретроспективы спринта позволяют ясно видеть достижения этого и ожидания от следующего спринта.  ● Контроль и управление рисками: Scrum управляет рисками с помощью инспекции и адаптации. Это позволяет команде быстро реагировать на проблемы и изменения, минимизируя риски и повышая вероятность успешной доставки продукта.  ● Улучшение качества: Scrum улучшает качество продукта. Регулярное тестирование, интеграция и обратная связь от заинтересованных сторон помогают команде быстро обнаруживать и исправлять ошибки, улучшать процессы и достигать высокого качества. | ● Акцент на функциональности: FDD сфокусирована на разработке фич продукта. Это помогает команде ясно определить задачи и цели, способствуя эффективной разработке.  ● Инкрементальная и итеративная разработка: FDD предполагает инкрементальную и итеративную разработку, где проект разбивается на полностью функциональные модули. Это помогает команде доставлять работающие инкременты продукта на ранних стадиях, что способствует обратной связи и удовлетворению требований заказчика.  ● Эффективное планирование и управление: FDD предлагает структурированный процесс планирования и управления разработкой. Команды FDD используют списки функциональностей и графики прогресса для контроля проекта, планирования ресурсов и управления рисками.  ● Сотрудничество и распределение ответственности: FDD поощряет сотрудничество и распределение ответственности в команде разработки. Каждая фича назначается отдельному разработчику, повышая вовлеченность и ответственность каждого участника.  ● Эмпирический подход: FDD основана на опыте и реальных данных. Решения принимаются на основе фактов и результатов, позволяя команде быстро адаптироваться и принимать обоснованные решения.  ● Быстрая поставка ценности: FDD приоритизирует быструю поставку ценности для бизнеса. Разработка фич происходит по приоритету, что позволяет команде доставлять полезные результаты в короткие сроки, удовлетворяя потребности заказчика. |
| Предполагаемые недостатки методологии | ● Сложность для больших и сложных проектов: Scrum вызывает трудности при применении к большим и сложным проектам. Управление большими командами и масштабирование становятся сложными задачами. Синхронизация между командами и управление зависимостями могут вызывать проблемы.  ● Зависимость от сильной команды и культуры: Scrum требует высокой самоорганизации и сотрудничества в команде. Участники должны быть самодостаточными, принимать решения и эффективно сотрудничать. Внедрение Scrum может быть сложным, если команда не обладает достаточным опытом или культурой сотрудничества.  ● Ограничения на предоставление деталей и документации: Scrum используется для быстрой разработки и поставки работающих инкрементов продукта. Это может привести к недостатку документации и деталей, таких как спецификации и архитектура, и вызвать проблемы в поддержке и сопровождении продукта.  ● Трудности в оценке времени и планировании: Scrum усложняет оценку времени и планирование из-за итеративной разработки. Команды могут столкнуться с трудностями в определении объема работы на спринт и прогнозировании сроков, возникает риск превышения сроков.  ● Ограниченность формальных процессов и документирования: Scrum ставит акцент на гибкость и адаптивность, что может привести к недостатку формальных процессов и документирования. Некоторые организации и проекты могут требовать более строгих процессов, контроля и документирования,  которые могут быть недостаточно поддержаны Scrum. | ● Ограниченность масштабирования: FDD может столкнуться с ограничениями при масштабировании на большие или распределенные проекты. Управление большими командами и согласование работы между ними могут быть сложными в рамках FDD.  ● Недостаток гибкости: FDD - структурированная методология, что ограничивает гибкость и адаптивность по сравнению с Scrum или Agile. Это проблематично при необходимости быстро реагировать на изменения в требованиях или условиях проекта.  ● Ограниченная коммуникация: FDD не ставит такой акцент на коммуникации и сотрудничестве внутри команды, как другие методологии. Это может снизить эффективность командной работы и взаимодействия разработчиков.  ● Ограниченное управление зависимостями: FDD не предоставляет явных инструментов или процессов для управления зависимостями между функциональностями или модулями. Это может усложнить управление и разрешение конфликтов между разными частями проекта.  ● Недостаток формальной документации: FDD уделяет меньше внимания формальной документации, такой как подробные технические спецификации или архитектурные документы. Это может затруднить поддержку и сопровождение продукта в долгосрочной перспективе. |