Проектирование информационных систем всегда начинается с определения цели проекта. Основная задача любого успешного проекта заключается в том, чтобы на момент запуска системы и в течение всего времени ее эксплуатации можно было обеспечить:

требуемую функциональность системы и степень адаптации к изменяющимся условиям ее функционирования;

требуемую пропускную способность системы;

требуемое время реакции системы на запрос;

безотказную работу системы в требуемом режиме, иными словами - готовность и доступность системы для обработки запросов пользователей;

простоту эксплуатации и поддержки системы;

необходимую безопасность.

Производительность является главным фактором, определяющим эффективность системы. Хорошее проектное решение служит основой высокопроизводительной системы.

Проектирование информационных систем охватывает три основные области:

проектирование объектов данных, которые будут реализованы в базе данных;

проектирование программ, экранных форм, отчетов, которые будут обеспечивать выполнение запросов к данным;

учет конкретной среды или технологии, а именно: топологии сети, конфигурации аппаратных средств, используемой архитектуры (файл-сервер или клиент-сервер), параллельной обработки, распределенной обработки данных и т.п.

В реальных условиях проектирование - это поиск способа, который удовлетворяет требованиям функциональности системы средствами имеющихся технологий с учетом заданных ограничений.

К любому проекту предъявляется ряд абсолютных требований, например максимальное время разработки проекта, максимальные денежные вложения в проект и т.д. Одна из сложностей проектирования состоит в том, что оно не является такой структурированной задачей, как анализ требований к проекту или реализация того или иного проектного решения.

Считается, что сложную систему невозможно описать в принципе. Это, в частности, касается систем управления предприятием. Одним из основных аргументов является изменение условий функционирования системы, например директивное изменение тех или иных потоков информации новым руководством. Еще один аргумент - объемы технического задания, которые для крупного проекта могут составлять сотни страниц, в то время как технический проект может содержать ошибки. Возникает вопрос: а может, лучше вообще не проводить обследования и не делать никакого технического проекта, а писать систему "с чистого листа" в надежде на то, что произойдет некое чудесное совпадение желания заказчика с тем, что написали программисты, а также на то, что все это будет стабильно работать?

Если разобраться, то так ли уж непредсказуемо развитие системы и действительно ли получить информацию о ней невозможно? Вероятно, представление о системе в целом и о предполагаемых (руководством) путях ее развития можно получить посредством семинаров. После этого разбить сложную систему на более простые компоненты, упростить связи между компонентами, предусмотреть независимость компонентов и описать интерфейсы между ними (чтобы изменение одного компонента автоматически не влекло за собой существенного изменения другого компонента), а также возможности расширения системы и "заглушки" для нереализуемых в той или иной версии системы функций. Исходя из подобных элементарных соображений описание того, что предполагается реализовать в информационной системе, уже не кажется столь нереальным. Можно придерживаться классических подходов к разработке информационных систем, один из которых - схема "водопада" (рис.1) - описан ниже. Кратко будут рассмотрены и некоторые другие подходы к разработке информационных систем, где использование элементов, описанных в схеме "водопада", также допустимо. Какого подхода из описываемых ниже придерживаться (и есть ли смысл придумывать собственный подход) - в какой-то мере дело вкуса и обстоятельств.

Рис.1. Cхема "водопада"

Жизненный цикл программного обеспечения представляет собой модель его создания и использования. Модель отражает его различные состояния, начиная с момента возникновения необходимости в данном ПО и заканчивая моментом его полного выхода из употребления у всех пользователей. Известны следующие модели жизненного цикла:

Каскадная модель. Переход на следующий этап означает полное завершение работ на предыдущем этапе.

Поэтапная модель с промежуточным контролем. Разработка ПО ведется итерациями с циклами обратной связи между этапами. Межэтапные корректировки позволяют уменьшить трудоемкость процесса разработки по сравнению с каскадной моделью; время жизни каждого из этапов растягивается на весь период разработки.

