Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

РЕФЕРАТ

**AGILE РАЗРАБОТКА**

Студента 1 курса 141 группы  
направления 02.03.02 Математическое обеспечение и администрирование информационных систем

факультета компьютерных наук и информационных технологий

Кулагина Андрея Юрьевича

Саратов 2018

**ОГЛАВЛЕНИЕ**

[Введение…………………………………………………………………….....3](#п)

[История………………………………………………………………………...4](#в)

[12 принципов Agile разработки……………………………………………....5](#т)

[Главные постулаты Agile разработки……………………………………......6](#ч)

[Преимущества Agile подхода…………………………………………………7](#пять)

[Практики Agile…………………………………………………………………8](#ш)

[Примеры Agile методологий………………………………………………….12](#семь)

[Вывод…………………………………………………………………………...15](#восемь)

[Список литературы...…………………………………………………………..16](#дев)

**Вве****дение**

Разработка программного обеспечения — это труд, который требует своевременного принятия правильных решений. CTO, архитекторы, тимлиды и сами разработчики регулярно делают выбор в пользу тех или иных инструментов, платформ и фреймворков.

Но все принимаемые решения нужно как-то «синхронизировать». Один из резидентов Hacker News отметил, что ему приходилось наблюдать за тем, как пяти сотням разработчиков в одной крупной компании разрешили самостоятельно принимать некоторые решения в «отрыве» от команды. По его словам, это был хаос. И хотя команда начала работать быстрее, она двигалась в никуда, потому что позднее возникали проблемы на этапах поддержки ПО.

Чтобы избежать подобных ситуаций, используется семейство процессов гибкой разработки. Их внедрение позволяет руководству компании повысить заинтересованность и мотивацию сотрудников, а также ускорить доставку продукта заказчику. В последнее время эта тема приобретает все большую популярность, потому что на некоторые методологии Agile обращают внимание общественности главы крупнейших корпораций.

**Исто****рия**

В 2001 году, 17 разработчиков, которые называли себя «организационными анархистами» встретились в городе Сноубёрд (Snowbird, Utah) чтобы поделиться идеями повышения эффективности. Сазерленд и другие сторонники Scrum были среди них. Но в группу входили приверженцы нескольких конкурирующих подходов: экстремального программирования (ХР), crystal, adaptive software development (ASD), feature-driven development (FDD); и dynamic-systems-development method (DSDM). Все эти подходы были известны как «лёгкие» фреймворки, потому что они используют более простые правила и процессы для быстрой адаптации к изменяющейся среде. Но не всех участников устраивала существующая терминология и название «лёгкие фреймворки».

Несмотря на разногласия, участники в конце концов выбрали новое название для движения: Agile. Название подсказал один из участник, который тогда читал книгу Agile Competitors and Virtual Organizations: Strategies for Enriching the Customer. В этой книге даётся 100 примеров компаний, включая АВВ, Federal Express, Boeing, Bose и Harley-Davidson, которые придумывали оригинальные способы бороться с неопределённостью и турбулентностью среды. После того, как имя было выбрано, участники съезда сформулировали знаменитый «Agile-манифест разработки программного обеспечения» (Manifesto for Agile Software Development), в котором были сформулированы ключевые ценности и принципы, которые разделили все участники. С 2001 года, все техники разработки, разделяющие эти ценности называются гибкими методологиями Agile.

Конечно, отдельные положения Agile-подхода появились появлялись и до этого, но только этот документ систематизировал и изложил их в достаточной для использования мере. Каждый год под манифестом подписываются новые компании, IT-специалисты и проектные менеджеры. Появляются новые методы и модификации гибкой системы разработки.

