УДК 621.3.049.77–048.24:537.2

РАЗРАБОТКА ИГР НА ПРИМЕРЕ ИГРЫ «ARCHER ADVENTURES»

Попко И.С., Казак Н.А.

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники» филиал «Минский радиотехнический колледж», г. Минск, Республика Беларусь

Научный руководитель: Гордеюк А. В. – преподаватель высшей категории, магистр

Аннотация. Руководство по организации разработки игры при помощи различных методологий, учитывая этапы продакшна на примере игры Archer Adventures.

Ключевые слова: методологии разработки программного обеспечения, препродакшн, продакшн, постпродакшн

Введение. Разработка компьютерных игр – это невероятно обширное и сложное направление в IT-сфере. Для создания сотен тысяч игр самых различных жанров существуют сотни методологий и техник. Но у каждой из них есть что-то общее, что-то характерное для каждой, без исключения, игры. В этом докладе я попытаюсь погрузиться в этот игровой хаос и рассказать о том, как делаются игры простым языком. Для этого я буду обращаться к своей собственной игре Archer Adventures, на примере которой и будет построено дальнейшее повествование.

Виды методологий. Проанализировав рынок труда и изучив вакансии программистов высокого уровня, я выяснил, какие методологии работодатель требует знать и уметь с ними работать чаще всего. Вот их список:

– «Waterfall Model» (каскадная модель или «водопад»);
– «Incremental Model» (инкрементная модель);
– «Iterative Model» (итеративная или итерационная модель);
– «Agile Model» (гибкая методология разработки);
– «V-Model» (V-образная модель).

Сейчас я расскажу про каждую из них, и мы попытаемся разобраться, в чем их главные отличия, плюсы и минусы [1].

Waterfall Model, или каскадная модель. Это классическая модель, которая использовалась на заре разработки, и продолжает активно применяться и сегодня. Ее принцип работы довольно прост: каждый последующий этап выполняется только тогда, когда полностью закончен предыдущий. Есть четкое разделение этапов, и разработка идет как бы каскадом, постепенно спускаясь от первого этапа к последнему.

Данная модель разработки ПО довольно жесткая, имеет строгие правила. Она четко определяет сроки прохождения каждого этапа. Но есть и недостаток: сделать шаг назад очень сложно. Внесение правок в существующий проект обходится очень дорого и сопряжено с проблемами. Такой способ подходит только для проектов, которые четко расписаны, есть полное понимание, что создается, для каких целей и какие требования выставляются.

Можно использовать такой подход в том случае, если есть подробный прототип или уже существующее подобное приложение. Кроме того, каскадную модель нужно использовать при работе с государственными учреждениями, где важно строго сдавать отчетности, следовать графику и не отклоняться от заданного первоначально плана.

Incremental Model. Инкрементная модель разработки ПО подходит в том случае, если в проекте есть четкий план действий, но продукт нужно запустить быстро, а изменения можно будет вносить позже. Ее суть заключается в том, что сначала прорабатывается набор задач, они распределяются по приоритетности. Далее каждый «блок» разрабатывается по традиционной каскадной модели. Первоначально делается «базовый» продукт с минимальными, но важными функциями. Постепенно он дополняется за счет разработки других компонентов, которые называются инкрементами. Процесс зацикливается, пока не будет полностью собранной единой системы.

Каждая часть представляет собой готовый элемент. Иногда его можно даже использовать отдельно. Как правило, он разрабатывается так, чтобы уже не переделывать его. Именно поэтому и используется каскадная модель внутри инкрементной модели.

Что дает такая методология разработки программного обеспечения? В первую очередь, минимизируются риски, обеспечивается быстрый выход на рынок и запуск продукта. Кроме того, базовый функционал уже будет работать и приносить выгоды для бизнеса, всегда можно внедрить новые сформированные инструменты, если появляется необходимость.

Iterative Model. Такой подход немного по своей конструкции напоминает предыдущий. Суть итеративной методологии разработки программного обеспечения заключается в том, чтобы создать базовый функционал и постепенно его улучшать. Пока звучит похоже с предыдущим вариантом, но есть разница.

Инкрементную модель можно сравнить с пазлом, где все элементы выкладываются поэтапно и постепенно собираются в единую картинку. Итеративная модель – это эскиз художника, который сначала делает набросок карандашом, далее берет в руки краски, а потом прорисовывает детали и создает картину.

Такой метод подходит для крупных проектов, в котором определены основные задачи и есть общее представление, что должно получиться в итоге. Но при этом детали не ясны - не до конца понятно, как будет работать та или иная функция. По такому методу разрабатываются, например, социальные сети, корпоративные платформы. Общий функционал понятен, но количество функций и возможности не определены до конца. Такой подход позволяет постепенно добавлять компоненты и улучшать существующие.

