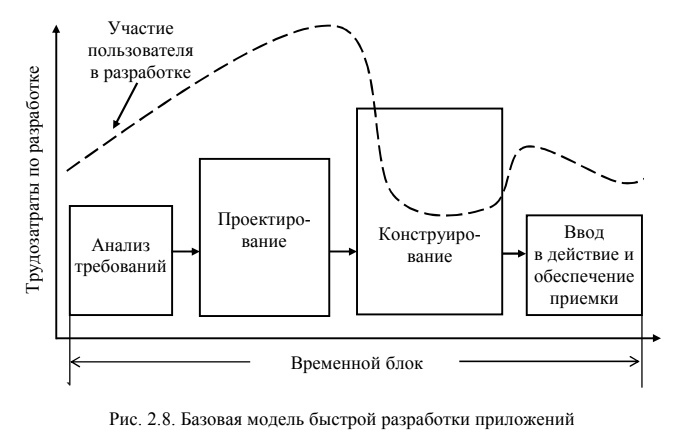
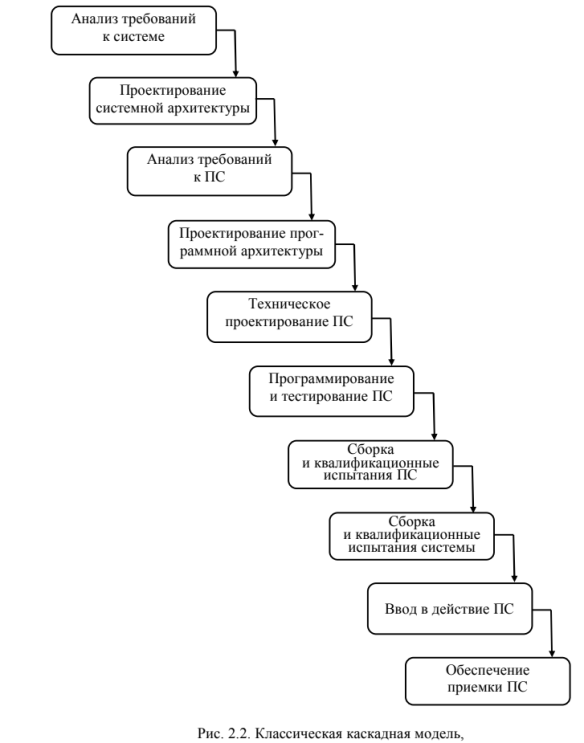
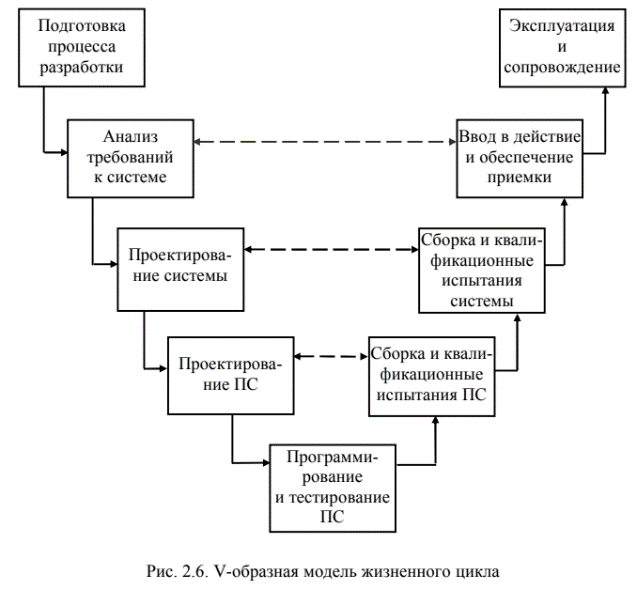
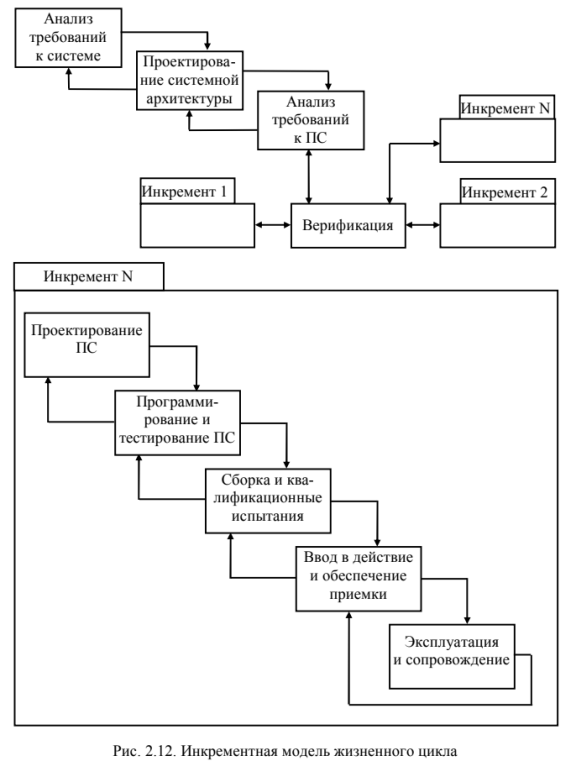
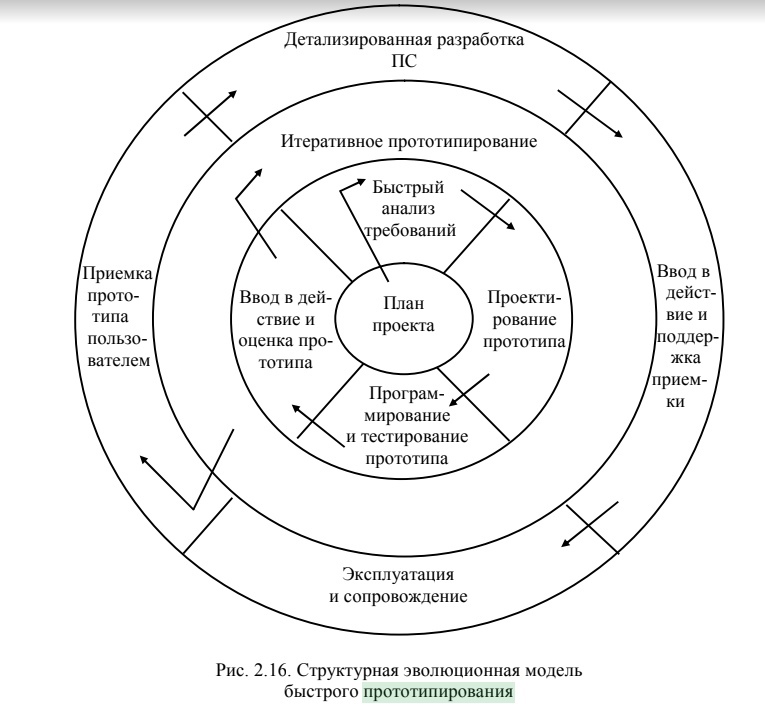
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Характеристика | Жизненный цикл | Недостатки и достоинства | Для каких проектов |
| 1.Каскадная | Однократный проход этапов  разработки. Данная стратегия основана на полном определении всех требований к разрабатываемому программному средству или системе в начале процесса разработки. Каждый этап разработки начинается после завершения предыдущего этапа. Возврат к уже выполненным этапам не предусматривается. |  | **Достоинства** 1) стабильность требований в течение ЖЦ разработки;  2)необходимость только одного прохода этапов разработки, что обеспечивает простоту применения стратегии;  3) простота планирования, контроля и управления проектом;  4) доступность для понимания заказчиками.  **Недостатки** 1) сложность полного формулирования требований в начале процесса  разработки и невозможность их динамического изменения на протяжении ЖЦ  2) разрабатываемые ПС или системы обычно слишком велики и сложны, чтобы все работы по их созданию выполнять однократно в результате возврат к предыдущим шагам для решения возникающих проблем приводит к увеличению финансовых затрат и нарушению графика работ  3) непригодность промежуточных продуктов для использования  4) недостаточное участие пользователя в процессе разработки ПС | 1) при разработке проектов с четкими, неизменяемыми в течение ЖЦ  требованиями и понятной реализацией;  2) при разработке проектов невысокой сложности  3) при выполнении больших проектов в качестве составной части моде-  лей ЖЦ |
| 2.**Инкрементная** | Многократный проход этапов разработки с запланированным улучшением результата.  Данная стратегия основана на полном определении всех требований к  разрабатываемому программному средству (системе) в начале процесса разработки. Однако полный набор требований реализуется постепенно в соответствии с планом в последовательных циклах разработки. |  | **Достоинства** 1) возможность получения функционального продукта после реализации  каждого инкремента;  2) короткая продолжительность создания инкремента; это приводит к сокращению сроков начальной поставки, позволяет снизить затраты на поставки программного продукта;  3)стабильность требований во время создания определенного инкремента; возможность учета изменившихся требований;  4) снижение рисков по сравнению с каскадной стратегией; **Недостатки** 1) необходимость полного функционального определения системы или программного средства в начале ЖЦ для обеспечения планирования инкрементов и управления проектом;  2) возможность текущего изменения требований к системе или программному средству, которые уже реализованы в предыдущих инкрементах;  3) сложность планирования и распределения