**12 принцип****ов Agile разработки**

1. Наивысшим приоритетом является удовлетворение потребностей заказчика, благодаря регулярной и ранней поставке ценного программного обеспечения.
2. Изменение требований приветствуется, даже на поздних стадиях разработки. Agile-процессы позволяют использовать изменения для обеспечения заказчику конкурентного преимущества.
3. Работающий продукт следует выпускать как можно чаще, с периодичностью от пары недель до пары месяцев.
4. На протяжении всего проекта разработчики и представители бизнеса должны ежедневно работать вместе.
5. Над проектом должны работать мотивированные профессионалы. Чтобы работа была сделана, создайте условия, обеспечьте поддержку и полностью доверьтесь им.
6. Непосредственное общение является наиболее практичным и эффективным способом обмена информацией как с самой командой, так и внутри команды.
7. Работающий продукт — основной показатель прогресса.
8. Инвесторы, разработчики и пользователи должны иметь возможность поддерживать постоянный ритм бесконечно. Agile помогает наладить такой устойчивый процесс разработки.
9. Постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта.
10. Простота — искусство минимизации лишней работы — крайне необходима.
11. Самые лучшие требования, архитектурные и технические решения рождаются у самоорганизующихся команд.
12. Команда должна систематически анализировать возможные способы улучшения эффективности и соответственно корректировать стиль своей работы.

**Главные постулат****ы Agile разработки**

1. Люди и взаимодействие важнее процессов и инструментов
2. Работающий продукт важнее исчерпывающей документации
3. Сотрудничество с заказчиком важнее согласования условий контракта
4. Готовность к изменениям важнее следования первоначальному плану

**Преимущества** **Agile подхода**

*Удовлетворенность и вовлеченность клиента*

Agile разработка предоставляет большое количество возможностей для взаимодействия клиента и разработчиков во время каждого спринта. Клиент постоянно общается с командой. Это позволяет разработчикам полностью понимать взгляд заказчика на проблему и часто выпускать качественные обновления, что ведет к увеличению доверия и последующему сотрудничеству между сторонами.

*Прозрачность*

При использовании Agile подхода клиент активно участвует в проекте с самого начала – планирует итерации, ставит задачи, расстанавливает приоритеты.

Стоит отметить - клиент должен понимать, что он видит лишь проект, над которым еще ведется работа, а не финальный продукт.

*Короткие и предсказуемые сроки выхода*

Спринты – отрезки времени от 1 до 4 недель. Итогом каждого спринта является релиз обновления. При использовании подобного подхода (работа в рамках ограниченного количества времени), новый функционал добавляется быстро и часто. Понять, когда будет очередное обновление, легко.

*Прогнозируемые сроки и бюджет*

Так как расписание спринтов фиксировано, стоимость разработки предсказуема и ограничена. Она напрямую зависит от проделанного количества работы. На основе стоимости 1-ого спринта, клиент может примерно понять предстоящие траты.

*Изменения – приветствуются*

Как сказано в 1-ом из принципов разработки по Agile, “изменение требований приветствуется”. Подход позволяет сделать изменения и добавить их во время следующей итерации, вне зависимости от стадии проекта.

*Ориентированность на пользователя*

Постоянный фидбэк от пользователей способствует постоянному улучшению продукта. Каждое обновление действительно важно.

**Практик****и Agile**

*Очередь задач*

Часто крупные задачи в проекте необходимо разделять на части. Многие из них скапливаются, образуя очередность. В этом случае менеджеру продукта необходимо тщательно поработать со всеми задачами бэклога, определив верные приоритеты для каждой.

Обычно в бэклог продукта входят следующие элементы: продуктовые фичи, возможные баги, приоритетные знания по продукту, некоторые технические работы, и др.

Все элементы в бэклоге упорядочены согласно их ценности. Чем весомее элемент, тем скорее он пойдет в работу. Верхние позиции будут более подробно описанными и четкими по сравнению с нижними элементами. Все они должны быть понятны для нетехнических членов команды и заинтересованных сторон.

В управлении бэклогом определяющую роль играет собрание backlog grooming, во время которого представители Agile команды обсуждают детали бэклога продукта и готовят очередное планирование спринта.

