Национальный открытый институт России

СОЦИАЛЬНО-ЭКОНОМИЧЕСКИЙ ФАКУЛЬТЕТ

Кафедра математики и информатики

Вводная лекция по дисциплине

«Разработка, внедрение и адаптация ПО»
(Введение, определение, стадии, этапы проектирования ПО)

Составитель: Рачева Наталья Васильевна

Санкт-Петербург
2018 г.

Программный продукт

Программный продукт (ПП) — программное обеспечение, готовое к промышленной эксплуатации (без вмешательства авторов – «отчуждение» от создателей). Спроектирована в едином стиле, оттестирована до требуемого уровня надежности, сопровождена подробной документацией и подготовлена для тиражирования. ***Программное изделие*** — программа на носителе данных, являющаяся продуктом промышленного производства.

Программное обеспечение автоматизированных систем (ПО АС) — совокупность программ на носителях данных и программных документов, предназначенная для проверки работоспособности АС.

Проект (от лат. *projectus* — брошенный вперед) — совокупность проектных документов в соответствии с установленным перечнем, которая отражает результат проектирования.

Проектирование — это разработка проекта, процесс создания спецификации, необходимой для построения в заданных условиях несуществующего объекта на основе первичного описания этого объекта. Результат - совокупность решений согласно требованиям в обязательной форме представления.

Программный продукт - продолжение

Альфа-тестирование *(системное тестирование, лабораторные испытания)* — фаза тестирования, выполняемая *разработчиками* для подтверждения, что все фрагменты правильно интегрированы в систему, система работает надежно.

Автономное тестирование *(тестирование модуля) (module testing)* — контроль отдельного модуля в изолированной среде (например, с помощью ведущей программы), инспекция текста модуля на сессии программистов, которая иногда дополняется математическим доказательством правильности модуля.

Артефакт реализации — нечто, что нельзя обнаружить в постановке решаемой задачи, но необходимое для составления программы.

Бета-тестирование — это фаза общего тестирования, при которой программное изделие поставляется ограниченному кругу *конечных пользователей* для *жесткого* тестирования.

Комплексное тестирование *(system testing)* — контроль и/или испытание системы по отношению к исходным целям. Является процессом контроля, если оно выполняется в моделируемой среде, и процессом испытания при выполнении в реальной среде.

Инженерия программирования - ИП

ИП – (англ. *software engineering) иначе разработка ПО,* формальный процесс на базе специфических методов (методиках), творческая техническая деятельность, имеющий практическую направленность.

Дата появления – 1968 год реакция на условия большой сложности.

Общие принципы:

Частотный принцип. Принцип основан на выделении в алгоритмах и данных особых групп по частоте использования. «Частые» операции стараются делать более короткими.

Принцип модульности. Под модулем в данном контексте понимают функциональный элемент рассматриваемой системы, имеющий оформление, законченное и выполненное в пределах требований системы

Принцип функциональной избирательности. Выделяется часть важных модулей, которые постоянно должны быть в состоянии готовности для эффективной организации вычислительного процесса, это называют ядром или монитором и постоянно хранятся в ОП.

Инженерия программирования – ИП - продолжение

Принцип генерируемости. Способ настройки на конкретную конфигурацию технических средств, круг решаемых проблем, условия работы пользователя..

Принцип функциональной избыточности. Учет проведения одной и той же работы различными средствами, это важно при разработке пользовательского интерфейса для выдачи одних и тех же данных из-за психологических различий в восприятии информации.

Принцип «по умолчанию». Облегчение организации связей с системой на стадии генерации и работе с уже готовыми программами.

Стратегия (долгосрочность) охватывает теорию и практику подготовки к выполнению проекта, общее планирование тактик ведения проектов. **Тактика** (фиксированная последовательность средств и приемов) определяет, как двигаться, какие конкретные действия предпринимать при затруднениях в ходе выполнения проекта.

Стратегия выполнения конкретного проекта описывается в программном документе — **техническом задании**.

Технология программирования (ПО) – Т_ПО

Т_ПО - описание совокупности методов и средств разработки программ и порядок применения этих методов и средств. синтез осуществляют на основе одной, двух и даже трех технологий.

