**Особенности и основные этапы разработки ПО – специфика и принципы работы**



Сегодня в мире тысячи программ и приложений, имеющих разное назначение. Главная задача любого ПО – решение потребности конкретной целевой группы. Однако разработка софта – сложная работа. Для ее выполнения требуется квалификация, опыт, знания со стороны разработчиков. При этом речь не только в сложности кода конкретного языка программирования. Любое веб-приложение или прикладная программа должна быть удобной, практичной, функциональной и интуитивно понятной.

Любая программа должна иметь преимущества, быть реализованной в короткие, оговоренные сроки. Это важно, чтобы конкуренты не смогли перехватить инициативу. Ведь если процесс разработки и тестирования затягивается, момент благоприятного выхода на рынок может быть упущен. Особый момент – функционал. Ведь ПО должно решать проблему целевой аудитории. В таком случае софт будет востребован, станет популярным среди активных пользователей.

**Главные этапы разработки ПО и особенности их прохождения**

Чтобы получить результат, составляют план четких действий. Это имеет особое значение, для чего разработаны разные модели жизненного цикла. Каждая из них применяется при разработке конкретного софта, имеет преимущества. Точный порядок разработки может отличаться, что зависит от потребностей и масштабов.

Важно понимать, что разработка программного обеспечения - это процесс, при помощи которого потребность ЦА удовлетворяется конкретным программным продуктом. Существуют определенные стадии разработки программного обеспечения, которые позволяют создать программу в предельно короткие сроки. Для лучшего понимания, рассмотрим этапы более подробно. Это позволит понять, на чем важно сделать акцент при разработке софта.

**Пять этапов разработки ПО**

Для начала рассмотрим жизненные этапы разработки ПО, которые обозначаются аббревиатурой SDLC (Software development lifecycle). Это определенный цикл, состоящий из 5ти этапов, через которые проходит любая программа при разработке.

**1. Сбор требований и их анализ**

Самый ответственный, важный этап, от которого зависит успех программного обеспечения. Специалисты собирают первичные данные, что позволяет создать основу. Параллельно анализируются риски, связанные с проектом. Это определяет возможность использования разных технических подходов, в основе которых лежит минимизации финансовых расходов.

Специалисты создают макеты и прототипы, определяют требования к проекту. Следующим шагом данного этапа является документирование требований со стороны клиента. Это дает полную правовую защищенность обеих сторон - разработчика и заказчика. В документе прописываются требования, которым должен соответствовать софт.

Чаще всего для этого разрабатывается SRS (Software Requirement Specification) – документ, в котором содержатся основные требования, которые предъявляются к программному продукту. Разработчикам важно точно выявить желание клиента, определить сроки разработки проекта. Здесь главная проблема - многостраничный список требований. Для их решения необходимо тесное взаимодействие с заказчиком, акцент на высокоуровневых требованиях.

**2. Разработка дизайна для программного продукта.**

Создаваемый софт должен быть не только функциональным, но также удобным, понятным для пользователя. Для этого требуется правильно разработать архитектуру, способ представления программы, его пользовательский интерфейс, графическое решение. Особое внимание нужно уделить дизайну, где ориентируются на Software Requirement Specification (сокр. SRS).

SRS – документ, в котором закрепляется перечень требований и свойств, которые предъявляются к правильной, корректной работе программы. Разработчиком и дизайнером нужно понять, в какой форме должен быть представлен продукт. Сделать это непросто. Сам заказчик зачастую не знает этого, полагаясь на опыт, квалификацию программистов.

Для этого обычно каждый из разработчиков предлагает свой подход. После все документируется в DDS (Design Document Specification). Далее информация анализируется, выявляются требования и связи архитектурного модуля продукта с внешними модулями. Чтобы добиться успеха, важно иметь в команде грамотных лидов, способных предложить оптимальную архитектуру на основе опыта выполнения аналогичных проектов.

Рекламная интеграция

* [Парные шрифты - правила подбора](https://blog.skedraw.ru/articles/parnye-shrifty-pravila-podbora)

**3. Непосредственная разработка программного обеспечения**

После того, как разобрались с архитектурой, согласовали функционал, дизайн и концепцию, приступают к разработке софта. Пишут программный код, выполняют сборку, последовательно создают необходимые модули и фичи согласно утвержденному DDS. Практика доказывает, что чем более четкими являются требования в Design Document Specification, тем лучше происходит имплементация.

Рассмотрим трудности, с которыми сталкиваются разработчики на данном этапе.

