МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЯЗАНСКОЙ ОБЛАСТИ

ОГБПОУ «РЯЗАНСКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

**РЕФЕРАТ**

на тему:

**«**Язык объектных ограничений OCL**»**

по дисциплине:

Технология разработки ПО

Выполнил:

студент 3 курса,

группы № 97,

Ляпустина Настя

Проверила:

Калашникова В. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Рязань, 2016 г.**

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc469702497)

[**1 ЭВОЛЮЦИЯ МОДЕЛЕЙ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ** 4](#_Toc469702498)

[**1 1 КАСКАДНАЯ МОДЕЛЬ** 4](#_Toc469702499)

[**1 2 СПИРАЛЬНАЯ МОДЕЛЬ** 6](#_Toc469702500)

[**1 3 БЫСТРАЯ РАЗРАБОТКА ПРИЛОЖЕНИЙ** 7](#_Toc469702501)

[**1 4 ИТЕРАЦИОННАЯ МОДЕЛЬ** 9](#_Toc469702502)

[**1 5 КОМПОНЕНТНО-ОРИЕНТИРОВАННАЯ МОДЕЛЬ** 9](#_Toc469702503)

[**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ** 10](#_Toc469702504)

**ВВЕДЕНИЕ**

Жизненный цикл программного обеспечения – это непрерывный процесс, который начинается с момента принятия решения о необходимости его создания и заканчивается в момент его полного изъятия из эксплуатации. Существует целый ряд стандартов, регламентирующих ЖЦ ПО, а в некоторых случаях и процессы разработки.

К программному обеспечению относятся компьютерные, как платные, так и бесплатные программы, используемые для управления компьютером.

Модель жизненного цикла программного обеспечения – это схема, которая определяет четкую и правильную последовательность выполнения и взаимосвязи процессов и действий на протяжении всего жизненного цикла.   
За последние три десятилетия в программировании сменились три основные модели жизненного цикла ПО:

* Каскадная
* Итерационная
* Спиральная

**1 ЭВОЛЮЦИЯ МОДЕЛЕЙ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

**1 1 КАСКАДНАЯ МОДЕЛЬ**

Водопадная или последовательная модель появилась в 1970 году. В ее основе заключается поэтапное выполнение всех стадий разрабатываемого проекта в строгом порядке.

**Содержание основных стадий:**

*Системный анализ* задает роль каждого элемента в компьютерной системе, взаимодействие элементов друг с другом. Поскольку ПО является лишь частью большой системы, то анализ начинается с определения требований ко всем системным элементам и назначения подмножества этих требований программному элементу. Необходимость системного подхода явно проявляется, когда формируется интерфейс ПО с другими элементами . На этом же этапе начинается решение задачи планирования проекта ПО. В ходе планирования проекта определяются объем проектных работ и их риск, необходимые трудозатраты, формируются рабочие задачи и план-график работ.

*Анализ требований* относится к программному элементу – программному обеспечению. Уточняются и детализируются его функции, характеристики и интерфейс.

Проектирование состоит в создании представлений:

* архитектуры ПО
* модульной структуры ПО
* алгоритмической структуры ПО
* структуры данных
* входного и выходного интерфейса (входных и выходных форм данных)

Исходные данные для проектирования содержатся в спецификации анализа, т. е. в ходе проектирования выполняется трансляция требований к ПО во множество проектных представлений: при решении задач проектирования основное внимание уделяется качеству будущего программного продукта.

*Кодирование* – перевод результатов проектирования в текст на языке программирования.

*Тестирование* – выполнение программы для выявления дефектов в функциях, логике и форме реализации программного продукта.

*Сопровождение* – это внесение изменений в эксплуатируемое ПО. **Цели изменений:**

* исправление ошибок
* адаптация к изменениям внешней для ПО среды
* усовершенствование ПО по требованиям заказчика

Сопровождение ПО состоит в повторном применении каждого из предшествующих шагов (этапов) жизненного цикла к существующей программе, но не в разработке новой программы.

Как и любая инженерная схема, классический жизненный цикл имеет достоинства и недостатки.

**Достоинства классического жизненного цикла:**

1. Получение в конце каждой стадии законченного набора проектной документации, отвечающего требованиям полноты и согласованности
2. Простота планирования процесса разработки

Недостатки классического жизненного цикла:

1. Реальные проекты часто требуют отклонения от стандартной последовательности шагов
2. Цикл основан на точной формулировке исходных требований к ПО (реально в начале проекта требования заказчика определены лишь частично)
3. результаты проекта доступны заказчику только в конце работы

На практике, последовательная модель фиксирует важные моменты жизненного цикла, такие, как: точное распределение рабочего процесса и поставленных задач, инициируя тем самым переход к следующему этапу, а также малые циклы, в результате которых достигается компромиссное задание.

**1 2 СПИРАЛЬНАЯ МОДЕЛЬ**

В процессе использования такой модели, программное обеспечение создается в несколько этапов путем прототипирования. Здесь, каждая итерация всецело соответствует фрагменту или версии ПО. На ее основе уточняются цели и общие характеристики проекта, а также оцениваются качественные характеристики полученного результата.