охватывает не только проектирование и производство, но и структуры организаций с взаимодействием людей. В терминах инженерного дела технология — инструментарий инженера, интеллектуальный либо отчужденный в виде искусственных систем.

В технологии программирования методы рассматриваются «сверху» — с позиции организации технологических процессов, в методологии (совокупность методов, накопленных на практике, в разработке ПО) — «снизу» — с точки зрения основ их построения.

На начальном этапе проектирования анализ потребностей определяет вид объекта; его функции (что объект делает при функционировании), свойства, состояние для удовлетворения потребностей и качества ПО.

При исследовании систем решаются задачи анализа и синтеза.

Анализ — процесс функционирования по заданному описанию системы.

Синтез — процесс построения описания системы по заданному функционированию.

Системный подход в Инженерии ПО

Системный подход — общенаучный обобщенный эвроритм, предусматривающий всестороннее исследование сложного объекта с использованием компонентного, структурного, функционального, параметрического и генетического видов анализа.

Компонентный анализ — выбор объекта, включающего в себя составные элементы и входящего в систему более высокого ранга.

Структурный анализ — выявление элементов объекта и связей между ними.

Функциональный анализ— оценка объекта на выполнение всех функций полезных и вредных.

Параметрический анализ — установление качественных пределов развития объекта — физических, экономических, экологических и др. , т. е. размер, время выполнения применяемого алгоритма, объем памяти и т. д.

Генетический — исследование объекта на соответствие законам развития программных систем, т.е. история развития (генезис – языки программирования, технологии, аналоги конструкций, объемы тиражей).

Блочно-иерархический подход

Бл.-иерарх. – объект разбивается на уровни. Высший уровень –общие черты. Далее детализация на каждом последовательном уровне.

Методология бл.-иерар. подхода основывается на разбиении и локальной оптимизации, абстрагировании, повторяемости.

Концепция *разбиения* – сложное разложить на простые задачи.

Локальная оптимизация – это улучшение в рамках простой задачи.

Абстрагирование - построение моделей, отражающих значимые в данных условиях свойства объектов.

Повторяемость – использование предыдущий опыт проектирования.

Достоинство бл.-иерар. подхода – документация становится обзримой и воспринимаемой одним исполнителем.

Недостаток бл.-иерар. подхода – неточность моделей высших уровней.

Парадигма программирования

Парадигма (Томас Кун, 1977 г. *modus operandi*) программирования – способ (правило развития научного мышления) мышления в программировании.

Приемы и способы программирования конкретного программиста определяются используемым языком.