Спиральная модель. Особое внимание уделяется начальным этапам разработки - выработке стратегии, анализу и проектированию, где реализуемость тех или иных технических решений проверяется и обосновывается посредством создания прототипов (макетирования). Каждый виток спирали предполагает создание некой версии продукта или какого-либо его компонента, при этом уточняются характеристики и цели проекта, определяется его качество и планируются работы следующего витка спирали.

Ниже мы рассмотрим некоторые схемы разработки проекта.

"**Водопад" - схема разработки проекта.**

Очень часто проектирование описывают как отдельный этап разработки проекта между анализом и разработкой. Однако в действительности четкого деления этапов разработки проекта нет - проектирование, как правило, не имеет явно выраженного начала и окончания и часто продолжается на этапах тестирования и реализации. Говоря об этапе тестирования, также следует отметить, что и этап анализа, и этап проектирования содержат элементы работы тестеров, например для получения экспериментального обоснования выбора того или иного решения, а также для оценки критериев качества получаемой системы. На этапе эксплуатации уместен разговор и о сопровождении системы.

Ниже мы рассмотрим каждый из этапов, подробнее остановившись на этапе проектирования.

Стратегия.

Определение стратегии предполагает обследование системы. Основная задача обследования - оценка реального объема проекта, его целей и задач, а также получение определений сущностей и функций на высоком уровне.

На этом этапе привлекаются высококвалифицированные бизнес-аналитики, которые имеют постоянный доступ к руководству фирмы; этап предполагает тесное взаимодействие с основными пользователями системы и бизнес-экспертами. Основная задача взаимодействия - получить как можно более полную информацию о системе (полное и однозначное понимание требований заказчика) и передать данную информацию в формализованном виде системным аналитикам для последующего проведения этапа анализа. Как правило, информация о системе может быть получена в результате бесед или семинаров с руководством, экспертами и пользователями. Таким образом определяются суть данного бизнеса, перспективы его развития и требования к системе.

По завершении основной стадии обследования системы технические специалисты формируют вероятные технические подходы и приблизительно рассчитывают затраты на аппаратное обеспечение, закупаемое программное обеспечение и разработку нового программного обеспечения (что, собственно, и предполагается проектом).

Результатом этапа определения стратегии является документ, где четко сформулировано, что получит заказчик, если согласится финансировать проект; когда он получит готовый продукт (график выполнения работ); сколько это будет стоить (для крупных проектов должен быть составлен график финансирования на разных этапах работ). В документе должны быть отражены не только затраты, но и выгода, например время окупаемости проекта, ожидаемый экономический эффект (если его удается оценить).

В документе обязательно должны быть описаны:

ограничения, риски, критические факторы, влияющие на успешность проекта, например время реакции системы на запрос является заданным ограничением, а не желательным фактором;

совокупность условий, при которых предполагается эксплуатировать будущую систему: архитектура системы, аппаратные и программные ресурсы, предоставляемые системе, внешние условия ее функционирования, состав людей и работ, которые обеспечивают бесперебойное функционирование системы;

сроки завершения отдельных этапов, форма сдачи работ, ресурсы, привлекаемые в процессе разработки проекта, меры по защите информации;

описание выполняемых системой функций;

будущие требования к системе в случае ее развития, например возможность работы пользователя с системой с помощью Интернета и т.п.;

сущности, необходимые для выполнения функций системы;

интерфейсы и распределение функций между человеком и системой;

требования к программным и информационным компонентам ПО, требования к СУБД (если проект предполагается реализовывать для нескольких СУБД, то требования к каждой из них, или общие требования к абстрактной СУБД и список рекомендуемых для данного проекта СУБД, которые удовлетворяют заданным условиям);

что не будет реализовано в рамках проекта.

Выполненная на данном этапе работа позволяет ответить на вопрос, стоит ли продолжать данный проект и какие требования заказчика могут быть удовлетворены при тех или иных условиях. Может оказаться, что проект продолжать не имеет смысла, например из-за того, что те или иные требования не могут быть удовлетворены по каким-то объективным причинам. Если принимается решение о продолжении проекта, то для проведения следующего этапа анализа уже имеются представление об объеме проекта и смета затрат.

