**Дополнительное задание**

**На тему**: «Методология информационного моделирования Мартина»

**ВВЕДЕНИЕ**

На рубеже 80-х г.г. Дж. Мартин выступил с проектом, названным новой информационной технологией (НИТ). Необходимость НИТ обуславливалась тем, что длительность традиционных методов разработки информационных систем превосходила время безусловного морального старения их спецификаций. С момента, когда были сформированы и утверждены требования к будущей системе и до начала ее опытной эксплуатации эти требования безнадежно устаревали. Для выхода из этой ситуации было предложено участие в процессе создания и проектирования системы будущих пользователей.

Используя языки программирования сверхвысокого уровня, специальные языки запросов к базам данных, и специальные языки для формальных спецификаций, пользователь, согласно замыслу автора НИТ, должен был реализовать прототип будущей системы, который предусматривал все нужные функции, но не удовлетворял требованиям эффективности использования ресурсов. Это реализовывалось профессиональными программистами, которые формировали ПО будущей системы.

Первый шаг к НИТ был сделан, когда ПЭВМ стали применяться при решении практических задач, таких как, управление деятельностью предприятий, планирование, информационный поиск в больших массивах информации, т.е. с появлением качественно нового типа - ИС. Сложился также определенный комплекс требований для ПЭВМ.

**Методология информационного моделирования Мартина**

Методология Мартина предоставляет общую стратегию разработки информационных систем, фокусирующую внимание на стратегическом планировании и бизнес-процессах. В то же время она является и инженерным подходом к разработке ПО, т.к. обеспечивает нисходящую пошаговую процедуру построения информационной системы (позволяя при этом работать с неиерархическими структурами данных). Подход Мартина базируется на двух концепциях:

* послойного целостного подхода к разработке интегрированных приложений, базирующегося на стратегическом плане развития информационных систем;
* первоначальной направленности на моделирование данных, а затем на функциональное моделирование

Основные этапы подхода Мартина приведены на рис. 1.

***Рис. 1****. Основные этапы подхода Мартина*

1. Этап *стратегического информационного планирования* начинается с построения стратегического плана для бизнес-системы, включающего цели и стратегии их достижения. Далее строится модель предметной области, отражающая существующую специфику и определяющая основные бизнес-процессы и организационную структуру бизнес-системы, а также определяется порядок разработки информационной системы. При моделировании используются диаграммы декомпозиции (иерархические древовидные структурные диаграммы) и диаграммы "сущность-связь" для представления основных бизнес-процессов и структур данных, соответственно.
2. На этапе *анализа* основные бизнес-процессы, разработанные на этапе 1), используются для разбиения общей задачи на частные, при этом основное внимание уделяется определению информационной и функциональной моделей для частных задач. При этом диаграммы "сущность-связь" трансформируются в нормализованную модель данных, а диаграммы декомпозиции распределяются по подзадачам. Для представления процессов служат DFD, диаграммы зависимости данных (диалект DFD) и диаграммы декомпозиции, а для соотнесения данных и процессов, в которых эти данные используются, применяются матрицы "сущность/процесс".
3. На этапе *логического проектирования* IE становится аналогична SE для разработки ПО. Базой для проектирования являются процессы, разработанные на этапе анализа. Используя методики нисходящей функциональной декомпозиции, проектируются спецификации обработки в процессах и их логические структуры данных. При этом используются диаграммы структуры данных (диалект ERD), определяющие типы сущностей, их атрибуты и связи, диаграммы декомпозиции и диаграммы деятельности (вид миниспецификации), детализирующие логику процессов. Для согласования требований пользователя создаются прототипы пользовательских интерфейсов с помощью схем экранов/отчетов.

На этапе *физического проектирования* и реализации производится преобразование логической модели ИС в физическую и ее реализация.

Для полного представления о программном продукте необходима также текстовая информация описательного характера.

Еще большую значимость информационные модели и структуры данных имеют для информационного моделирования предметной области*,* в основе которого положение об определяющей роли данных при проектировании алгоритмов и программ. Подход появился в условиях развития программных средств организации хранения и обработки данных - СУБД.

