**МОДЕЛИ РАЗРАБОТКИ ПО (ИС)**

Основными моделями разработки программного обеспечения (или ИС) являются:

* каскадная,
* спиральная,
* итеративная.

***Каскадная модель*** проектирования появилась в 70-х годах прошлого столетия. Другое её название – водопад (рис. 1).

Формирование

Требований к ИС

Проектиро-вание

Реализация

Тестирование

Ввод в действие

Эксплуатация и сопровождение

*Рис. 1. Каскадная схема разработки ПО*

Особенностью каскадной схемы является: переход на следующую стадию осуществляется только после того, как будет полностью завершена работа на текущей стадии. Возвратов на пройденные стадии не предусматривается. Каждая стадия завершается получением некоторых результатов. При этом они служат в качестве исходных данных для других стадий. Требования к разрабатываемому ПО (ИС) строго документируются в виде технического задания. Каждая стадия завершается выпуском полного комплекта документации.

Критерием качества разработки при такой схеме является точность выполнения спецификаций технического задания.

*Преимуществами* данной схемы являются:

1. на каждой стадии формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;
2. выполняемые последовательности работ позволяют планировать сроки завершения всех работ и затраты по ним.

*Недостатками* данной схемы являются:

1. реальный процесс проектирования по каскадной модели никогда не укладывается в жесткую схему;
2. процесс создания ПО (ИС) носит, как правило, итерационный характер, а именно, результаты очередной стадии часто вызывают изменения в проектных решениях, выработанных на ранних стадиях;
3. возникает необходимость в возврате к предыдущим стадиям;
4. существенное запаздывание реальных результатов;
5. получение в итоге ПО (ИС) с параметрами существенно отличающимися от ожидаемых.

***V-образная модель*** является усовершенствованием и разновидностью каскадной.

Была предложена именно для того, чтобы устранить недостатки каскадной модели, а название – V-образная, или шарнирная – появилось из-за ее специфического графического представления (рис. 2).



*Рис. 2. V-образная модель*

V-образная модель позволяет гораздо лучше контролировать результат на предмет его соответствия ожиданиям, поскольку сфокусирована на тестировании.

V-образная модель дала возможность значительно повысить качество ПО за счет своей ориентации на тестирование, а также во многом разрешила проблему соответствия созданного продукта выдвигаемым требованиям благодаря процедурам верификации и аттестации на ранних стадиях разработки (пунктирные линии на рисунке указывают на зависимость этапов планирования/постановки задачи и тестирования/приемки).

Однако в целом V-образная модель является всего лишь модификацией каскадной и обладает многими ее недостатками. В частности, и та и другая слабо приспособлены к возможным изменениям требований заказчика. Если процесс разработки занимает продолжительное время (иногда до нескольких лет), то полученный в результате продукт может оказаться фактически ненужным заказчику, поскольку его потребности существенно изменились.

Для преодоления недостатков каскадной и V-образная моделей проектирования в середине 80-х годов была предложена *спиральная модель* ЖЦ разработки ПО (ИС).

Принципиальными положениями спиральной модели ЖЦ являются:

1. прикладное ПО (ИС) создается не сразу, как в случае каскадной схемы, а по частям, с использованием метода прототипирования. Под прототипом понимается программный компонент, реализующий отдельные функции и внешние интерфейсы, разрабатываемого ПО;
2. создание ПО (ИС) осуществляется не сразу, а за несколько витков спирали;
3. на каждом витке производится тщательная проверка риска превышения сроков и стоимости проекта.

Преимуществами спиральной модели является:

1. она избавляет от необходимости полного и точного формулирования требований к системе на начальной стадии проекта;
2. значения параметров системы уточняются на каждом витке спиральной модели;
3. в итоге получается обоснованный вариант, который и является искомым.

1

2

3

4

**Разработка продукта на очередной итерации**

**Планирование**

**Определение целей, альтернатив и ограничений**

**Идентификация и разрешение рисков, оценка альтернатив**

*Рис. 3. Спиральная модель ЖЦ ИС: 1 – начало; 2 – анализ требований; 3 – проектирование; 4 – кодирование и тестирование*

Спиральная модель сфокусирована на проектировании. Собственно разработка ПО происходит лишь на последнем витке спирали по обычной каскадной модели, однако этому предшествует несколько итераций проектирования на основе создания прототипов – при этом каждая итерация включает стадию выявления и анализа рисков и наиболее сложных задач.

Поскольку спиральная модель в основном охватывает именно проектирование, то в первоначальном виде она не получила широкого распространения в качестве метода управления всем жизненным циклом создания ПО. Однако главная ее идея, заключающаяся в том, что процесс работы над проектом может состоять из циклов, проходящих одни и те же этапы, послужила исходным пунктом для дальнейших исследований и стала основой большинства современных моделей процесса разработки ПО.

***Итеративная модель***. Впервые предложена Филиппом Крачтеном в 1995г., данная модель объединяет главные преимущества спиральной, и каскадной моделей, а также методов разработки на основе создания прототипов и объектно-ориентированного подхода (рис. 4). Она завоевала большую популярность и в том или ином виде используется во многих современных проектах.



*Рис. 4. Итеративная модель предлагает использование итераций на всех этапах жизненного цикла.*

В соответствии с итеративной моделью имеются четыре основные фазы жизненного цикла разработки ПО: начало, исследование, построение и внедрение. На каждой фазе проект проходит множество итераций, приводящих к созданию работоспособных версий: на начальных создаются прототипы, уточняются требования, прорабатываются наиболее сложные проблемы; конечные приводят к созданию продукта, его совершенствованию и расширению функциональности.

Итеративная модель, помимо основных фаз, выделяет еще две группы процессов: рабочие (управление требованиями, анализ и проектирование, реализация, тестирование, развертывание) и вспомогательные (управление конфигурацией и изменениями, проектом и процессом). Количество и суть процессов варьируются в зависимости от потребностей разработчика, они также могут иметь свои циклы, которые не обязательно даже соответствуют основным фазам. Однако результатом рабочих процессов всегда является создание версий продукта.

Итеративная модель подобно спиральной дает возможность успешно справляться с рисками. Если во время работы над очередной версией будет установлено, что трудозатраты на реализацию необходимой функциональности слишком велики, то превышения бюджета и нарушения сроков можно будет избежать путем соотнесения приоритетов разработки и трудозатрат в начале каждой итерации. Таким образом, данная модель хорошо подходит для большинства типов программных проектов, но особенно ее преимущества заметны при работе над продуктами, предназначенными для выхода на свободный рынок, в силу изначальной ориентации на выпуск последовательных версий.