Основные парадигмы программирования

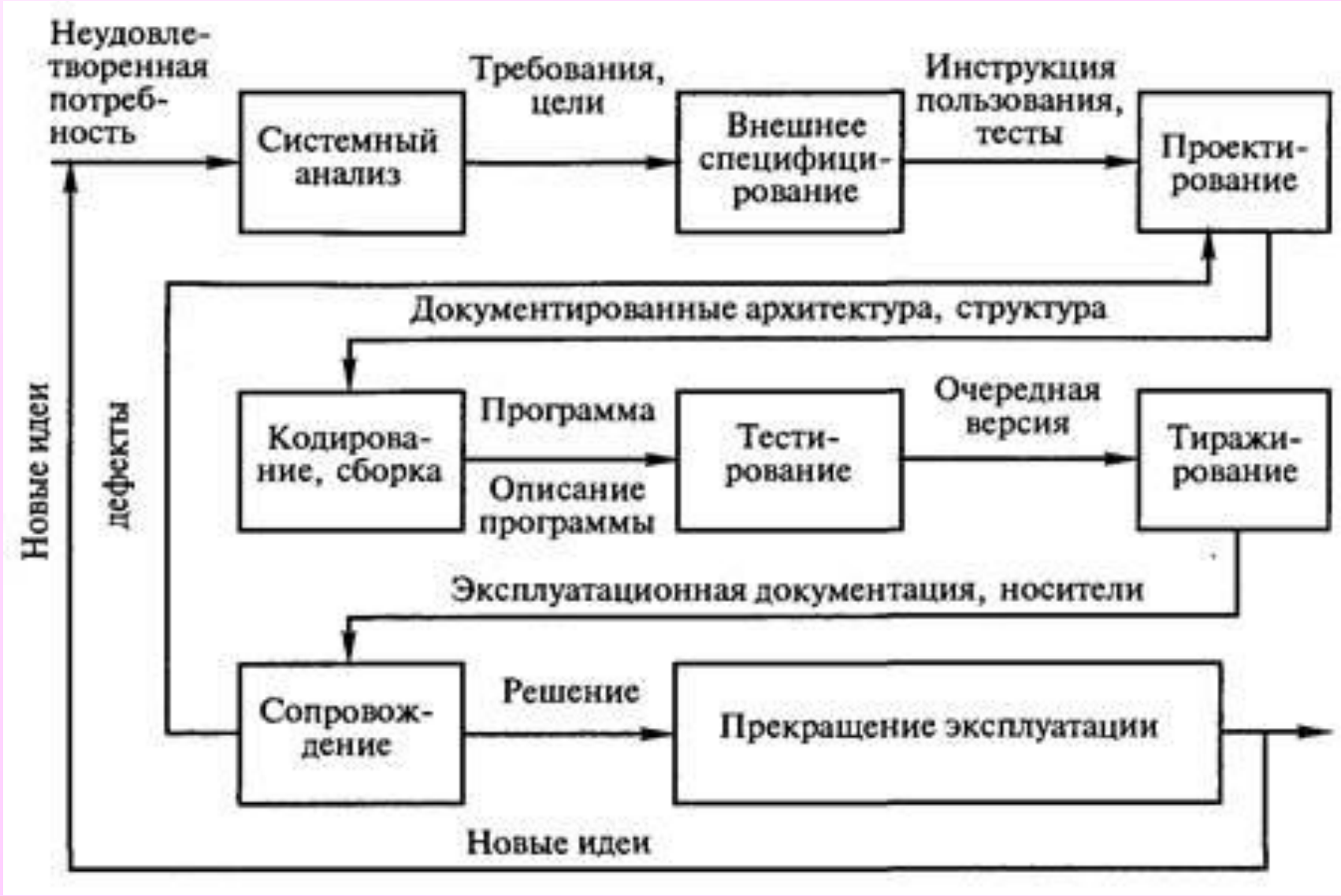
- *процедурно-ориентированные (алгоритмы);*
- *объектно-ориентированные (объекты, классы);*
- *логически-ориентированные – цели в исчислении предикатов;*
- *ориентированные на правила – «ЕСЛИ….ТО»;*

-ориентированные на ограничения - инвариантные соотношения;

-параллельное программирование – потоки данных.

Объектно-ориентированная парадигма является наиболее приемлемой для широкого круга задач, в которых основной проблемой является сложность.

Жизненный цикл ПО



Стадии разработки ПО

Стадия проекта — одна из частей процесса создания программы, установленная нормативными документами и заканчивающаяся выпуском *проектной документации*, содержащей описание полной, в рамках заданных требований, модели программы на заданном для данной стадии уровне, или изготовлением программ. По достижении окончания стадии заказчик имеет возможность рассмотреть состояние проекта и принять решение по дальнейшему продолжению проектных работ.

Например, заказчик может принять решение о продолжении работ по одному из согласованных вариантов.

Этап проекта —это часть стадии проекта, выделенная по соображениям единства характера работ и (или) завершающего результата или специализации исполнителей. Но выделяют и этапы (фазы), которые охватывают несколько стадий.

Например, этап проектирования программы включает стадии ЭП и ТП. Описания этапов регламентируют порядок выполнения отдельных видов работ для достижения стадии. Одни и те же виды работ могут продолжаться на нескольких этапах.

Техническое задание (ТЗ)

ТЗ – основные требования к ПО, порядок взаимодействия «заказчик- исполнитель», перечень стадий и этапов и возможность изменений. В ТЗ определяется стратегия проекта.