*Итерации*

Agile-команды выбирают количество работ, которые необходимо выполнить в определенное время. Итеративная разработка означает, что сама команда может решить, что она может сделать, исходя из своих возможностей и опыта предыдущей итерации.

*Клиентоориентированность*

Сотрудничество с клиентами является ключевым моментом в методологии Agile. Согласно гибкому подходу, команда должна предоставить всю информацию, необходимую клиентам, и сообщать им о прогрессе. Постоянная связь также должна быть частью внутренней командной работы.

*User stories*

В Agile описывается функциональность по общению с клиентами, а затем с позиции продукта определенным образом (помните формулу “Я как <тип пользователя>, хочу <действие>, потому что <причина>”?). История пользователей в Agile project management означает единицу работы, которая должна быть завершена в одном спринте.

User stories включают описание, критерии приемки и оценку времени. Когда они слишком сложны, менеджеры продуктов разделяют их на более мелкие.

*Agile-роли*

Методология включает разные роли и, соответственно, разные их названия. Если обобщать, роли в Agile можно разделить по группам, включающим:

* Team Lead, Project Lead и Скрам мастера
* Членов команды
* Собственника продукта для Scrum и On-site customer для XP
* Заинтересованные стороны (stakeholders)

Команды Agile могут также состоять из дополнительно привлекаемых технических специалистов.

*Value stream analysis*

Анализ потока ценностей — это метод управления для анализа текущего состояния и разработки будущего состояния продукта. Цель анализа в том, чтобы идентифицировать и удалять «отходы» в потоках создания ценности, тем самым повышая эффективность потока данных.

Здесь методология знакомит с двумя принципами. Первый — это определение продукта на основе пользовательских историй, основанных на анализе бизнеса. Второй — определение зависимостей между бизнес-и технической функциональностью.

*Timeboxing*

Timeboxing используется в качестве метода планирования проекта. Расписание делится на несколько отдельных периодов времени (таймбоксы), каждый из которых имеет свои конечные результаты, срок и бюджет.

Спринты продолжаются в соответствии с указанными таймфреймами. Обычно от двух недель до одного месяца. Scrum-митинги обычно продолжаются около 15 минут.

*Ежедневные собрания*

Например, Scrum meeting — это ежедневное мероприятие, короткая утренняя или дневная встреча, обычно организованная менеджером продукта или владельцем продукта. Длится 10-15 минут и требует присутствия Скрам-мастера и всей команды. Такая встреча организуется, чтобы:

* вспомнить, что было сделано вчера
* определить, что будет сделано сегодня
* выявить любые препятствия, если такие есть

**Примеры Agile** **методологий**

Существует некоторое количество Agile методологий. Все они имеют схожие характеристики и приемы. Тем не менее, терминология и подходы могут отличаться и подходить/не подходить в определенных ситуациях.

*Scrum*

Скрам – фреймворк, уже давно используемый не только в сфере разработки, но и в инженерной среде. Применение этой системе нашлось даже в строительстве.

В настоящее время, Scrum является одной из наиболее популярных «методологий» разработки ПО. Согласно определению, Scrum — это каркас разработки, с использованием которого люди могут решать появляющиеся проблемы, при этом продуктивно и производя продукты высочайшей значимости.

Авторами Scrum заявлены следующие особенности:

* Легкий (англ. Lightweight)
* Понятный, доступный
* Сложный в освоении

Рекомендуемый размер команды — 7 (плюс-минус 2) человека. Согласно идеологам Scrum, команды большего размера требуют слишком больших ресурсов на коммуникации, в то время как команды меньшего размера повышают риски (за счет возможного отсутствия требуемых навыков) и уменьшают размер работы, который команда может выполнить в единицу времени.

*Kanban*

Организации используют Канбан для того, чтобы управлять процессом работы над проектом. Эта методология создана для более эффективной работы в команде.

