Проектирование ПО по структурному подходу

Процесс проектирования любого ПО начинается с уточнения его структуры (структурных компонентов и связей между ними). Результат уточнения может быть представлен в виде структурной или функциональной схемы и описания спецификации компонентов.

Структурная схема – схема, отражающая состав и взаимодействие по управлению частей ПО.

Структурные схемы никогда не информативны с точки зрения структуры отдельных компонентов (черные ящики) связано с тем, что не предусматривается отображения передачи управления между отдельными частями.

Поэтому структурные схемы разрабатываются для каждого программного пакета, а список программ пакета определяют отдельно, учитывая требования к техническому заданию.

Разрабтку проводят методом пошаговой детализации, отображая, каждый отедльынй структурный компонент в схеме и взаимосвязи с другими компонентами. В таком ПО всегда есть

программа диспетчер, котоорая напрямую может управлять отдлеьынми компонентами.

Структурная схема – практически всегда иерархическая структура за очень редким исключением, когда указываются взаимосвязи между отдлеными компонентами.

Более полное рпедставлемом разрабатываемом ПО представляет функциональная схема, представляющая схему взаимодействия компонентов ПО с описанием информационных потоков, состава данных в этих потоках и указания использования отдельных файлов, устройств и интерфейсов.

Для каждого компонента определяем ввод/вывод данных.

Описание происходит для каждого входа и выхода из системы

Эти схемы – описание программного комплекса ПО.

Если из программного комплекса сделать программную систему, то вместо диспетчера будет система управления компонентами. В структурной тоже самое, но без входов выходов. В функциональной: указываем входы выходы файлы порты интерфейсы.

В основном у нас функц схемы прогр систем.

Используя структурный подход к проектированию ПОзачастую надо проводитть декомпозицию функц схем.

Декомпозицию делают по методике Констатайна или Джексона

Констанстайн:

Отношения между модулями ПО в виде графа, вершины – модули и общие области данных (файлы бд удаленные стеевые ресурсы). А дуги – межмодульные вызовы и обращения общим областям данных.

4 типа вершин.

1. Модуль (подпрограмма)
2. Подсистема (самостоятельная програ использ межмеодульные вызовы)
3. Библа (совокупность подпрограмм, размещенная в отдельном модуле, всегда управляющаяся из вне)
4. Область данных – спец образом оформленная совокупность данных, к которой возможно обращение из вне

Методика Джексона появилась на равне с методикой Варнье Орра. Эти методики основаны на декомпозиции данных, предназанчены для создания простых прог, работающих с иерархическими структурами данных: т.е. методики К и Дж можно совмещать: по К отображать общую структуру ПО, декомпозируя отдельные модули до самых простых, а по Дж представляется структура отдельных простых модулей в сложной иерархии.

Методика Джексона основана на посике соответствия структур исходных данных и результатов, однако при ее применении возможны ситуации, когда на каких-то уровнях соответствия отсутствуют. Такие ситуации – столкновения и по Джексону не могут быть разрешены (на схеме изображается так как есть as is). Но методика Варнье Орра действует иначе: базируется на том же положении, что и Дж, но при построении программ (компонентов) считаются структуры входных данных и соответсвующие им стркутуры выходных данных. Должны быть взаимоувязаны. Если случается столкновение, то по Варнье структуры выходных данных допускается менять в соответсвии от входных данных и функционала компонента.

Варнье Орра есть в ЕСПД.

Основываясь на свойствах ООП, для этапа проектирования были выделены свои требования и свойства.

1. Уменьшение сложности ПО (разработка такой структуры, позволяющей разрабатывать минимум объектов с четкими и понятными характеристикиами)
2. Повышение надежности ПО (тестируются и формируются отдельные объекты доводя их отказоустойчивость до 100% и только потом их них собирается структура ПО)
3. Обеспечение возможности модификации ПО (когда изменения вносятся в конкретные объекты не затрагивая остальные)
4. Отдельно разработанные объекты ПО, можно использовать в сборке других систем и ПО.
5. В полной мере объектный процесс организован в процессе Хуанна???
6. Для разработки ПО, при использовании объектного языка UML, могут применяться разные модели проектирования. Классика: разработка 8 диаграмм последовательно, причем вытекают друг из друга.

ДВИ ДК кооперации состояний деятельности последовательности компонентов разветка

Объектный подход предполагает выбор из следующих:

* 1. Логическая модлеь (ключевыве абстракции предм обл обеспечивающей функциональность ПО) дви состояния последовательности
  2. Модель процессов (проектирование процессов и нитей, определяющие производительность и масштабируемость) (кооперации деятельности последовательности классов)
  3. Модель реализации (реальная организация программных модулей и отдельных файлов в среде разработки (компоненты, классы, состояний)
  4. Модель развёртывания (территориальное размещение с учетом особенностей и взаимосвязей) (компоненты, развертывания,

Группа разработки при таком подходе – узконаправленные специалисты, в области программирования 80%, проектирования 20%, никогда не войдут аналитики, специалисты по активности пользователей, без руководителей группы.