работ;  4) проявление человеческого фактора, связанного с тенденцией к оттягиванию решения трудных проблем на поздние инкременты, что может нарушить график работ или снизить качество программного продукта. | 1) при разработке проектов, в которых большинство требований можно  сформулировать заранее, но часть из них могут быть уточнены через определенный период времени;  2) при разработке сложных проектов с заранее сформулированными требованиями; для них разработка системы или программного средства за один  цикл связана с большими трудностями;  3) при необходимости быстро поставить на рынок продукт, имеющий базовые функциональные свойства;  4) при разработке проектов с низкой или средней степенью рисков;  5) при выполнении проекта с применением новых технологий. |
| 3.V-образная модель. | Модель представляет собой разновидность каскадной модели.  Данная модель поддерживает каскадную стратегию однократного выполнения  этапов процесса разработки ПС или систем и базируется на предварительном  полном формировании требований. В классической V-образной модели каждый шаг начинается после завершения предыдущего шага.  Отличием V образной модели от каскадной является то, что в ней выделены связи между шагами, предшествующими программированию, и соответствующими видами тестирования и испытаний. |  | **Достоинства** 1) планирование тестирования и испытаний на ранних стадиях разработ-  ки системы и программного средства;  2) упрощение аттестации и верификации промежуточных результатов  разработки;  3) упрощение управления и контроля хода процесса разработки. **Недостатки** 1) поздние сроки тестирования требований в жизненном цикле, что оказывает существенное влияние на график выполнения проекта при необходимости изменения требований;  2) отсутствие, как и в остальных каскадных моделях, действий, направленных на анализ рисков. | 1)Если требуется тщательное тестирование продукта, то V-модель оправдает заложенную в себя идею: validation and verification. 2)Для малых и средних проектов, где требования четко определены и фиксированы. 3)В условиях доступности инженеров необходимой квалификации, особенно тестировщиков. |
| 4.Эволюционная | представляет собой многократный проход этапов разработки. Данная стратегия основана на частичном определении требований к разрабатываемому программному средству или системе в начале процесса разработки. Требования постепенно уточняются в последовательных  циклах разработки. |  | **Достоинства** 1) возможность уточнения и внесения новых требований в процессе раз-  работки;  2) пригодность промежуточного продукта для использования;  3) возможность управления рисками;  4) обеспечение широкого участия пользователя в проекте, начиная с ранних этапов, что минимизирует возможность разногласий между заказчиками и  разработчиками  5) реализация преимуществ каскадной и инкрементной стратегий. **Недостатки** 1) неизвестность точного количества необходимых итераций и сложность определения критериев для продолжения процесса разработки на следующей итерации; это может вызвать задержку реализации конечной версии системы.  