Agile Model. Гибкая методология разработки программного обеспечения – это отличное решение для создания продукта, который не до конца сформирован в своей идее. Особенность данного метода заключается в том, что заказчик может сразу наблюдать за изменениями в разработке и корректировать действия. Это возможно благодаря определению спринтов – отрезков, за которые выполняются задачи. В ходе обсуждения ставятся цели, определяется время отрезка, и разработчики выполняют задачу. Далее проходит обсуждение, вносятся изменения, назначается новый спринт.

Метод гибкой разработки очень эффективен, но имеет свои недостатки. Из-за того, что невозможно определить точные результаты и понять, сколько понадобится времени на реализацию идеи, нельзя точно заранее определить стоимость.

Если проект настроен на длительный жизненный цикл, должен иметь адаптивность к изменениям на рынке, то Agile метод отлично подходит. Он позволяет подстраиваться под требования и вносить правки [2].

Spiral Model. Спиральные модели разработки ПО подходят для крупных проектов, где ошибки сулят большие потери или непредсказуемые последствия. Акцент делается на оценке рисков и проработке конкретных бизнес-задач. Принцип работы, следующий:

Работа идет по спирали, на каждом витке которой осуществляется 4 основных действия: создание плана, анализ рисков, разработка и конструирование, оценка результата и сбор фидбека. Если все 4 этапа успешно пройдены, то разработка переходит на новый виток.

Сам же переход на новый виток напоминает инкрементную модель, где каждый блок разрабатывается независимо и присоединяется к первоначально созданному базовому функционалу. Особенность подхода заключается в том, что большинство времени уходит на анализ и попытку выполнить расчеты и оценить последствия, чем на саму разработку. Программисты иногда гораздо быстрее реализовывают функцию, чем она изучается аналитиками.

Спиральная модель подходит крупным компаниям, так как обходится недешево. Кроме того, разработка занимает больше времени. Этот метод для тех, кто ставит на кон все. Если в случае провала проект перестает существовать, то нужно обезопасить себя и устранить риски. А для этого используется данный метод.

V-Model. Особенность подхода заключается в упоре на тестирование и проверку работоспособности систем в ходе разработки. Тесты проводятся параллельно с самим процессом создания продукта. Сам же принцип наследует базовый подход при каскадной разработке. Процесс идет пошагово, есть четкий план действий, составляется строгое техническое зада-

ние. Но параллельно проводятся тесты, в случае обнаружения ошибок они сразу же исправляются, независимо от этапа разработки.

Особенность этой методологии разработки ПО в том, что уже на ранних стадиях создания проводятся тесты, а к новой стадии можно перейти только тогда, когда устраняются все ошибки. При этом на новой стадии тесты анализируют не только новый этап, но и все предыдущие. Это позволяет контролировать взаимосвязь компонентов и их работоспособность.

При разработке Archer Adventures мы использовали инкрементную модель. Сначала был разработан минимально рабочий продукт, а после к нему добавлялись другие функции, постепенно усложняющие игру и делая её все больше и больше похожей на задуманный изначально проект.

Этапы продакшна. Ещё один момент, свойственный индустрии разработки игр (а также любому фактически конечному продукту), это разбивка проекта на три этапа – препродакшн, продакшн и постпродакшн. Точные определения данных вех в разработке могут меняться в зависимости от проекта, но в общем виде я их представлю так:

– Препродакшн – это период, в который мы работаем над тем, какой должна быть игра, и как мы собираемся её делать.

– Продакшн – период, в который, мы, собственно, делаем игру.

– Постпродакшн – период, в который мы заканчиваем разработку и превращаем проект в нормальный, готовый к доставке пользователю продукт. Данный этап делится на две фазы – альфа (полировка игровых моментов, отработка баланса игровых качеств и процесса) и бета (чистка от багов) [3].

Препродакшн. Препродакшн – наиболее креативный период создания любой игры. Начальные этапы препродакшна на редкость свободны и органичны, в них есть элемент творчества и изобретательства. Более поздние стадии данного периода разработки сфокусированы на решение специфических вопросов и выбора технологий, чтобы последующая разработка была эффективной и правильно размеренной. Во время этого этапа проходит разработка дизайн-документа. Дизайн-документ – это детальное описание разрабатываемой компьютерной игры. Дизайн-документ создается и редактируется командой разработчиков или одним разработчиком и в основном используется для организации работы разработчиков. Документ создается в результате сотрудничества между дизайнерами, художниками и программистами как руководство, которое используется в процессе разработки. Разработчики должны придерживаться дизайн-документа во время процесса формирования игры.

При разработке Archer Adventures в первую очередь мы написали дизайн-документ, который состоит из следующих пунктов:

– Схема игры. Что должен делать игрок, какова конечная цель, что мешает ее достижению.

