|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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.Спиральная | Данная модель жизненного цикла характерна при разработке новаторских (нетиповых) систем. В начале работы над проектом у заказчика и разработчика нет четкого видения итогового продукта (требования не могут быть четко определены) или стопроцентной уверенности в успешной реализации проекта (риски очень велики). В связи с этим принимается решение разработки системы по частям с возможностью изменения требований или отказа от ее дальнейшего развития. Как видно из рис.3.4, развитие проекта может быть завершено не только после стадии внедрения, но и после стадии анализа риска |  | **Достоинства** 1)позволяет быстрее показать пользователям системы работоспособный продукт, тем самым, активизируя процесс уточнения и дополнения требований; 2) обеспечивает большую гибкость в управлении проектом; 3) уменьшаются риски заказчика. Заказчик может с минимальными для себя финансовыми потерями завершить развитие неперспективного проекта.  **Недостатки** 1) увеличивается неопределенность у разработчика в перспективах развития проекта. Этот недостаток вытекает из предыдущего достоинства модели;  2) затруднены операции временного и ресурсного планирования всего проекта в целом. Для решения этой проблемы необходимо ввести временные ограничения на каждую из стадий жизненного цикла. Переход осуществляется в соответствии с планом, даже если не вся запланированная работа выполнена. План составляется на основе статистических данных, полученных в предыдущих проектах и личного опыта разработчиков. | 1) при разработке новаторских (нетиповых) систем. 2) 3) |
| 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.RAD-модель | представляет собой инкрементную модель, в которой множество разработок маленьких - маленьких кусков выбираются и развиваются одновременно для достижения большей картины. Кроме того, обрабатывается инкрементная модель, в которой основные характеристики, подлежащие разработке, делятся на более мелкие, выполнимые куски. |  | **Достоинства 1)** в модели используются современные методы моделирования данных; 2) + достоинства структурной эволюционной модели быстрого прототипирования. 3) требуется меньшее количество специалистов, т.к. специалисты хорошо владеют предметной областью 4) благодаря использованию мощных инструментальных средств время цикла разработки для всего проекта можно сократить;  **Недостатки** 1)Для больших проектов в RAD требуются существенные людские ресурсы (необходимо создать достаточное количество групп).2)RAD применима только для таких приложений, которые могут декомпозироваться на отдельные модули и в которых производительность не является критической величиной.3)RAD не применима в условиях высоких технических рисков (то есть при использовании новой технологии). | 1) в системах, которые поддаются моделированию (основанных на использовании компонентных объектов), а также в масштабируемых системах; 2) в системах, требования для которых в достаточной мере хорошо известны; 3) при выполнении проектов, разработка которых должна быть выполнена в сокращенные сроки; 4) когда затраты и соблюдение графика не являются самым важным вопросом; |
| 5. многопроходная | Представляет собой несколько итераций процесса построения прототипа программного продукта с добавлением на каждой следующей итерации новых функциональных возможностей или повышением эффективности программного продукта. |  | **Достоинства**  1) в начале разработки требуются средства только для разработки и реализации основных функций программного продукта; 2) после каждого инкремента получается функциональный продукт; 3) снижается риск неудачи и изменения требований; 4) улучшается понимание как разработчиками, так и пользователями программного продукта требований для более поздних итераций;  **Недостатки**  1) не предусмотрены итерации внутри каждого инкремента; 2) определение полной функциональности должно быть осуществлено в самом начале жизненного цикла разработки; 3) может возникнуть тенденция оттягивания решения трудных задач;4) общие затраты на создание программного продукта не будут снижены по сравнению с другими моделями; обязательным условием является наличие хорошего планирования и проектирования. | Многопроходная модель может быть применена, если большинство требований к программному продукту будут сформулированы заранее, а для выполнения проекта будет выделен большой период времени. |
| 6. прототипирования | Модель прототипитования позволяет создать прототип программного продукта до или в течение этапа составления требований к программному продукту. |  | Достоинства 1. Взаимодействие заказчика с разрабатываемой системой начинается на раннем этапе;2. Благодаря реакции заказчика на прототип сводится к минимуму число неточностей в требованиях;3. Снижается вероятность возникновения путаницы, искажения информации или недоразумений при определении требований к программному прдукту, что приводит к созданию более качественного программного продукта;4. В процессе разработки всегда можно учесть новые, даже неожиданные требования заказчика;5. Прототип представляет собой формальную спецификацию, воплощенную в программный продукт;  Недостатки  1. Решение сложных задач может отодвигаться на будущее;2. Заказчик может предпочесть получить прототип, а не законченную полную версию программного продукта;3. Прототипирование может неоправданно затянуться;4. Перед началом работы неизвестно, сколько итераций придется выполнить. |  |

2. Научиться осуществлять выбор стратегии и модели жизненного цикла для разработки конкретного проекта и обосновывать свой выбор. Процедура выбора модели жизненного цикла программных средств и систем

Для выбора подходящей к условиям конкретного проекта модели ЖЦ ПС и систем Институтом качества программного обеспечения SQI рекомендуется использовать специальную процедуру. Данная процедура базируется на применении четырех таблиц вопросов.

Каждая из табл. 3.1 – 3.4 представляет одну из категорий классификации проектов. Каждый из вопросов (строка в таблице) предназначен для классификации анализируемого проекта по определенному критерию категории. Столбцы данных таблиц соответствуют обобщенным моделям ЖЦ, фактически представляющим стратегии разработки ПС. При этом под RAD-моделью подразумевается независимая RAD-модель, не встроенная в другие модели жизненного цикла.

Рассматриваемая процедура состоит из следующей последовательности шагов:

*1-й шаг.* Проанализировать отличительные черты проекта по критериям категорий, представленным в виде вопросов.

*2-й шаг.* Ответить на вопросы по анализируемому проекту, отметив слова «да» или «нет» в соответствующих строках табл. 3.1 – 3.4. Если слов «да» или «нет» в строке несколько, необходимо отметить все из них (все «да» или все «нет»).

В качестве примера в табл. 3.1 выделены варианты ответов для проекта разработки сложного и критичного программного средства, требования к которому заранее не известны и будут уточняться по ходу разработки.

*3-й шаг.* Расположить по степени важности категории (таблицы) и/или критерии, относящиеся к каждой категории (вопросы внутри таблиц), относительно проекта, для которого выбирается модель ЖЦ.

*4-й шаг.* Выбрать из моделей (см. табл. 3.1 – 3.4) ту модель, которая соответствует столбцу с наибольшим количеством отмеченных ответов с учетом их степени важности (с наибольшим количеством отмеченных ответов в верхней части приоритетных таблиц). Выбранная модель ЖЦ является наиболее приемлемой для анализируемого проекта.

Таблица 3.1 – Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № 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.