Следует отметить, что и на этапе выбора стратегии, и на этапе анализа, и при проектировании независимо от метода, применяемого при разработке проекта, всегда следует классифицировать планируемые функции системы по степени важности. Один из возможных форматов представления такой классификации - MoSCoW - предложен в Clegg, Dai and Richard Barker, Case Method Fast-track: A RAD Approach, Adison-Wesley, 1994.

Эта аббревиатура расшифровывается так: Must have - необходимые функции; Should have - желательные функции; Could have - возможные функции; Won't have - отсутствующие функции.

Функции первой категории обеспечивают критичные для успешной работы системы возможности.

Реализация функций второй и третьей категорий ограничивается временными и финансовыми рамками: разрабатываем то, что необходимо, а также максимально возможное в порядке приоритета число функций второй и третьей категорий.

Последняя категория функций особенно важна, поскольку необходимо четко представлять границы проекта и набор функций, которые будут отсутствовать в системе.

В начало. Анализ.

Этап анализа предполагает подробное исследование бизнес-процессов (функций, определенных на этапе выбора стратегии) и информации, необходимой для их выполнения (сущностей, их атрибутов и связей (отношений)). На этом этапе создается информационная модель, а на следующем за ним этапе проектирования - модель данных.

Вся информация о системе, собранная на этапе определения стратегии, формализуется и уточняется на этапе анализа. Особое внимание следует уделить полноте переданной информации, анализу информации на предмет отсутствия противоречий, а также поиску неиспользуемой вообще или дублирующейся информации. Как правило, заказчик не сразу формирует требования к системе в целом, а формулирует требования к отдельным ее компонентам. Уделите внимание согласованности этих компонентов.

Аналитики собирают и фиксируют информацию в двух взаимосвязанных формах:

функции - информация о событиях и процессах, которые происходят в бизнесе;

сущности - информация о вещах, имеющих значение для организации и о которых что-то известно.

Двумя классическими результатами анализа являются:

иерархия функций, которая разбивает процесс обработки на составные части (что делается и из чего это состоит);

модель "сущность-связь" (Entry Relationship model, ER-модель), которая описывает сущности, их атрибуты и связи (отношения) между ними.

Эти результаты являются необходимыми, но не достаточными. К достаточным результатам следует отнести диаграммы потоков данных и диаграммы жизненных циклов сущностей. Довольно часто ошибки анализа возникают при попытке показать жизненный цикл сущности на диаграмме ER.

Ниже мы рассмотрим три наиболее часто применяемые методологии структурного анализа:

диаграммы "сущность-связь" (Entity-Relationship Diagrams, ERD), которые служат для формализации информации о сущностях и их отношениях;

диаграммы потоков данных (Data Flow Diagrams, DFD), которые служат для формализации представления функций системы;

диаграммы переходов состояний (State Transition Diagrams, STD), которые отражают поведение системы, зависящее от времени; диаграммы жизненных циклов сущностей относятся именно к этому классу диаграмм.

Методы проектирования информационных систем

Индустрия разработки автоматизированных информационных систем управления родилась в 50-х - 60-х годах и к концу века приобрела вполне законченные формы. Материалы данного руководства являются обобщением цикла лекций по Автоматизированным Банковским Системам (АБС) и Автоматизированным системам управления конструкторско-технологическим проектированием (АСУ КТП), читаемым в МГТУ им. Н.Э. Баумана. Не смотря на имеющиеся различия в реализации функциональных модулей данных систем, общие подходы к их разработки во многом схожи, что позволило нам объединить вопросы их проектирования в рамках одного издания.

На рынке автоматизированных систем для крупных корпораций и финансово-промышленных групп на сегодня можно выделить два основных субъекта: это ранок автоматизированных банковских систем (АБС) и рынок корпоративных информационных систем промышленных предприятий. Не смотря на сильную взаимосвязь этих двух рынков систем автоматизации, предлагаемые на них решения пока еще не достаточно интегрированы между собой, чего следует ожидать в недалеком будущем.