* Слабая коммуникация между членами команды. Особенно важно согласованность всех действий. Для этого разработчики регулярно должны встречаться, обсуждать работу, предлагать новые идеи, стремиться оптимизировать затраты сил, времени.
* Желание заказчика ускорить разработку ПО. Зачастую клиент не понимает, что разработка программного обеспечения - сложная, многоуровневая задача. Ее нельзя решать в спешке. Ведь в спешке могут быть недоработаны разные аспекты, возникать багги, в результате чего софт не будет удовлетворять заказчика. Здесь руководитель проекта должен настоять на своем. Аргументировать, обосновать сроки.
* Добавление в программу новых фич. В 90% случаев заказчик желает добавить новые функции, не оговоренные изначально в ТЗ. Это приводит к сдвигу сроков сдачи работы. Необходимости выделения дополнительного бюджета. Этот вопрос согласовывают заранее. Это позволит избежать недопонимания.

**4. Тестирование продукта**

После разработки софта, специалисты приступают к тестированию. Этот процесс затрагивает все этапы жизненного цикла. Все баги и недочеты фиксируются, регистрируются и отслеживаются. Недочеты исправляются, программный продукт тестируется заново. Это процесс происходит до тех пор, пока готовый информационный продукт не достигнет тех стандартов качества, которые прописаны в SRS.

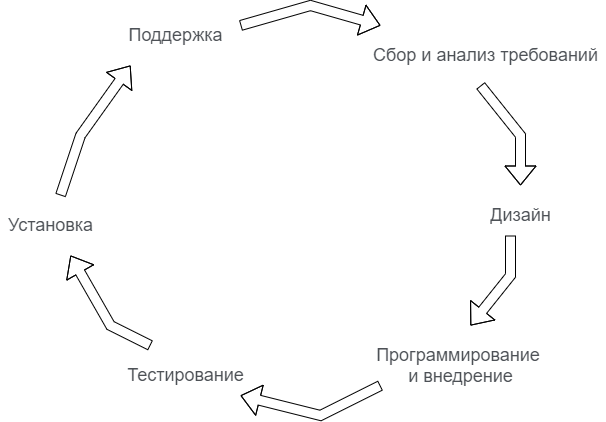
Здесь начинает активно действовать команда автоматизаторов, тестировщиков. Главная сложность этого этапа – время, необходимое на выявление причин багов. Поиск ошибок в коде - сложная задача. Тестирование лучше проводить параллельно с разработкой. Это позволит не возвращаться к ним после запуска ПО.

**5. Внедрение и поддержка продукта**

После устранения всех багов, ПО выходит в релиз. Начинается поэтапное внедрение программы согласно выбранной бизнес-стратегии. Изначально софт может быть выпущен в ограниченном сегменте, протестирован в конкретной бизнес-среде. Для этого выполняют UAT-тестирование (User Acceptance Testing). В его основе - получение реальных отзывов со стороны.

Это позволяет проанализировать обратную связь, увидеть недочеты, произвести улучшение продукта. В дальнейшем, после выпуска продукта на рынок, к работе подключается команда специалистов поддержки (Support команда).

Иногда заказчик предпочитает устанавливать сервера приложений в своей внутренней сети, а не в Google, Azure или AWS. Менеджер должен заранее проинформировать клиента о том, что команда разработчиков не сможет таком случае гарантировать стабильную работу ПО.



Пример жизненного цикла разработки программного обеспечения

Данные этапы разработки ПО выполняются в рамках каскадной («водопад») модели. Она является распространенной, востребованной. Существуют и другие - итерационная, спиральная, гибкая (Agile Model), быстрая (RAD Model), прочие.

Выбор конкретной модели жизненного цикла остается за командой разработчиков. Заказчику важно получить результат. Механизм разработки и внедрения продукта его обычно интересует мало.

**Что будет, если не соблюдать этапы разработки ПО**

Практика доказывает, что далеко не все программы переходят в категорию востребованных. Выходят в лидеры по запросам, популярности у пользователей. Если не выдерживать четкой последовательности действий на этапах разработки программного обеспечения, возрастает риск не достигнуть цели. Иногда готовое приложение остается маловостребованным. Это обуславливается такими причинами:

* Ограниченный функционал.
* Неудобный пользовательский интерфейс.
* Слабые конкурентные преимущества.
* Сложная настройка.
* Повышенные требования к «железу» при инсталляции.

