

# Жизненный цикл


Определение, этапы,  
модели

Software Life Cycle Model

# Определение



Жизненный цикл программного обеспечения (Software Life Cycle Model) — это период времени, который начинается с момента принятия решения о создании программного продукта и заканчивается в момент его полного изъятия из эксплуатации. Этот цикл — процесс построения и развития ПО.





## Этапы жизненного цикла

**В жизненном цикле разработки ПО  
можно выделить 6 основных этапов**

- Анализ, составление требований к продукту
- Проектирование и дизайн
- Реализация
- Тестирование
- Релиз
- Поддержка и обслуживание

# Этапы жизненного цикла



# Модели Жизненного цикла

Каскадная модель – модель процесса разработки программного обеспечения, жизненный цикл которой выглядит как поток, последовательно проходящий фазы анализа требований, проектирования, реализации, тестирования, интеграции и поддержки.

## Достоинства модели

- стабильность требований в течение всего жизненного цикла разработки;
- на каждой стадии формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;
- определенность и понятность шагов модели и простота её применения;
- выполняемые в логической последовательности этапы работ позволяют планировать сроки завершения всех работ и соответствующие ресурсы (денежные, материальные и людские).

## Недостатки модели

- сложность чёткого формулирования требований и невозможность их динамического изменения на протяжении пока идет полный жизненный цикл;
- низкая гибкость в управлении проектом;
- последовательность линейной структуры процесса разработки, в результате возврат к предыдущим шагам для решения возникающих проблем приводит к увеличению затрат и нарушению графика работ;
- непригодность промежуточного продукта для использования;
- невозможность гибкого моделирования уникальных систем;

# Модели Жизненного цикла

**Инкрементная модель** - подразумевает разработку программного обеспечения с линейной последовательностью стадий, но в несколько инкрементов, т.е. с запланированным улучшением продукта за все время пока Жизненный цикл разработки ПО не подойдет к окончанию.

## Достоинства модели

- затраты, которые получаются в связи с изменением требований пользователей, уменьшаются, повторный анализ и совокупность документации значительно сокращаются по сравнению с каскадной моделью;
- легче получить отзывы от клиента о проделанной работе — клиенты могут озвучить свои комментарии в отношении готовых частей и могут видеть, что уже сделано. Т.к. первые части системы являются прототипом системы в целом.
- у клиента есть возможность быстро получить и освоить программное обеспечение — клиенты могут получить реальные преимущества от системы раньше, чем это было бы возможно с каскадной моделью.

## Недостатки модели

- менеджеры должны постоянно измерять прогресс процесса. в случае быстрой разработки не стоит создавать документы для каждого минимального изменения версии;
- структура системы имеет тенденцию к ухудшению при добавлении новых компонентов — постоянные изменения нарушают структуру системы. Чтобы избежать этого требуется дополнительное время и деньги на рефакторинг. Плохая структура делает программное обеспечение сложным и дорогостоящим для последующих изменений. А прерванный Жизненный цикл ПО приводит еще к большим потерям.

# Модели Жизненного цикла

**Спиральная модель:** Жизненный цикл — на каждом витке спирали выполняется создание очередной версии продукта, уточняются требования проекта, определяется его качество и планируются работы следующего витка.

## Достоинства модели

- позволяет быстрее показать пользователям системы работоспособный продукт, тем самым, активизируя процесс уточнения и дополнения требований;
- допускает изменение требований при разработке программного обеспечения, что характерно для большинства разработок, в том числе и типовых;
- в модели предусмотрена возможность гибкого проектирования, поскольку в ней воплощены преимущества каскадной модели, и в то же время разрешены итерации по всем фазам этой же модели;
- позволяет получить более надежную и устойчивую систему. По мере развития программного обеспечения ошибки и слабые места обнаруживаются и исправляются на каждой итерации;

## Недостатки модели

- если проект имеет низкую степень риска или небольшие размеры, модель может оказаться дорогостоящей. Оценка рисков после прохождения каждой спирали связана с большими затратами;
- Жизненный цикл модели имеет усложненную структуру, поэтому может быть затруднено её применение разработчиками, менеджерами и заказчиками;
- спираль может продолжаться до бесконечности, поскольку каждая ответная реакция заказчика на созданную версию может порождать новый цикл, что отдалает окончание работы над проектом;

Спасибо за внимание=)

---