– Интерфейс. Подробно описанная функциональная часть (что можно делать, каким образом — меню, мышь, горячие клавиши, кнопки…).

– Игровая механика. Как устроен игровой мир, какие характеристики есть у его объектов, формулы движения, боя и всего остального, ролевая система, физика – по вкусу.

– Программные механизмы и алгоритмы. Какими характеристиками будут обладать графический движок, ИИ, сетевой код, интерфейс, редактор карт, звук.

– Графика. Сколько и каких вам понадобится моделей, анимаций, двумерной графики, роликов, обоев (да, и их тоже стоит запланировать заранее). Здесь крайне желательны (может, лучше сказать – необходимы) хоть какие-то наброски, concept art, по которым можно почувствовать визуальный стиль игры.

– Звуки и музыка. Темы, вид и способ отображения звуков, набор звуковых эффектов.

– Игровой мир. Основные персонажи / монстры / виды войск с параметрами и примерным расположением / способом добычи и производства.

– Сотрудники, сроки и план работы.

Вся дальнейшая разработка игры основывалась на этом документе, в нем были описаны все механики, сеттинг, концепт-арты и все прочее, что могло понадобиться в разработке.

Продакшн. Это самая важная часть и та область, где требуется максимальная экспертность, и тратится больше всего денежных средств. Все отделы начинают работать одновременно и занимаются той работой, которая была им определена. У вас есть программисты ин-

струментов, программисты движка, разработчики пользовательского интерфейса и программисты искусственного интеллекта (AI), работающие синхронно друг с другом. Аниматоры и художники работают над визуализацией и анимацией персонажей, объектов, фонов, текстов и дизайном интерфейсов.

Программное обеспечение для разработки должно учитывать все эти моменты, иначе игра имеет все шансы оказаться провальной.

Во время разработки Archer Adventures на этом этапе проходила разработка большей части проекта и практически всех механик, были созданы пользовательский интерфейс и модели персонажей и окружения, настроен игровой баланс. Этот этап занимал 80 % всего времени разработки. На презентации вы видите 3д модели, и скриншоты игры в состоянии, которое получилась после этого этапа. Модели создавались в программе Magica Voxel, пользовательский интерфейс – в Photoshop, а все механики были описаны при помощи языка C#. Затем все это было объединено в игровом движке Unity.

Постпродакшн. Наконец наступает этап тестирования. После того, как игры была разработана, тестеры должны играть в неё и найти все возможные ошибки, которые в ней могли оказаться. Для всех выявленных ошибок разрабатывается решение (bug fix). Затем производится настройка графики и игрового процесса, и вот продукт уже готов. И это ещё не всё. Некоторые неисправности (баги) могут возникнуть только после того, как геймеры начнут использовать игру. Если вдруг выявляются какие-то проблемы, то обязательно должны быть выпущены исправления (патчи), в которых они будут исправлены. На презентации вы видите готовую и последнюю версию игры. До того, как игра стало такой как, вы видите было проделано много работы, выпущено большое количество патчей, в которых все неисправности были исправлены.

Заключение. До того, как игра стало такой, как вы видите было проделано много работы, написано много кода, созданы десятки 3д моделей, выпущено большое количество патчей, в которых все неисправности были исправлены. Таким образом, разработка игр это довольно трудоемкий и время затратный процесс, который состоит из огромного количества шагов и правил, благодаря которым разрабатываются успешные и качественные продукты.

Список литературы

1. *Интерактивные методы обучения усвоения учебного материала учащимися / Ручаевская Е. Г. // Актуальные вопросы профессионального образования = Actual issues of professional education: материалы Ш Международной научно-практической конференции (Республика Беларусь, г. Минск, 1–2 октября 2020 г.) / редкол.: С. Н. Анкуда [и др.]. – Минск: БГУИР, 2020. – С. 257-259. Научное электронное издание ISBN 978-985-543-593-9*

2. *Ручаевская, Е.Г. Основы информационно-коммуникационных технологий: учеб. Пособие для студентов специальности «Профессиональное обучение» высших учебных заведений / Е. Г. Ручаевская, С.И.Иванова. – Мн.: МГВРК, 2007. – 564 с. ISBN 978-985-6851-12-7.*

UDC 621.3.049.77–048.24:537.2

GAME DEVELOPMENT ON THE EXAMPLE OF THE GAME «ARCHER ADVENTURES»

Popko I.S., Kazak N.A.

Educational institution "Belarusian State University of Informatics and Radioelectronics" branch "Minsk Radio Engineering College", Minsk, Republic of Belarus

Scientific supervisor Gordeyuk A.V. – teacher of the highest category, master

Annotation. A guide to organizing game development using various methodologies, considering the stages of production using the example of the game Archer Adventures.

Keywords. software development methodologies, preproduction, production, postproduction.