Тема 7. Разработка структуры программы и модульное программирование

7.1. Цель модульного программирования

Обычно программы ПС являются большими системами, поэтому их разрабатывают по частям – модулям.

Программный модуль – это любой фрагмент описания процесса, оформляемый как самостоятельный программный продукт.

Каждый программный модуль программируется, компилируется и отлаживается отдельно от других модулей программы.

7.2. Основные характеристики программного модуля

Майерс предлагает использовать характеристики модуля:

А – размер модуля

Б – прочность модуля

В – сцепление с другими модулями

Г – рутинность модуля

А – размер модуля

Измеряется числом содержащихся в нем операторов или строк.

//Рекомендуются программные модули размером от нескольких десятков до нескольких сотен операторов.

Б – прочность модуля – это мера его внутренних связей

Чем выше прочность модуля, тем больше связей он может спрятать от внешней по отношению к нему части программы, и, следовательно, тем больший вклад в упрощение программы он может внести.

Майерс предлагает упорядоченный по степени прочности набор из 7 классов модулей:

1) Самая слабая степень прочности – у модуля «прочного по совпадению». Это такой модуль, между элементами которого нет осмысленных связей. Использовать не рекомендуется.

2-5) не рекомендуется

6) Функционально прочный модуль – модуль, выполняющий одну какую-либо определенную функции.

7) Информационно-прочный модуль – выполняется несколько операций или функций над одной и той же структурой данных (информационным объектом). Для каждой из этих операций в таком модуле имеется свой вход со своей формой обращения к нему. Такой модуль является модулем с наивысшей степенью прочности.

B – сцепление с другими модулями

Это мера его зависимости по данным от других модулей. Характеризуется способом передачи данных.

Виды сцепления модулей:

- Сцепление по содержимому – худший вариант

(один из модулей имеет прямые ссылки на содержимое другого модуля: например, на константу, содержащуюся в другом модуле; такое сцепление модулей недопустимо)

- Сцепление по общей области

(не сколько модулей используют одну и ту же область памяти, не рекомендуется)

- Параметрическое сцепление

(данные передаются модулю либо при обращении к нему как значения его параметров, либо как результат его обращения к другому модулю для вычисления некоторой функции).

Г – рутинность модуля – это его независимость от предыстории обращений к нему.

Модуль рутинный, если результат обращения к нему зависит только от значений его параметров.

Модуль зависящий от предыстории – когда результат обращения к нему зависит от внутреннего состояния этого модуля, изменяемого в результате предыдущих обращений к нему. В некоторых случаях является лучшей реализацией информационно прочного модуля (в спецификации зависящего от предыстории модуля должна быть четко сформулирована эта зависимость таким образом, чтобы было возможно прогнозировать поведение (эффект выполнения) данного модуля при разных последующих обращениях к нему).

7.3. Методы разработки структуры программы

В качестве модульной структуры программы принято использовать древовидную структуру, включая деревья со сросшимися ветвями.

В узлах такого дерева размещаются программные модули, а направленные дуги показывают статическую подчиненность модулей, т.е. каждая дуга показывает, что в тексте модуля, из которого она исходит, имеется ссылка на модуль, в который она включена.

Модульная структура программы должна включать совокупность спецификаций модулей, образующих эту программу.

Спецификация программного модуля содержит:

- Синтаксическую спецификацию его входов, позволяющую построить на используемом языке программирования синтаксически правильное обращение к нему (к любому его входу).
- Функциональную спецификацию модуля (описание семантики функций, выполняемых этим модулей по каждому из его входов)

В процессе разработки программы ее модульная структура может по-разному формироваться и использоваться для определения порядка программирования и отладки и отладки модулей, указанных в структуре ПС.

Метод восходящей разработки

1. Строится модульная структура программы в виде дерева.
2. Поочередно программируются модули программы, начиная с модулей самого нижнего уровня (листья) таким образом, чтобы для каждого программируемого модуля были уже запрограммированы все модули, к которым он может обращаться.