Эскизный проект (ЭП) необходим для разработки нескольких альтернативных вариантов реализации будущего изделия и уточнения требований на основе их анализа. Степень проработки при этом должна быть достаточной для достижения возможности сравнения вариантов.

Технический проект (ТП) - однозначное описание конечного) варианта построения ПО и порядка его реализации.

Рабочий проект (РП) - реализации ПО в соответствии с ранее намеченным планом.

Основанием для заключения договора между заказчиком и исполнителем служит гарантийное письмо заказчика. На основании гарантийного письма составляется договор. Обязательным приложением к договору является ТЗ.

Этапы проектирования ПО

1) Анализ требований, предъявляемых к ПО (системный анализ). (проводится на основе первичного исследования потоков информации при существующем порядке: фиксация последовательности и типов).

2) Определение целей, достигаемых разрабатываемым ПО;

3) Выявление аналогов, обеспечивающих достижение подобных целей.

4) Постановка задачи на разработку новых программ, определение внешних спецификаций (т. е. описаний входной и выходной информации и их форм) и способов (алгоритмов, методов) обработки информации.

5) Оценка достижения целей разработки (при необходимости, этапы 1–5 м.б. итеративно повторены до достижения удовлетворительного облика ПО с описанием выполняемых им функций и ясностью реализации ПО).

6) Рассмотрение возможных вариантов структурного построения ПО, результатом этой работы являются варианты архитектуры ПО и (или) требования к структуре отдельных программных компонент; организация файлов для межпрограммного обмена данными.

7) Разработка окончательного варианта архитектуры ПО и разработка окончательной структуры программных компонент.

Этапы проектирования ПО - продолжение

8) Составление и проверка спецификаций модулей. Спецификация — в проектной деятельности это описание всех моментов в точных терминах.

9) Составление описаний логики модулей.

10) Составление конечного плана реализации программных модулей.

11) Кодирование и тестирование отдельных модулей и совокупности готовых модулей до получения готового ПО.

12) Комплексное тестирование ПО.

13) Разработка эксплуатационной документации на ПО.

14) Проведение приемо-сдаточных и других испытаний ПО.

15) Корректировка ПО по результатам испытаний.

16) Окончательная сдача ПП (ПО) заказчику.

17) Тиражирование ПП (программного изделия).

18) Сопровождение ПО.

Современные технологии проектирования ПО направлены на частичную автоматизацию этапов и на совмещение их во времени с целью сокращения сроков выполнения проектов.

Типовые ошибки при составлении ТЗ

Типовыми ошибками являются написания неконкретных требований, которые никого ничему не обязывают. Требования не должны быть противоречивыми.

Типовые ошибки при написании требований к функциональным характеристикам:

- непонимание термина «функциональные характеристики» (смысл этого термина характеризуется следующим предложением: фраза должна содержать как назначение проектируемого объекта, так и то, *что выполняет ПО* и при каких ограничениях. Для правильного написания требований к функциональным характеристикам необходимо сторонними глазами будущего пользователя рассмотреть, что делает полезного ПО и при каких ограничениях.);
- описание требований к функциональным характеристикам всеобщего, универсального объекта (если по конкретным требованиям можно реализовать целый спектр функций, то они не конкретны);
- написание заведомо нереализуемых требований.

Понятие спецификаций ПО

Под *спецификацией* понимается достаточно полное и точное описание решаемой задачи на этапах проекта. Спецификация является моделью проектируемого объекта (программы).

Сложность проектируемых систем привела к созданию специальных абстрактных языков, графических нотаций и поддерживающих их автоматизированных систем, облегчающих процесс создания спецификаций.

Первичные спецификации составляют в терминах решаемой задачи, а не программы. В ходе выполнения проекта спецификации последовательно претерпевают изменения до программных документов стадий и вплоть до документации, которая необходима для сопровождения программы.

В первичных спецификациях можно выделить две части: функциональную и эксплуатационную.

Спецификации на разработку ПО

Первичная функциональная спецификация в первую очередь описывает:

— *объекты*, участвующие в задаче (что делает программа и что делает человек, работающий с этой программой);

— *процессы и действия* — эвроритмы для человека, алгоритмы с указанием сути и порядка обработки информации;

Эвроритм - порядок действия человека при выполнении какой-то деятельности. В отличие от алгоритма может изменяться в процессе исполнения благодаря разумности исполнителя.

