**Лекция №3**

**Модели жизненного цикла информационной системы**

**Моделью жизненного цикла** информационной системы будем называть некоторую структуру, определяющую последовательность осуществления процессов, действий и задач, выполняемых на протяжении жизненного цикла информационной системы, а также взаимосвязи между этими процессами, действиями и задачами.  
В стандарте ISO/IEC 12207 не конкретизируются в деталях методы выполнения действий и решения задач, входящих в процессы жизненного цикла информационной системы, а лишь описываются структуры этих процессов. Это вполне понятно, так как регламенты стандарта являются общими для любых моделей жизненного цикла, методологий и технологий разработки. Модель же жизненного цикла зависит от специфики информационной системы и условий, в которых она создается и функционирует.   
К настоящему времени наибольшее распространение получили две основные (классические) модели жизненного цикла:

* каскадная модель, иногда также называемая моделью водопада (waterfall);
* спиральная модель.

**Каскадная модель жизненного цикла информационной системы**

Каскадная модель предусматривает последовательную организацию работ. При этом основной особенностью является разбиение всей разработки на этапы, причем переход с одного этапа на следующий происходит только после того, как полностью завершены все работы на предыдущем этапе. Каждый этап завершается выпуском полного комплекта документации, достаточной для того, чтобы разработка могла быть продолжена другой командой разработчиков.

**Основные этапы разработки по каскадной модели**

Можно выделить следующий ряд устойчивых этапов разработки, практически не зависящих от предметной области:

* анализ требований заказчика;
* проектирование;
* разработка;
* тестирование и опытная эксплуатация;
* сдача готового продукта.

Результатом, получаемым на первом этапе, является техническое задание (задание на разработку), согласованное со всеми заинтересованными сторонами.  
Результатом второго этапа является комплект проектной документации, содержащей все необходимые данные для реализации проекта.  
Результатом выполнения третьего этапа является готовый программный продукт.  
Результатом выполнения четвертого этапа являются различного рода скрытые недостатки, про­являющиеся в реальных условиях работы информационной системы.  
Главная задача пятого этапа — убедить заказчика, что все его требования выполнены в полной мере.  
В действительности жизненный цикл самой системы существенно сложнее и длиннее. Он может включать в себя произвольное число циклов уточнения, изменения и дополнения уже принятых и реализованных проектных решений. В этих циклах происходит развитие информационной системы и модернизация отдельных ее компонентов.

**Основные достоинства каскадной модели**

Рассмотрим ее основные достоинства.

* На каждом этапе формируется законченный набор проектной документации, отвечающей критериям полноты и согласованности. На заключительных эта­пах также разрабатывается пользовательская документация, охватывающая все предусмотренные стандартами виды обеспечения информационной системы (организационное, методическое, информационное, программное, аппаратное).
* Выполняемые в логичной последовательности этапы работ позволяют плани­ровать сроки завершения и соответствующие затраты.

Каскадный подход хорошо зарекомендовал себя при разработке определенных информационных систем. Имеются в виду системы, для которых в самом начале разработки можно достаточно точно и полно сформулировать все требования, с тем, чтобы предоставить разработчикам свободу выбора реализации, наилучшей с технической точки зрения. К таким информационным системам, в частности, относятся сложные расчетные системы, системы реального времени.

**Недостатки каскадной модели**

Перечень недостатков каскадной модели при ее использовании для разработки информационных систем достаточно обширен:

* существенная задержка в получении результатов;
* ошибки и недоработки на любом из этапов проявляются, как правило, на последующих этапах работ, что приводит к необходимости возврата назад;
* сложность параллельного ведения работ по проекту;
* чрезмерная информационная перенасыщенность каждого из этапов;
* сложность управления проектом;
* высокий уровень риска и ненадежность инвестиций.

**Задержка в получении результатов** обычно считается главным недостатком каскадной схемы. Данный недостаток проявляется в основном в том, что из-за последовательного подхода к разработке согласование результатов с заинтересован­ными сторонами производится только после завершения очередного этапа работ.  
Используемые при разработке информационной системы модели автоматизируемого объекта, отвечающие критериям внутренней согласованности и полноты, могут в силу различных причин устареть за время разработки (например, из-за внесения изменений в законодательство, колебания курса валют и т.п.). Это относится и к функциональной модели, и к информационной модели, и к проектам интерфейса пользователя, и к пользовательской документации.  
**Возврат на более ранние стадии**. Данный недостаток каскадной модели, в общем-то, является одним из проявлений предыдущего. Возврат может служить причиной срыва графика работ и усложнения взаимоотношений между группами разработчиков, выполняющих отдельные этапы работы.  
Самым же неприятным является то, что недоработки предыдущего этапа могут обнаруживаться не сразу на последующем этапе, а позднее (например, на стадии опытной эксплуатации могут проявиться ошибки в описании предметной области). Это означает, что часть проекта должна быть возвращена на начальный этап работы. Вообще, работа может быть возвращена с любого этапа на любой предыдущий этап.  
**Сложность параллельного ведения работ**. Проблемы возникают вследствие того, что работа над проектом строится в виде цепочки последовательных шагов. Причем даже в том случае, когда разработку некоторых частей проекта (подсистем) можно вести параллельно, при использовании каскадной схемы распараллеливание работ весьма затруднительно. Сложности параллельного ведения работ связаны с необходимостью постоянного согласования различных частей проекта. Чем сильнее взаимозависимость отдельных частей проекта, тем чаще и тщательнее должна выполняться синхронизация, тем сильнее зависят друг от друга группы разработчиков.  
**Информационная перенасыщенность**. Проблема информационной перенасыщенности возникает вследствие сильной зависимости между различными группами разработчиков. Данная проблема заключается в том, что при внесении изменений в одну из частей проекта необходимо оповещать всех разработчиков, которые ис­пользовали или могли бы использовать эту часть в своей работе. Как следствие, объем документации по мере разработки проекта растет очень быстро, так что требуется все больше времени для составления документации и ознакомления с ней.  
Следует также отметить, что, помимо изучения нового материала, не отпадает необходимость в изучении старой информации. Это связано с тем, что вполне вероятна ситуация, когда в процессе разработки изменяется состав группы разработчиков (этот процесс носит название **ротации кадров**).  
**Сложность управления проектом** при использовании каскадной схемы в основном обусловлена строгой последовательностью стадий разработки и наличием сложных взаимосвязей между различными частями проекта.  
Последовательность разработки проекта приводит к тому, что одни группы разработчиков должны ожидать результатов работы других команд. Поэтому требуется административное вмешательство для согласования сроков работы и состава передаваемой документации.  
**Высокий уровень риска**. Чем сложнее проект, тем больше продолжительность каждого из этапов разработки и тем сложнее взаимосвязи между отдельными частями проекта, количество которых также увеличивается. Возврат на предыдущие стадии может быть связан не только с ошибками, но и с изменениями, произошедшими в предметной области или в требованиях заказчика за время разработки. Причем возврат проекта на доработку вследствие этих причин не гарантирует, что предметная область снова не изменится к тому моменту, когда будет готова следующая версия проекта. Фак­тически это означает, что существует вероятность того, что процесс разработки «зациклится» и система никогда не дойдет до сдачи в эксплуатацию.   
Поэтому можно утверждать, что сложные проекты, разрабатываемые по каскадной схеме, имеют повышенный уровень риска.