Основоположников **информационной инженерии** - Дж. Мартин - выделяет следующие составляющие данного подхода:

* информационный анализ предметных областей *(бизнес - областей);*
* информационное моделирование - построение комплекса взаимосвязанных моделей данных;
* системное проектирование функций обработки данных;
* детальное конструирование процедур обработки данных.

Первоначально строятся информационные модели различных уровней представления:

* информационно-логическая модель, не зависящая от средств программной реализации хранения и обработки данных, отражающая интегрированные структуры данных предметной области;
* даталогические модели, ориентированные на среду хранения и обработки данных.

Даталогические модели имеют логический и физический уровни представления. *Физический уровень* соответствует организации хранения данных в памяти компьютера. *Логический уровень* данных применительно к СУБД реализован в виде:

* концептуальной модели базы данных - интегрированные структуры данных под управлением СУБД;
* внешних моделей данных - подмножество структур данных для реализации приложений.

Средствами структур данных моделируются функции предметной области, прослеживается взаимосвязь функций обработки, уточняется состав входной и выходной информации, логика преобразования входных структур данных в выходные. Алгоритм обработки данных можно представить как совокупность процедур преобразований структур данных в соответствии с внешними моделями данных.

Выбор средств реализации базы данных определяет вид даталогические моделей и, следовательно, алгоритмы преобразования данных. В большинстве случаев используется реляционное представление данных базы данных и соответствующие реляционные языки для программирования (манипулирования) обработки данных СУБД и реализации алгоритмов обработки. Данный подход использован во многих CASE-технологиях.

*Объектно-ориентированный подход* к проектированию программных продуктов основан на:

* выделении классов объектов;
* установлении характерных *свойств* объектов и *методов* их обработки;
* создании иерархии *классов,* наследовании свойств объектов и методов их обработки.

Каждый объект объединяет как данные, так и программу обработки этих данных и относится к определенному классу. С помощью класса один и тот же программный код можно использовать для относящихся к нему различных объектов.

*Объектный подход* при разработке алгоритмов и программ предполагает:

* объектно-ориентированный анализ предметной области;
* объектно-ориентированное проектирование.

*Объектно-ориентированный анализ* - анализ предметной области и выделение объектов, определение свойств и методов обработки объектов, установление их взаимосвязей.

*Объектно-ориентированное проектирование* соединяет процесс объектной декомпозиции и представления с использованием моделей данных проектируемой системы на логическом и физическом уровнях, в статике и динамике.

Для проектирования программных продуктов разработаны объектно-ориентированные технологии, которые включают в себя специализированные языки программирования и инструментальные средства разработки пользовательского интерфейса.

Традиционные подходы к разработке программных продуктов всегда подчеркивали различия между *данными* и *процессами* их обработки. Так, технологии, ориентированные на информационное моделирование, сначала специфицируют данные, а затем описывают процессы, использующие эти данные. Технологии структурного подхода ориентированы, в первую очередь, на процессы обработки данных с последующим установлением необходимых для этого данных и организации информационных потоков между связанными процессами.

Объектно-ориентированная технология разработки программных продуктов объединяет данные и процессы в логические сущности - *объекты,* которые имеют способность наследовать характеристики (методы и данные) одного или более объектов, обеспечивая тем самым повторное использование программного кода. Это приводит к значительному уменьшению затрат на создание программных продуктов, повышает эффективность жизненного цикла программных продуктов (сокращается длительность фазы разработки).При выполнении программы объекту посылается сообщение, которое инициирует обработку данных объекта.

**Пакет Visible Analyst Workbench (Visible Systems)**

Visible Analyst Workbench представляет собой сетевое многопользовательское средство проектирования информационных систем, базирующееся на репозитарии, хранимом на сервере SQLBase, Oracle или Informix. Пакет основан на методологии Мартина и поддерживает следующие диаграммные техники:

* диаграммы функциональной декомпозиции
* диаграммы потоков данных в нотациях Йодана и Гейна-Сарсона
* диаграммы “сущность-связь”
* структурные карты в нотации Константайна.