Метод нисходящей разработки

1. Строится модульная структура программы в виде дерева.

2. Поочередно программируются модули программы, начиная с модуля самого верхнего уровня (головного).

После того, как все модули запрограммированы, производится их поочередное тестирование и отладка в таком же порядке.

Особенность данных методов – требование, чтобы модульная структура программы была разработана до начала программирования модулей.

Это требование находится в полном соответствии с водопадным подходом к разработке ПС.

Недостатки: не всегда можно разработать структуру программы до начала программирования.

Поэтому предлагаются конструктивный и архитектурный подходы к разработке программ, в которых модульная структура формируется в процессе программирования модулей.

Конструктивный подход представляет собой модификацию нисходящей, при которой модульная древовидная структура программы формируется в процессе программирования модулей.

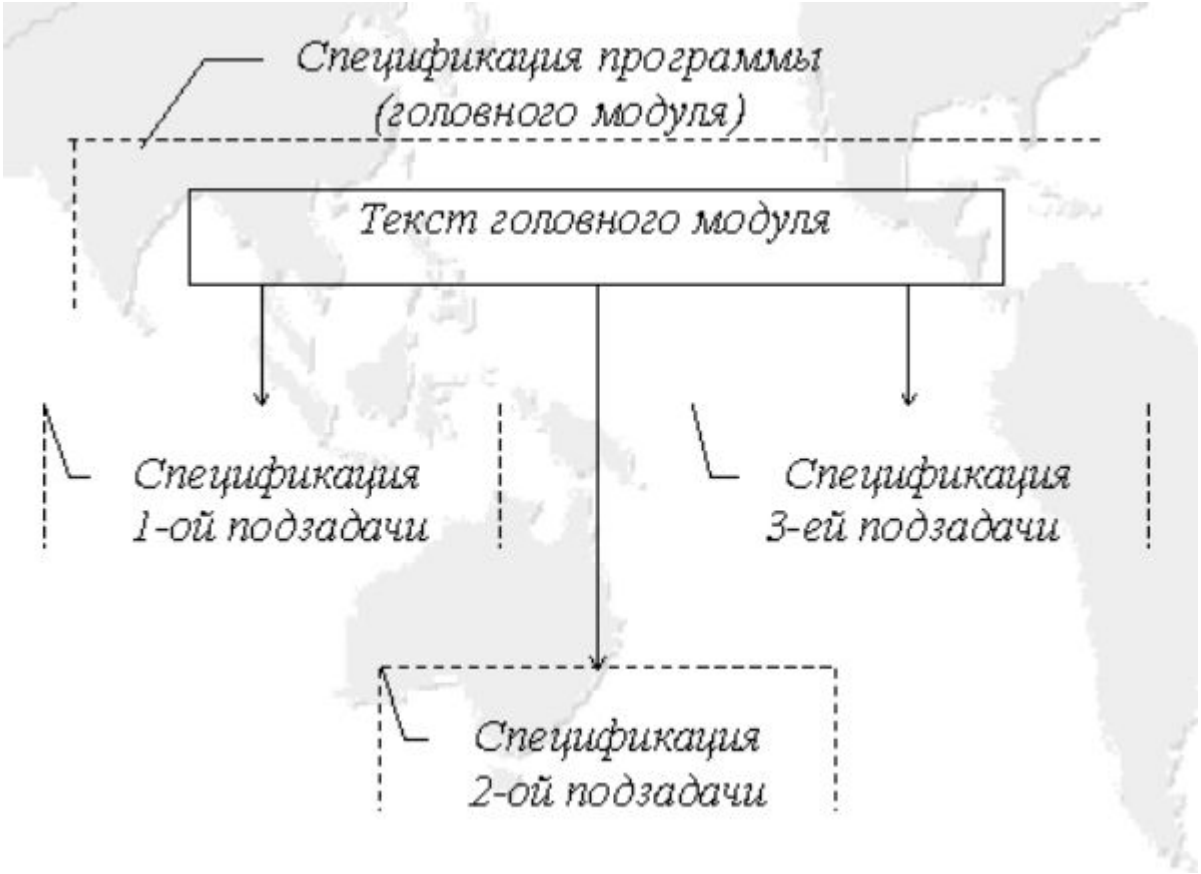
Разработка программы начинается с программирования головного модуля, исходя из спецификации программы в целом.