2) сложность планирования и управления проектом;  3) необходимость активного участия пользователей в проекте, что реально не всегда осуществимо;  4) необходимость в мощных инструментальных средствах и методах  прототипирования;  5) возможность отодвигания решения трудных проблем на последующие  циклы, что может привести к несоответствию полученных продуктов требованиям заказчиков. | 1) при разработке проектов, для которых требования слишком сложны,  неизвестны заранее, непостоянны или требуют уточнения;  2) при разработке сложных проектов, в том числе:  1.больших долгосрочных проектов;  2.проектов по созданию новых, не имеющих аналогов ПС или систем;  3.проектов со средней и высокой степенью рисков;  4.проектов, для которых нужна проверка концепции, демонстрация  технической осуществимости или промежуточных продуктов;  3) при разработке проектов, использующих новые технологии. |
| 5. Модель быстрого прототипирования | Используется, когда умышленно разрабатываются неполные требования. На их основе выполняются укрупнённое проектирование программирование и тестирование программных элементов, тестирование системы в целом. Таким образом реализуется построение исходного прототипа. |  | **Недостатки:** вероятность недостаточного качества результирующего ПС и системы в целом. Возможна задержка реализации конечного ПС. **Достоинства:** модель быстрого прототипирования является ускорением разработки ПС и системы. | Используется для ускорения разработки системы и ПС |
| 6. RAD-модель | Четкое время перехода от анализа требований до создания полной системы или программного средства. Разработка прототипа часто ограничивается определённым периодом времени (временным блоком; 60 дней) |  | **Недостатки:**  1) необходимость в постоянном участии пользователя в процессе разработки. 2) необходимость в высококвалифицированных разработчиках; 3) возможность применения только для систем или ПС, для которых отсутствует требование высокой производительности; 4) жесткость временных ограничений на разработку прототипа; 5) сложность ограничения затрат и определения сроков завершения работы; 6) неприменимость в условиях высоких технических рисков, при использовании новых технологий.  **Достоинства:** сокращение продолжительности цикла разработки и всего проекта. Сокращение риска несоблюдения графика за счет использования принципа временного блока и связанная с этим упрощения планирования. Сокращения риска связанного с неудовлетворённостью заказчика/пользователя. Возможность повторного использования существующих компонентов.  6. ЖЦ RAD-модели | Данный тип разработки подходит для проектов в котором постоянно поддерживается коммуникация с заказчиком и/или пользователем |