Данная модель определяет четыре действия, представляемые четырьмя квадрантами спирали:

1. *Планирование* – определение целей, вариантов и ограничений.
2. *Анализ* *риска* – анализ вариантов и распознавание/выбор риска.
3. *Конструирование* – разработка продукта следующего уровня.
4. *Оценивание* – оценка заказчиком текущих результатов конструирования.

В современной спиральной модели существует набор контрольных точек:

* Концепция системы
* Цели жизненного цикла и его архитектура
* Первый вариант создаваемой программы, которая подлежит эксплуатации
* Готовый продукт

**Достоинства спиральной модели:**

1. Наиболее реально (в виде эволюции) отображает разработку программного обеспечения;
2. Позволяет явно учитывать риск на каждом витке эволюции разработки;
3. Включает шаг системного подхода в итерационную структуру разработки;
4. Использует моделирование для уменьшения риска и совершенствования программного изделия.

**Недостатки спиральной модели:**

1. Повышенные требования к заказчику;
2. Трудности контроля и управления временем разработки.

**1 3 БЫСТРАЯ РАЗРАБОТКА ПРИЛОЖЕНИЙ**

Модель быстрой разработки приложений (Rapid Application Development) обеспечивает экстремально короткий цикл разработки. RAD – высокоскоростная адаптация линейной последовательной модели, в которой быстрая разработка достигается за счет использования компонентно-ориентированного конструирования. Если требования полностью определены, а проектная область ограничена, RAD-процесс позволяет группе создать полностью функциональную систему за очень короткое время (60–90 дней).

RAD-подход ориентирован на разработку информационных систем и

выделяет следующие этапы:

Бизнес-моделирование. Моделируется информационный поток между бизнес-функциями. Ищутся ответы на следующие вопросы: Какая информация руководит бизнес-процессом? Какая информация генерируется? Кто генерирует ее? Где информация применяется? Кто обрабатывает ее?

Моделирование данных. Информационный поток, определенный на этапе бизнес-моделирования, отображается в наборе объектов данных, которые требуются для поддержки бизнеса. Идентифицируются характеристики (свойства, атрибуты) каждого объекта, определяются отношения между объектами;

Моделирование обработки. Определяются преобразования объектов данных, обеспечивающие реализацию бизнес-функций. Создаются описания обработки для добавления, модификации, удаления или нахождения (исправления) объектов данных;

Генерация приложения. Предполагается использование методов, ориентированных на языки программирования 4-го поколения. Вместо создания ПО с помощью языков программирования 3-го поколения RAD-процесс работает с повторно используемыми программными компонентами или создает повторно используемые компоненты. Для обеспечения конструирования применяются утилиты автоматизации;

Тестирование и объединение. Поскольку применяются повторно используемые компоненты, многие программные элементы уже протестированы. Это уменьшает время тестирования (хотя все новые элементы должны быть протестированы).

Применение RAD возможно в том случае, когда каждая главная функция может быть завершена за 3 месяца. Каждая главная функция адресуется отдельной группе разработчиков, а затем интегрируется в целую систему.

Применение RAD имеет и свои недостатки, и ограничения:

Для больших проектов в RAD требуются существенные людские ресурсы (необходимо создать достаточное количество групп);

RAD применима только для таких приложений, которые могут декомпозироваться на отдельные модули и в которых производительность не является критической величиной;

RAD неприменима в условиях высоких технических рисков (т. е. при использовании новой технологии).

**1 4 ИТЕРАЦИОННАЯ МОДЕЛЬ**

Данная модель является альтернативой каскадной модели. Она предполагает распределение жизненного цикла программы на поэтапность итераций, целью которых есть получение функционирующей версии программной системы. Конечный результат последней итерации обладает всей необходимой функциональностью продукта. Таким образом, с завершением каждой итерации, программа развивается эволюционно.

**1 5 КОМПОНЕНТНО-ОРИЕНТИРОВАННАЯ МОДЕЛЬ**

Компонентно-ориентированная модель является развитием спиральной. В этой модели конкретизируется содержание квадранта конструирования – оно отражает тот факт, что в современных условиях новая разработка должна основываться на повторном использовании существующих программных компонентов.

Программные компоненты, созданные в реализованных программных проектах, хранятся в библиотеке. В новом программном проекте, исходя из требований заказчика, выявляются кандидаты в компоненты. Далее проверяется наличие этих кандидатов в библиотеке. Если они найдены, то компоненты извлекаются из библиотеки и используются повторно. В противном случае создаются новые компоненты, они применяются в проекте и включаются в библиотеку.

Достоинства компонентно-ориентированной модели:

1. уменьшает на 30 % время разработки программного продукта
2. уменьшает стоимость программной разработки до 70 %.
3. увеличивает в полтора раза производительность разработки.
4. Недостатком такой модели является сложность организации процесса разработки ПО по данной модели.

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. [www.besplatnyeprogrammy.net](http://www.besplatnyeprogrammy.net)
2. <http://www.administrating.ru/evolyuciya-modeley-zhiznennogo-cikla-programmnogo-obespecheniya/>