Если программа большая, то выделяют подзадачи.

Таким образом, на первом шаге разработки формируется верхняя часть дерева (рисунок 1.

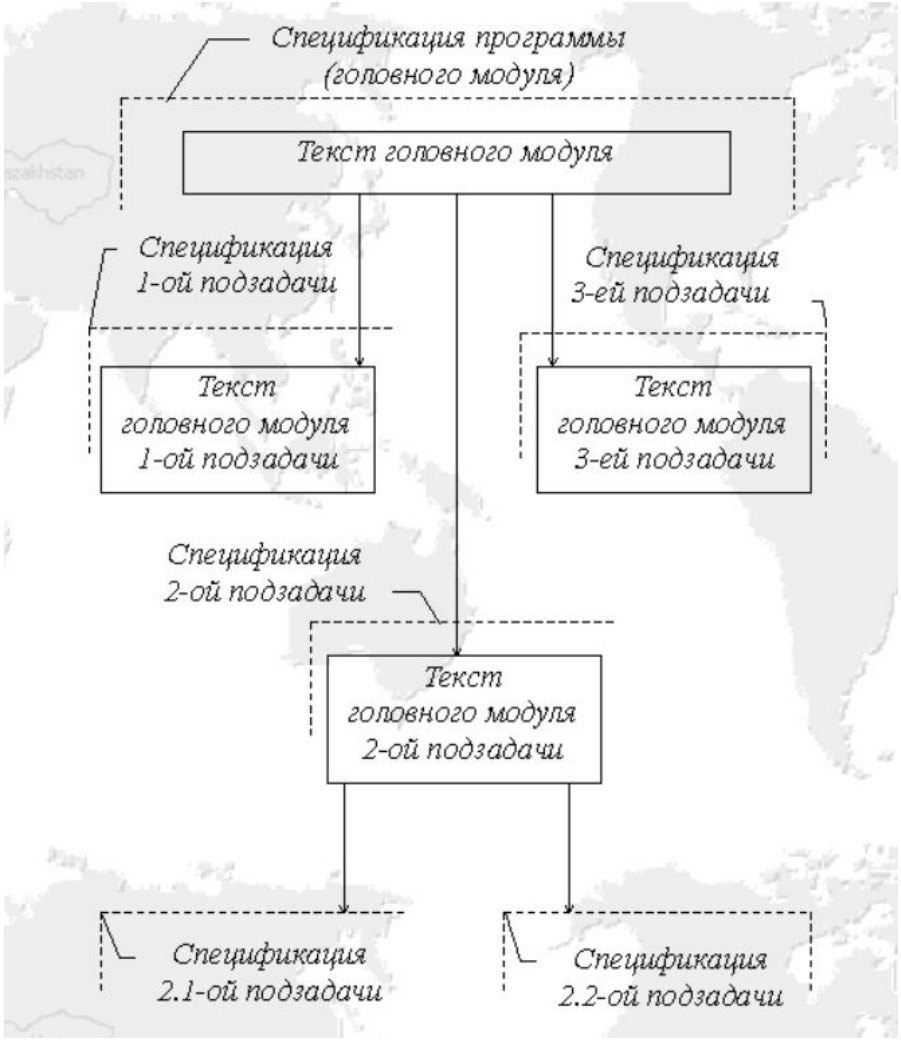
Рисунок 1. Первый шаг формирования модульной структуры программы при конструктивном подходе.

7.1.



Аналогичные действия производятся при программировании любого другого модуля, который выбирается из текущего состояния дерева программы.