**Спиральная модель жизненного цикла**

Спиральная модель, в отличие от каскадной, предполагает итерационный процесс разработки информационной системы. При этом возрастает значение начальных этапов жизненного цикла, таких как анализ и проектирование. На этих этапах проверяется и обосновывается реализуемость технических решений путем создания прототипов.

**Итерации**

Каждая итерация представляет собой законченный цикл разработки, приводящий к выпуску внутренней или внешней версии изделия (или подмножества конечного продукта), которое совершенствуется от итерации к итерации, чтобы стать закопченной системой (рис. 3).  
Таким образом, каждый виток спирали соответствует созданию фрагмента или версии программного изделия, на нем уточняются цели и характеристики проекта, определяется его качество, планируются работы на следующем витке спирали. На каждой итерации углубляются и последовательно конкретизируются детали проекта, в результате чего выбирается обоснованный вариант, который доводится до окончательной реализации.  
Использование спиральной модели позволяет осуществлять переход на следующий этап выполнения проекта, не дожидаясь полного завершения текущего — недоделанную работу можно будет выполнить на следующей итерации. Главная за­дача каждой итерации — как можно быстрее создать работоспособный продукт, который можно показать пользователям системы. Таким образом, существенно упрощается процесс внесения уточнений и дополнений в проект.  
Рис. 3. Спиральная модель жизненного цикла информационной системы.

**Преимущества спиральной модели**

Спиральный подход к разработке программного обеспечения позволяет преодолеть большинство недостатков каскадной модели и, кроме того, обеспечивает ряд дополнительных возможностей, делая процесс разработки более гибким.  
Рассмотрим преимущества итерационного подхода более подробно.

* Итерационная разработка существенно упрощает внесение изменений в проект при изменении требований заказчика.
* При использовании спиральной модели отдельные элементы информацион­ной системы интегрируются в единое целое постепенно. При итерационном подходе интеграция производится фактически непрерывно. Поскольку интеграция начинается с меньшего количества элементов, то возникает гораздо меньше проблем при ее проведении.
* Уменьшение уровня рисков. Данное преимущество является следствием предыдущего, так как риски обнаруживаются именно во время интеграции. Поэтому уровень рисков максимален в начале разработки проекта. По мере продвижения разработки ожидаемый уровень рисков снижается. На рис. 4 приведены в сравнении графики зависимости уровня рисков от времени разработки для каскадного и спирального подходов.

Рис. 4. Зависимость рисков от времени разработки.

* Итерационная разработка обеспечивает большую гибкость в управлении про­ектом, давая возможность внесения тактических изменений в разрабатыва­емое изделие.
* Итерационный подход упрощает повторное использование компонентов. Это обусловлено тем, что гораздо про­ще выявить общие части проекта, когда они уже частично разработаны, чем пытаться выделить их в самом начале проекта. Анализ проекта после проведения нескольких начальных итераций позволяет выявить общие многократно используемые компоненты, которые на последующих итера­циях будут совершенствоваться.
* Спиральная модель позволяет получить более надежную и устойчивую систе­му. Это связано с тем, что по мере развития системы ошибки и слабые места обнаруживаются и исправляются на каждой итерации.
* Итерационный подход дает возможность совершенствовать процесс разработки — анализ, проводимый в конце каждой итерации, позволяет проводить оцен­ку того, что должно быть изменено в организации разработки, и улучшить ее на следующей итерации.

**Недостатки спиральной модели**

Основная проблема спирального цикла — определение момента перехода на следующий этап. Для ее решения необходимо ввести временные ограничения на каждый из этапов жизненного цикла. Иначе процесс разработки может превратиться в бесконечное совершенствование уже сделанного. При итерационном подходе полезно следовать принципу «лучшее — враг хорошего». Поэтому завершение итерации должно производиться строго в соответствии с планом, даже если не вся запланированная работа закончена.