**Методы проектирования и разработки приложения**

**Метод проектирования**

Сутью структурного программирования является возможность разбиения программы на составляющие элементы. Распространены две методики (стратегии) разработки программ, относящиеся к структурному программированию: программирование «сверху вниз» и программирование «снизу вверх».

Достоинства структурного программирования:

* повышается надежность программ (благодаря хорошему структурированию при проектировании, программа легко поддается тестированию и не создает проблем при отладке);
* повышается эффективность программ (структурирование программы позволяет легко находить и корректировать ошибки, а отдельные подпрограммы можно переделывать (модифицировать) независимо от других);
* уменьшается время и стоимость программной разработки;
* улучшается читабельность программ.

Для разработки дипломного проекта была выбрана более предпочтительная структура «сверху вниз».

Программирование «сверху вниз», или нисходящее программирование – это методика разработки программ, при которой разработка начинается с определения целей решения проблемы, после чего идет последовательная детализация, заканчивающаяся детальной программой. Является противоположной методике программирования «снизу вверх».

При нисходящем проектировании задача анализируется с целью определения возможности разбиения ее на ряд подзадач. Затем каждая из полученных подзадач также анализируется для возможного разбиения на подзадачи. Процесс заканчивается, когда подзадачу невозможно или нецелесообразно далее разбивать на подзадачи.

В данном случае программа конструируется иерархически - сверху вниз: от главной программы к подпрограммам самого нижнего уровня, причем на каждом уровне используются только простые последовательности инструкций, циклы и условные разветвления.

**Метод разработки**

В качестве метода разработки проекта была выбрана модель жизненного цикла «Incremental Model» (инкрементная модель).

В инкрементной модели полные требования к системе делятся на различные сборки. Терминология часто используется для описания поэтапной сборки ПО. Имеют место несколько циклов разработки, и вместе они составляют жизненный цикл «мульти-водопад». Цикл разделен на более мелкие легко создаваемые модули. Каждый модуль проходит через фазы определения требований, проектирования, кодирования, внедрения и тестирования. Процедура разработки по инкрементной модели предполагает выпуск на первом большом этапе продукта в базовой функциональности, а затем уже последовательное добавление новых функций, так называемых «инкрементов». Процесс продолжается до тех пор, пока не будет создана полная система.



Рисунок 1. Схема инкрементной модели

Использовать инкрементную модель следует, когда:

* основные требования к системе четко определены и понятны (в то же время некоторые детали могут дорабатываться с течением времени);
* требуется ранний вывод продукта на рынок;
* есть несколько рисковых целей.

Каждый инкремент поставляется клиенту по мере готовности, что позволяет сразу начать использовать продукт и избежать длительной разработки. Также это стимулирует большие инвестиционные затраты, но сокращает время ожидания результата. Модель помогает «сгладить углы», вместо того чтобы выкатывать пользователю совершенно новую систему разом.

Есть и проблемы. Например, каждый новый инкремент должен быть интегрирован с предыдущим во всех системах. Разборка на инкременты тоже не элементарна. Как бы то ни было, большое количество сборок означает малое количество изменений между ними. Для того, чтобы лучше понять, о чём идет речь, рассмотрим преимущества и недостатки данной модели.

Преимущества инкрементной модели:

* рабочее приложение выходит на ранней стадии жизненного цикла продукта;
* гибкость (изменить масштабы и требования проекта относительно менее затратно);
* небольшие итерации упрощают тестирование и внесение правок;
* проще идентифицировать риски, справиться с ними;
* каждая итерация — простая в управлении контрольная точка проекта;

Недостатки инкрементной модели:

* каждая фаза итерации неподвижна;
* могут возникнуть проблемы относительно архитектуры системы, так как не все требования собраны заранее для всего жизненного цикла ПО;