— *входные и выходные данные*, их организацию.

Внешние функциональные спецификации включают:

— *описание* того, что делает программа;

— определение, что *делает человек (*по каким эвроритмам работает человек, откуда он берет информацию и как ее готовит к вводу в ЭВМ);

— спецификации *входных и выходных данных*;

— реакции на *исключительные ситуации*.

Внутренние функциональные спецификации

Описания состава внутренних частей программы, их взаимосвязи - внутренние функциональные спецификации. ПО можно представить как набор процедурных абстракций.

Абстракции процедур наиболее полно воплотились в технологии структурного программирования.

Правила при анализе спецификаций

Абстракция через *параметризацию* позволяет представить фактически неограниченный набор различных вычислений одной процедурой. *Например,* необходима процедура сортировки массива целых чисел A. При дальнейшей разработке программы возможно возникновение потребности в сортировке другого массива, но уже с другим именем.

Правило 1. Выполнение конечного условия при соблюдении начальных условий — это то, ради чего и построена процедура. Процедура поиска максимального значения в массиве, конечное условие — факт, что максимальный элемент найден. Процедура вычисления квадратного корня, конечное условие — нахождение квадратного корня.

Правило 2. Можно ограничиться только той информацией, которую подразумевает конечное условие.

Подробности алгоритма отыскания квадратного корня, программист не знакомиться с текстом процедуры благодаря абстракции.

Абстракции через параметризацию позволяют определить два вида абстракций: *процедурную* абстракцию и абстракцию *данных*.

DFD — Data Flow Diagram

ДПД (**DFD**) отражает функциональные зависимости значений, вычисляемых в системе, включая входные значения, выходные значения и внутренние хранилища данных.

Диаграммы потоков данных (ДПД)

а — копирование данных (числа); *б* — расщепление данных; *в* — активный объект «Клиент» посредством операции «Нахождение цены» работает с хранилищем «Прайс-лист»

Абстракция объекта

Парадигма программирования на основе абстрактных данных Дейкстры: определить организацию данных и выявить все состояния значений данных, считать, что процедуры — это нечто, изменяющее данные. При структурном подходе информационные потоки протекают в одном направлении: от исходных данных к результату.

Объекты именуются именами существительными, глаголами, именами прилагательными.

Имена методов образуются от глаголов.

Объектно-ориентированный подход

Размещенный в отдельном файле набор связанных процедур вместе с данными, которые они обрабатывают, называют программной единицей (Unit). Слово «Unit» переводят как модуль. Так возник термин «модульное программирование». В модульном программировании акцент сместился от *проектирования процедур* в сторону организации *данных*. Это явилось отражением факта увеличения размеров ПО. *Парадигма*: реши, какие требуются модули; разбей программу так, чтобы скрыть данные в модулях.

В объектно-ориентированном программировании (ООП) абстракция данных является *фундаментальным* аспектом качественного проектирования. *Парадигма:* программа представляется набором объектов, которые, взаимодействуя друг с другом посредством сообщений, меняют себя и окружающие объекты. Принцип модульного программирования используется и в ООП. В одном Unit описывается либо один класс, либо несколько классов, наследуемых от одного класса. Механизмы Unit реализуют сокрытие информации. Объектно-ориентированный подход характеризуется разнонаправленностью и разнородностью информационных потоков.

Мнемоника имен в ПО

Методика составления имен (идентификаторов) носит рекомендательный характер. После адаптационной переработки методика может стать составной частью стандарта проекта.

Рекомендации:

— соответствовать назначению (из имени должно однозначно следовать его назначение и, наоборот, из назначения — его имя);

— обладать узнаваемостью (это свойство имени позволяет улучшить читаемость исходных текстов программ);

— обеспечивать запоминаемость (имя необходимо легко запомнить для того, чтобы каждый раз не возвращаться к документации или к тексту программы, в котором это имя определено);

— быть краткими (слишком длинные имена не запоминаемы и, несмотря на повышенную узнаваемость по сравнению с короткими именами, усложняют чтение исходного текста программы);

— обладать уникальностью (как следствие, «соответствовать назначению»: имена должны составляться таким образом, чтобы во всей системе не было двух одинаковых глобальных имен).