В дальнейшем под Автоматизированной Банковской Системой (АБС) будем понимать комплекс аппаратно-программных средств реализующих мультивалютную информационную систему, обеспечивающую современные финансовые и управленческие технологии в режиме реального времени при транзакционной обработке данных.

Под Автоматизированной Информационной Системой промышленного предприятия (АСУ КТП) будем понимать комплекс аппаратно-программных средств реализующих мультикомпонентную информационную систему, обеспечивающую современное управление процессами принятия решений, проектирования, производства и сбыта в режиме реального времени при транзакционной обработке данных.

Как вы видите, оба определения достаточно схожи. На сегодня существования нескольких методов построения автоматизированных информационных систем (АИС), среди которых можно выделить следующие:

Метод "снизу-вверх".

Менталитет российских программистов сформировался именно в крупных вычислительных центрах (ВЦ), основной целью которых было не создание тиражируемых продуктов, а обслуживание сотрудников конкретного учреждения. Этот подход во многом сохранялся и при автоматизации и сегодня. В условиях постоянно изменяющихся законодательства, правил ведения производственной, финансово-хозяйственной деятельности и бухгалтерского учета руководителю удобно иметь рядом посредника между спущенной сверху новой инструкцией и компьютером. С другой стороны, программистов, зараженных "вирусом самодеятельности", оказалось предостаточно, тем более что за такую работу предлагалось вполне приличное вознаграждение.

Создавая свои отделы и управления автоматизации, предприятия и банки пытались обустроиться своими силами. Однако периодическое "перетряхивание" инструкций, сложности, связанные с разными представлениями пользователей об одних и тех же данных, непрерывная работа программистов по удовлетворению все новых и новых пожеланий отдельных работников и как следствие - недовольство руководителей своими программистами несколько остудило пыл как тех, так и других. Итак, первый подход сводился к проектированию "**снизу-вверх".** В этом случае, при наличии квалифицированного штата программистов, вполне сносно были автоматизированы отдельные, важные с точки зрения руководства рабочие места. Общая же картина "автоматизированного предприятия" просматривалась недостаточно хорошо, особенно в перспективе.

Метод "сверху-вниз".

Быстрый рост числа акционерных и частных предприятий и банков позволил некоторым компаниям увидеть здесь будущий рынок и инвестировать средства в создание программного аппарата для этого растущего рынка. Из всего спектра проблем разработчики выделили наиболее заметные: автоматизацию ведения бухгалтерского аналитического учета и технологических процессов (для банков это в основном - расчетно-кассовое обслуживание, для промышленных предприятий - автоматизация процессов проектирования и производства, имеется в виду не конкретных станков и т.п., а информационных потоков). Учитывая тот факт, что ядром АИС безусловно является аппарат, обеспечивающий автоматизированное ведение аналитического учета, большинство фирм начали с детальной проработки данной проблемы. Системы были спроектированы "сверху", т.е. в предположении что одна программа должна удовлетворять потребности всех пользователей.

Сама идея использования "одной программы для всех" резко ограничила возможности разработчиков в структуре информационных множеств базы данных, использовании вариантов экранных форм, алгоритмов расчета и, следовательно, лишила возможности принципиально расширить круг решаемых задач - автоматизировать повседневную деятельность каждого работника. Заложенные "сверху" жесткие рамки ("общие для всех") ограничивали возможности таких систем по ведению глубокого, часто специфического аналитического и производственно - технологического учета. Работники проводили эту работу вручную, а результаты вводили в компьютер. При этом интерфейс каждого рабочего места не мог быть определен функциями, возложенными на пользователя, и принятой технологией работы. Стало очевидно, что для успешной реализации задачи полной автоматизации банка следует изменить идеологию построения АИС.

Принципы "дуализма" и многокомпонентности.

Развитие банковских структур и промышленных предприятий, увеличение числа филиалов, рост количества клиентов, необходимость повышения качества обслуживания предъявляли к автоматизированным системам новые требования. Новый подход к проектированию АИС заключается в сбалансированном сочетании двух предыдущих. В первую очередь это относилось к идеологии построения ядра системы: "Автоматизированная бухгалтерия - аналитический учет".