Рис.7.2. Второй шаг формирования модульной структуры программы при конструктивном подходе

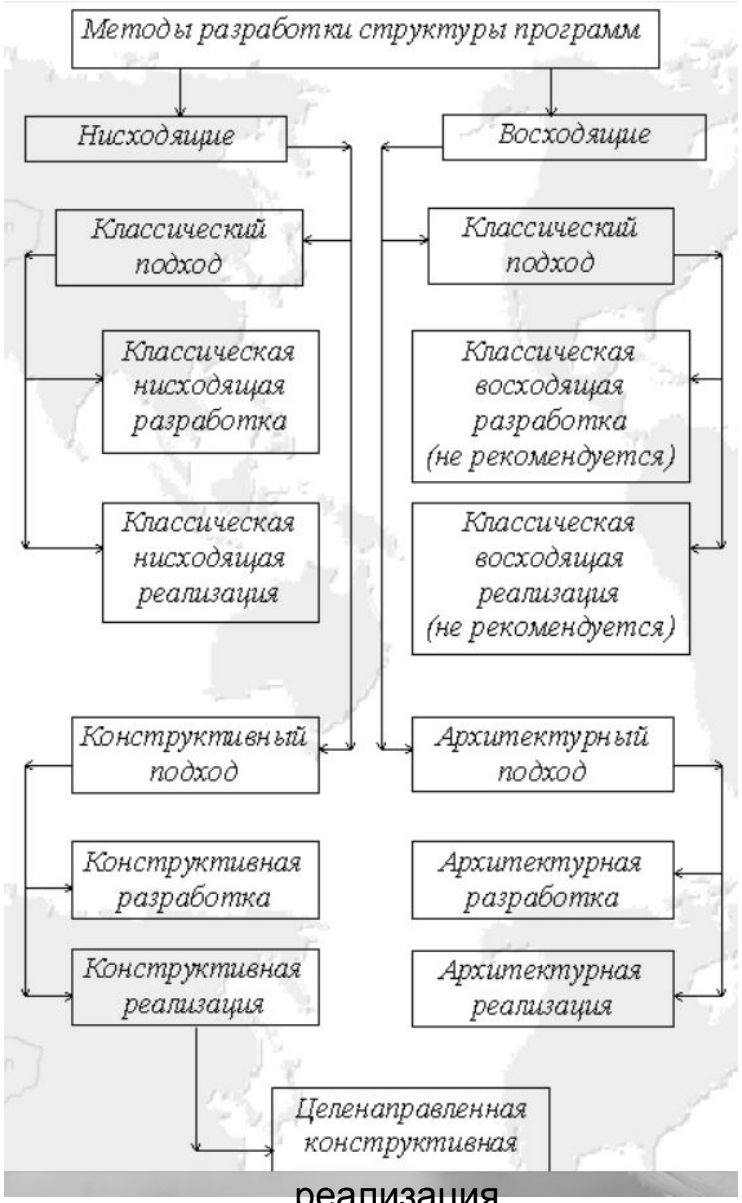


Архитектурный подход к разработке программы представляет собой модификацию восходящей разработки, при которой модульная структура программы формируется в процессе программирования модуля.

При этом ставится существенно другая цель разработки: повышение уровня используемого языка программирования, а не разработка конкретной программы.

Это означает, что для заданной предметной области выделяются типичные функции, каждая из которых может использоваться при решении разных задач в этой области, специфицируются, а затем и программируются отдельные модули.

Рис.7.3. Классификация методов разработки структуры программ



7.4. Контроль структуры программы

При классическом подходе разработки ПС:

1. Статический контроль

Оценка структуры с учетом характеристик модуля

2. Смежный контроль

Контроль со стороны разработчиков архитектуры и внешнего описания ПС.

3. Сквозной контроль

Это мысленная проверка структуры программы при выполнении заранее разработанных тестов.

При конструктивном и архитектурном подходах контроль структуры программы осуществляется в процессе программирования модулей в подходящие моменты времени.