Мнемоника имен в ПО - продолжение

Имена — это очень важная часть программы, нельзя преуменьшать значимость имен.

Квалифицированные программисты тратят некоторое время на вычищение своего кода для простоты его последующего использования. Ясность кода определяется ясностью имен данных, понятностью последовательности действий, ясностью имен процедур и объектов.

Дополнительные рекомендации по составлению имен:

- не начинайте и не заканчивайте имена символом подчеркивания;
- не используйте имена, состоящие только из строчных букв (исключения составляют имена констант и макроопределений);
- не следует в одной и той же программе использовать имена, различающиеся лишь написанием букв — строчной или прописной;
- в зависимости от языка программирования можно разделять части имен символом подчеркивания, написанием с большой буквы;
- использование строчных букв в начале каждого слова имени затрудняет трансляцию текста с одного языка программирования на другой язык.

Мнемоника имен в ПО - примеры

Пример 1. is_passport_privilege_valid — плохое имя. Префикс в глобальном имени излишен.

Пример 2. passport_privilege_valid — хорошее имя.

Пример 3. i_order — хорошее имя для поля в таблице, указывающее на порядок чего-либо. Префикс "i" характеризует тип целый.

Пример 4. passport_is_login_valid — плохое имя. Слово "is" — лишнее.

Пример 5. passport_login_valid — хорошее имя. Есть подсистема управления аккаунтами passport. В ней есть часть, которая занимается управлением логинами пользователей passport_login. Функция passport_login_valid проверяет, является ли "логин" правильным.

Пример 1. change_user_password — плохое имя. Первое слово — глагол и оно не может обозначать имя объекта. В таком случае в двух разных местах может потребоваться ввести две функции (изменить пароль для доступа к account и изменить пароль для входа в чат) с одинаковыми именами, что противоречит пункту "соответствовать назначению" общих требований к именам.

Пример 2. passport_password_change или passport_password_change — хорошие имена.

Типовые элементы в программировании

Первыми укрупненными типовыми элементами были подпрограммы. До сих пор библиотеки математических методов обычно поставляются в виде набора подпрограмм. Пример - редактор текстов характеризуется десятком подпрограмм.

Объектно-ориентированные языки программирования дали четыре новых механизма использования кубиков:

1) механизм классов, порождающих при выполнении любое количество однотипных объектов, например, ряд однотипных кнопок;

2) возможность тиражирования объектов от породившей программы во все новые программы;

3) динамически линкуемые библиотеки с порождающими объекты классами;

4) механизм сборки программ из "кубиков" — объектов в процессе их выполнения.

Первый механизм облегчил развитие систем визуального программирования. Отбор мышкой стандартных "кубиков".

Второй механизм привел к возникновению объектно-ориентир. СУБД, поставляющих данные и код, обрабатывающий эти данные.

Типовые элементы в программировании - продолжение

Третий и четвертый механизмы позволили попытаться строить гибкие программы, обладающие свойством возможного развития при изменении условий их эксплуатации. Эволюционное программирование недостаточно для обеспечения гибкости программ. Согласно третьему механизму возникли COM-технологии. На основе четвертого механизма из базы готовых "кубиков П-объектов" создаются все новые "кубики" и из них новые программы.

В теории программирования проблема "кубиков" остается важнейшей проблемой, поэтапное решение которой уже позволило создавать самые большие по количеству составных частей искусственные системы — программы. Второй аспект проблемы "кубиков" — удешевление программных разработок за счет повторного использования во все новых программах частей, созданных в предшествующих разработках.

Проектирование — высокоинтеллектуальный процесс. В процессе выполнения проекта предусматриваются отдельные моменты времени, Согласно ГОСТ возможны следующие стадии разработки: ТЗ; ЭП; ТП; РП; внедрение. Возможны и нестандартные этапы и стадии.