Для банковских структур это дало: с одной стороны, в ядре системы сохранялась возможность работы "от лицевого счета", с автоматическим формированием соответствующих бухгалтерских проводок, с другой стороны, отменялись жесткие требования работы только с лицевыми счетами. Появилась возможность ведения бухгалтерского учета по балансовым счетам любого порядка без углубления до уровня лицевых счетов клиентов. При этом ведение аналитического учета по лицевым счетам клиентов опускалось на уровень специализированного программного обеспечения (СПО), установленного на рабочих местах банковских работников (контролеров, кредитных бухгалтеров, инспекторов и т.д.). Таким образом, принципиальное отличие нового подхода к созданию АБС заключается в идее распределения плана счетов по уровням экспертизы. При этом и сам справочник плана счетов с соответствующими описаниями, и информационное множество клиентов проектировались по принципу распределенной базы данных. Результатом этого явилось:

формирование всех необходимых бухгалтерских проводок, уже агрегированных по балансовым счетам, и автоматическая их передача в базу данных "Автоматизированной бухгалтерии";

реализация специфических требований каждого банковского работника, в том числе по формированию произвольных отчетов и справок, мемориальных ордеров, операционных дневников; выполнение любых вспомогательных и технологических расчетов и пр.

С использованием гибкой системы настроек СПО (компонентов АБС) появилась реальная возможность адаптации программного аппарата к практически любым условиям и различным требованиям инструктивных материалов и правилам работы, принятым либо в вышестоящей организации, либо в данном банковском учреждении. Кроме того, при многокомпонентной схеме организации АБС при проведении модернизации одного из компонентов центральная часть (ядро) АБС и другие ее компоненты не затрагивались, что значительно повышало надежность, продолжительность жизни автоматизированной системы и обеспечивало наиболее полное выполнение требуемых функций.

Двойственный подход к формированию ежедневного баланса лег в основу т. н. "принципа дуализма" - одного из важных принципов построения современных банковских систем. Реализация принципа дуализма неизбежно требовала построения АБС нового поколения в виде программных модулей, органически связанных между собой, но в то же время способных работать и автономно.

Задача проектирования АИС промышленных предприятий более сложна, т.к. характер обрабатываемой информации еще более разнороден и сложно формализуем. Однако и здесь можно выделить основную модель работы - это работа "от кода проекта". В общем случае код проекта представляет собой аналог (функциональный) лицевого счета, он имеет определенную разрядность, порядок (т.е. конкретная группа цифро-буквенного обозначения характеризует деталь, сборочную единицу, изделие и их уровень взаимосвязи). Причем конкретная часть кода характеризует технологические, конструкторские, финансовые и др. документы. Все это регламентируется соответствующими ГОСТами (аналог инструкций ЦБ для банков), поэтому может быть формализовано. При этом модульный подход к реализации АИС в этом случае еще более важен.

Двойственный подход к формированию ежедневного производственного плана лег в основу т. н. "принципа дуализма" для АИС промышленных предприятий. Реализация принципа дуализма неизбежно также требовала построения АИС предприятий нового поколения в виде программных модулей, органически связанных между собой, но в то же время способных работать и автономно.

Такая многокомпонентная система обеспечивала соблюдение основополагающего принципа построения автоматизированных информационных систем - отсутствия дублирования ввода исходных данных. Информация по операциям, проведенным с применением одного из компонентов системы, могла быть использована любым другим ее компонентом. Модульность построения АИС нового поколения и принцип одноразового ввода дают возможность гибко варьировать конфигурацией этих систем. Так, в банках, имеющих разветвленную филиальную сеть и не передающих данные в режиме реального времени, установка всего СПО во всех филиалах не всегда экономически оправдано. В этих случаях возможна эксплуатация в филиалах ПО общего назначения, предназначенного для первичного ввода информации и последующей автоматизированной обработки данных в СПО, установленном в головном офисе банка. Такая структура дает возможность органически включить в АБС нового поколения компонент для создания хранилища данных, разделяя системы оперативного действия и системы поддержки принятия решения.