Чтобы не допустить этих ошибок, важно на этапе разработки все тщательно продумать. Обсудить с заказчиком сроки и бюджет, проинформировать о возможности выделения дополнительных денежных средств на создание новых функций. Далее разработчик с командой анализирует задачу, выбирает нужную методологию и модель разработки ПО.

**Основные фазы процесса разработки программного обеспечения**

Программное обеспечение может состоять и одного софта или нескольких модулей, объединенных в единый комплекс. Во втором случае документация с подробной задачей сроками и бюджетом должна присутствовать на каждый из модулей. Рассмотрим подробней фазы, которые проходит программа при разработке:

**1. Постановка задачи.**

Любая работа по созданию начинается с постановки технического задания. В нем указывается название, система программирования, требования к аппаратному обеспечению. В описании присутствует метод обработки данных, используемая математическая модель. ТЗ содержит также входные и выходные сведения, способы диагностирования ошибок, вид выходных данных.

**2. Метод решения задачи.**

Здесь создается логическая или математическая модель. Если задача вычислительного типа, разработчик рассматривает формулы с подробными комментариями. Если задача вычислительного характера, то выполняется словесное описание логической модели. Метод достижения цели в виде разработки ПО должен быть выбран правильно. От этого зависит скорость решения задачи, успех выведения софта на рынок.

**3. Разработка алгоритма.**

Выполняется общая структура ПО. Далее происходит разбивка на определенные части - блоки или программные модули. Для каждого из них формулируются требования по функциям, алгоритмам, который должен быть предельно четким. На выходе получают готовую блок-схему, по которой разработчики могут действовать согласованно.

**4. Программирование и кодирование алгоритмов.**

Задача заключается в переводе математических моделей, которые разработаны для каждого модуля в программу Результат - получение файла с исходным кодом. Для этого используются специальные редакторы.

Рекомендовано, вот специально для вас

* [Скевоморфизм в веб дизайне. Добавляем реальности на сайт](https://blog.skedraw.ru/articles/skevomorfizm-v-veb-dizajne)
* [Футер в веб-дизайне. Создаем дизайн подвала](https://blog.skedraw.ru/articles/futer-v-veb-dizajne)

**5. Компиляция, трансляция программы.**

После кодирования исходный текст вводят в память компьютера, производят компиляцию и тестирование. Для начала исходный текст анализируют и проверяют на наличие синтаксических ошибок. При ее обнаружении цикл проверки автоматически останавливается до устранения.

Процедура продолжается до самого конца, пока весь код не будет тщательно протестирован. Следующий этап - полная компиляция, при которой подключаются функции, модули, библиотеки, прочее. По окончании, если все в порядке, создается файл с расширением EXE.

**6. Тестирование софта.**

Выделяют комплексное и автономное тестирование. При автономной проверке подвергаются отдельные модули, из которых состоит весь комплекс. При комплексом тестировании задействуется все модули ПО. Для корректного тестирования выбираются такие данные, где результат выполнения программы известен заранее. Если обнаружена ошибка, специалисты приступают к откладке программного модуля.

**7. Создание всех необходимой документации.**

Все официальные бумаги классифицируются и разбиваются на группы. Здесь можно выделить такие документы:

Общее описание – документ, в котором описывается характеристика программного продукта, требования к базовому обеспечению, сфера применения, комплекс технических средств обработки данных.

Руководство для пользователя. Здесь описываются возможности, технологические особенности работы ПО для клиентов. Документ обычно встраивается в софт в виде онлайн подсказки. Может оформляться и в печатном виде, по желания заказчика.

Руководство для программиста. Документ предназначен для разработчиков и специалистов, которые занимаются разработкой, поддержкой ПО. Включает в себя задание на разработку, разбиение на программные модули, спецификацию, листинги, планы взаимодействия модулей, прочее.

**8. Эксплуатация программного комплекса, сопровождение.**

После успешного тестирования программа сдается заказчику и запускается в пользование. В ходе эксплуатации разработчики получают обратную связь от пользователей. В результате может возникнуть нужда в добавлении новых модулей, устранении дополнительных ошибок, необходимости расширения функционала. Этим занимаются специалисты из группы Support.

**Выводы**

Современные программы отличаются функционалом, назначением. Каждый софт предназначен для конкретной целевой группы. Чтобы создать ПО, которое будет удовлетворять запрос пользователей, требуется сложная, планомерная работа со стороны разработчиков. Только правильное прохождение всех этапов может привести к достойному результату.