Основными принципами Kanban являются:

1. *Визуализация потока задач.*

Канбан опирается на визуализацию. Все задачи записываются на видном месте, чтобы в любой момент знать, как обстоят дела. Запишите все задачи. Чтобы создать визуализацию, нужно записать все задачи, которые вы делаете сейчас и собираетесь сделать в ближайшие дни. После этого станет понятно, сколько у вас реальной работы, а сколько в планах. Определите статусы задач. Статусы задач — это колонки на доске. Колонки можно использовать разные, конкретных правил нет. Для начала мы предлагаем три: «Сделать», «В работе» и «Готово». Потом их можно разбить на более мелкие, если нужно, или придумать новые статусы.

1. *Лимитирование количества одновременной работы.*

Зафиксируйте лимит. Договоритесь с коллегами, сколько задач из каждой колонки сможете делать одновременно. Запишите эти ограничения цифрами над колонками или ограничьте место на доске, чтобы новые карточки не могли на неё поместиться.

Приоритизируйте задачи. После ограничения количества одновременной работы в колонке «Сделать» окажется много карточек. Чтобы их упорядочить, нужна приоритизация. Можно пометить карточки цветом, расположить в определённом порядке или составить рейтинг с баллами. Главное, чтобы все однозначно понимали, какие задачи нужно сделать сейчас, а какие можно отложить на пару дней.

Заканчивайте начатые дела, а не беритесь за несколько новых параллельно

1. *Контроль потока задач.*

Визуализация помогает следить за скоростью продвижения карточек и равномерной загрузкой сотрудников. Когда появляется пробка, действует принцип: один за всех, и все за одного. Сотрудники, которые остались без дела, не отсиживаются в уголке, а помогают разбирать завал. Например, дизайнеры подключаются к тестированию или составлению документов, когда нет своей работы.

*Extreme Programming(XP)*

Экстремальное программирование (XP) – это упрощенная методология организации разработки программ для небольших и средних по размеру команд разработчиков, занимающихся созданием программного продукта в условиях неясных или быстро меняющихся требований. XP сосредоточено на минимизации ошибок на ранних стадиях разработки. Это позволяет добиться максимальной скорости выпуска готового продукта и даёт возможность говорить о прогнозируемости работы.

Принципы ХР:

1. Итеративность - разработка ведется короткими итерациями.
2. Простота решений. Принимается первое простейшее рабочее решение.
3. Интенсивная разработка малыми группами (не больше 10 человек) и парное программирование , активное общение в группе и между группами
4. Обратная связь с заказчиком, представитель которого фактически вовлечен в процесс разработки.

**Выво****д**

Agile – прекрасная система разработки ПО. При подключении к процессу разработки и заказчиков, и программистов, вторым работать становится проще, а первые получают удовлетворительный результат в короткие сроки.

Когда Agile разработка организована правильно, организации могут на протяжении долгого времени постоянно улучшать качество своего продукта, что хорошо и для самой организации, и для пользователей.

**Список использо****ванных источников**

1. Хабрахабр, ресурс для ИТ-специалистов [Электронный ресурс]. – URL: https://habr.com/ru/post/247319/ (дата обращения: 29.10.2018).
2. Что такое Канбан [Электронный ресурс] - https://www.agilebasics.ru/chto-takoe-kanban/ (дата обращения: 04.11.2018).
3. WHAT IS AGILE METHODOLOGY by Luís Gonçalves [Электронный ресурс] - https://luis-goncalves.com/what-is-agile-methodology/ (дата обращения: 05.11.2018).
4. Основополагающие принципы Agile-манифеста [Электронный ресурс] - https://agilemanifesto.org/iso/ru/principles.html (дата обращения: 05.11.2018).
5. WHAT IS SCRUM? [Электронный ресурс] - https://www.scrum.org/resources/what-is-scrum (дата обращения: 07.11.2018)