Кроме того, одно из достоинств **принципа многокомпонентности**, являющегося базовым при создании АИС нового поколения, состоит в возможности их поэтапного внедрения. На первом этапе внедрения устанавливаются (или заменяются уже устаревшие) компоненты системы на те рабочие места, которые нуждаются в обновлении ПО. На втором этапе происходит развитие системы с подсоединением новых компонентов и отработкой межкомпонентных связей. Возможность применения такой методики внедрения обеспечивает ее достаточно простое тиражирование и адаптацию к местным условиям. ***Таким образом, автоматизированная информационная система нового поколения - это многокомпонентная система с распределенной базой данных по уровням экспертизы.***

Что же заставляет банки разрабатывать предприятия и банки свои АИС собственными силами [1,2]:

**Во-первых**, это конечно относительно низкая стоимость таких разработок (по сравнению с покупными). Как правило, к существующим подразделениям департамента информатизации, таким как: управление эксплуатации, управление эксплуатации вычислительной сети и средств связи, экспертно-аналитическое управление (постановка задач), добавляется лишь новая структура: управление развития и разработки АИС, что, как правило, не влечет за собой больших финансовых затрат.

**Во-вторых**, собственная разработка - это максимальная ориентация на реализацию бизнес - процессов предприятия или банка, его уникальных финансовых и управленческих технологий, складывающихся годами.

**В-третьих**, это позволяет обеспечивать значительно более высокий уровень безопасности и независимости от внешних факторов.

**В-четвертых**, оперативная реакция на изменения правил игры на рынке.

Вместе с тем при собственной разработке необходимо решить целый комплекс организационно-технических задач, которые позволили бы избежать ошибочных решений [1,2]:

**Во-первых,** правильный выбор архитектуры построения вычислительно-коммуникационной сети и ориентация на профессиональные СУБД. По экспертным оценкам собственные разработки АИС в 53% базируются на СУБД Oracle, около 15% на Informix, 22% - другие СУБД.

**Во-вторых**, использование при разработке современного инструментария (CASE средства, эффективные средства разработки: Delphi, Designer2000, Developer2000, SQL-Stations и т.п.).

**В-третьих**, мультизадачная инфраструктура разработки проекта, когда конкретный модуль АИС ведет группа разработчиков с взаимосвязанным перечнем задач, построенная на принципах полной взаимозаменяемости, т.е. функционирование данного модуля АИС и его развитие не связано с одним конкретным разработчиком.

**В-четвертых**, применение эффективных организационно-технических средств по управлению проектом и контролю версий АИС.

Только при соблюдении этих основных положений можно рассчитывать, что собственная разработка окажется конкурентной и эффективной. В противном же случае можно столкнуться с эффектом "неоправданных ожиданий" - это в лучшем случае, а в крайнем случае вообще задуматься о смене АИС. При этом, смена АИС может вызвать как непосредственно смену клиентских модулей и табличной структуры БД, так и потребовать замены серверного и клиентского аппаратного и общесистемного программного обеспечения, включая СУБД, а это дело не дешевое. Поэтому очень важно при выборе варианта реализации АИС сразу решить вопрос о возможностях экспорта/импорта данных в создаваемой системе. При правильном решении данного вопроса смена АИС, если в ней все-таки возникнет необходимость, произойдем практически безболезненно для функциональных подразделений.

В отличие от банковских структур крупные отечественные промышленные предприятия сейчас только подходят к осознанию явной необходимости внедрения и развития корпоративных информационных систем как одной из основных компонент стратегического развития бизнеса. В связи с этим в недалеком будущем можно ожидать расширение рынка корпоративных информационных систем и в последующем его значительно роста. Учитывая тесную интеграцию финансовых и промышленных структур можно полагать, что основой построения корпоративных систем финансово-промышленных групп будут являться, используемые в их финансовых учреждениях, АБС.