Пакет обеспечивает генерацию схем БД для вышеперечисленных СУБД и поддерживает технологию FRE. Имеется возможность экспорта проектов в системы SQLWindows, PowerBuilder и Uniface.

К достоинствам пакета может быть отнесено наличие развитых средств верификации проекта, и прежде всего возможностей вертикального и горизонтального балансирования диаграмм. Так функциональная и информационная модели сильно коррелированы, что позволяет избавиться от лишних объектов моделей.

**ВЫВОД**

Деятельность Дж. Мартина принесла огромное значение. В середине 80-х годов на фирме ***Du Pont*** был формализован подход к разработке информационных систем, использующий последовательный выпуск прототипов системы, жесткие ограничения по времени и вовлечение конечных пользователей системы в ее разработку. После публикации в 1991г. книги Дж. Мартина «Rapid Application Development» (быстрая разработка приложений) этот подход получил широкую известность как RAD-технология. Технология RAD более всего подходит при разработке интерактивных приложений, в которых функциональные возможности реализуются на уровне пользовательского интерфейса. Четко определяется группа пользователей такого приложения. Большие приложения подвергаются разбиению на более мелкие функциональные компоненты.

В основе RAD-технологии лежали следующие положения:

* Пользователи активно участвуют в разработке системы от начала обследования предметной области до внедрения приложения. Несколько представителей пользователей включаются непосредственно в команду разработчиков. Представители остальных периодически участвуют в сессиях по пересмотру результатов работы, что позволяет устранять недопонимание между разработчиками и будущими пользователями системы.
* Не требуется полного определения требований к системе, детали могут быть добавлены в ходе разработки. Это позволяет сократить длительность этапа анализа и дает разработчикам определенную свободу в определении требований низкого уровня в ходе построения прототипов системы и их обсуждения с конечными пользователями. Выявленные в процессе разработки дополнительные требования ранжируются по важности. В условиях жестких временных ограничений менее приоритетные требования могут быть опущены.
* Система разрабатывается небольшой командой из 4—б человек, включая 1—2 представителей пользователей. Члены команды должны быть уполномочены принимать необходимые решения. Во время разработки проекта состав команды практически не меняется, что позволяет уменьшить необходимость в промежуточной документации.
* Разработка ведется итерациями при тесном вовлечении пользователей на протяжении всего цикла разработки системы. Основную роль играет правило 80/20, которое гласит, что 80 % работы может быть выполнено за 20 % времени, затрачиваемого на всю работу. Это означает, что нет смысла прикладывать усилия на тонкую настройку системы, когда еще до конца не определены основные требования к ней. Каждый шаг должен быть закончен настолько, насколько это необходимо для выполнения следующей работы.
* На срок выпуска каждого прототипа накладываются жесткие ограничения по времени, превышать которые не разрешается. По истечении установленного срока прототип предъявляется заказчику для обсуждения.
* Тестирование выполняется постепенно на протяжении всего жизненного цикла системы.
* Разрабатываемая система разбивается на части, которые бригада из 4-6 человек способна разработать за З—б месяцев. При наличии нескольких команд возможна параллельная разработка системы. В этом случае проводится более тщательный анализ прикладной области.

Одним из важных положений является необходимость сотрудничества между всеми участниками проекта. Сильная сторона подхода RAD состоит в том, что он позволяет непосредственно в ходе разработки быстро выявлять и уточнять, необходимый набор функциональных возможностей .

**ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА:**

1. Мартин Дж. Организация баз данных в вычислительных системах. М.: Мир, 1980, 662 с.

2. Мартин Дж. Планирование развития автоматизированных систем. - М.: "Финансы и статистика", 1984.

3. http://www.interface.ru/fset.asp?Url=/case/defs8.htm

4. http://www.techno.edu.ru:16001/db/msg/15641.html