3. ЖЦ V-образной модели

2. ЖЦ Инкрементной модели

1. Каскадная модель

5. ЖЦ Эволюционной модели быстрого прототипирования

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории требований | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли требования к проекту легко определимыми и реализуемыми? | Да | Да | Да | Нет | Нет | Нет |
| 2. | Могут ли требования быть сформулированы в начале ЖЦ? | Да | Да | Да | Да | Нет | Нет |
| 3. | Часто ли будут изменяться требования на протяжении ЖЦ? | Нет | Нет | Нет | Нет | Да | Да |

Продолжение таблицы 3.1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 4. | Нужно ли демонстрировать требования с целью их определения? | Нет | Нет | Да | Нет | Да | Да |
| 5. | Требуется ли проверка концепции программного средства или системы? | Нет | Нет | Да | Нет | Да | Да |
| 6. | Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Нужно ли реализовать основные требования на ранних этапах разработки? | Нет | Нет | Да | Да | Да | Да |

Каскадная – 5, V-образная – 5, RAD – 4, инкрементная – 6, быстрого прототипирования – 2, эволюционная – 2.

На основе результатов заполнения табл. 3.1 наиболее подходящей является инкрементная модель.

Таблица 3.2 – Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории команды разработчиков  проекта | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли проблемы предметной области проекта новыми для большинства разработчиков? | Нет | Нет | Нет | Нет | Да | Да |
| 2. | Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков? | Да | Да | Нет | Нет | Нет | Да |
| 3. | Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Да | Да | Нет | Да | Нет | Нет |
| 5. | Важна ли легкость распределения человеческих ресурсов проекта? | Да | Да | Да | Да | Нет | Нет |
| 6. | Приемлет ли команда разработчиков оценки, проверки, стадии разработки? | Да | Да | Нет | Да | Да | Да |
|  | | | | | | | | |

Каскадная – 4, V-образная – 5, RAD – 4, инкрементная – 5, быстрого прототипирования – 2, эволюционная – 1.

На основе результатов заполнения табл. 3.2 наиболее подходящими являются RAD и инкрементная модели.

Таблица 3.3 – Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории коллектива пользователей | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да | Да | Нет | Да | Нет | Да |
| 2. | Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки? | Нет | Нет | Нет | Да | Да | Да |

Продолжение таблицы 3.3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 3. | Будут ли пользователи вовлечены во все фазы ЖЦ разработки? | Нет | Нет | Да | Нет | Да | Нет |
| 4. | Будет ли заказчик отслеживать ход выполнения проекта? | Нет | Нет | Нет | Нет | Да | Да |

Каскадная – 3, V-образная – 3, RAD – 1, инкрементная – 1, быстрого прототипирования – 1, эволюционная – 3.

На основе результатов заполнения табл. 3.3 наиболее подходящими являются каскадная и эволюционная модели.

Таблица 3.4 – Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории типов проекта и рисков | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Разрабатывается ли в проекте продукт нового для организации направления? | Нет | Нет | Нет | Да | Да | Да |
| 2. | Будет ли проект являться расширением существующей системы? | Да | Да | Да | Да | Нет | Нет |
| 3. | Будет ли проект крупно- или среднемасштабным? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Ожидается ли длительная эксплуатация продукта? | Да | Да | Нет | Да | Нет | Да |
| 5. | Необходим ли высокий уровень надежности продукта проекта? | Нет | Да | Нет | Да | Нет | Да |

Продолжение таблицы 3.4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 6. | Предполагается ли эволюция продукта проекта в течение ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Велика ли вероятность изменения системы (продукта) на этапе сопровождения? | Нет | Нет | Нет | Да | Да | Да |
| 8. | Является ли график сжатым? | Нет | Нет | Да | Да | Да | Да |
| 9. | Предполагается ли повторное использование компонентов? | Нет | Нет | Да | Да | Да | Да |
| 10. | Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)? | Нет | Нет | Нет | Нет | Да | Да |

Каскадная – 5, V-образная – 7, RAD – 3, инкрементная – 5, быстрого прототипирования – 3, эволюционная – 5.

На основе результатов заполнения табл. 3.4 наиболее подходящей является V-образная модель.

Исходя из результатов заполнения табл. 3.1 – 3.4 (каскадная – 16, V-образная – 19, RAD – 10, инкрементная – 17, быстрого прототипирования – 9, эволюционная – 12) наиболее подходящей моделью, для разработки данного ПП, является V-образная модель.