**Лекция №1**

**Информационный процесс**

Все процессы в природе сопровождаются **сигналами**. Такие изменения можно наблюдать, измерять или фиксировать, при этом возникают и **регистрируются** новые сигналы, то есть, образуются данные.   
**Данные** –это зарегистрированные сигналы.   
Данные несут в себе **информацию** о событиях, произошедших в материальном мире, поскольку они являются регистрацией сигналов, возникших в результате этих событий. Однако данные **не тождественны**информации. Для того чтобы данные дали информацию необходимо наличие метода обработки данных.   
**Информация** – это продукт взаимодействия данных и адекватных им методов.   
Информация есть обработанные данные, а данные есть зарегистрированные сигналы. Таким образом, информацию можно считать некоторой материальной величиной, которую можно получать, хранить, передавать, обрабатывать, воспроизводить. Все перечисленные возможности работы с информацией являются основными составляющими **информационного процесса**.   
**Информационный процесс** – это любой процесс, в котором присутствует хотя бы один из элементов: прием информации, ее хранение, обработка, передача, воспроизведение.   
Так как понятие «данные» используется на самом низком уровне обработки, то в дальнейшем будем пользоваться только понятием «информация» – мало-мальски обработанные данные.

**Информационная система**

**Информационная система** – это любая система, реализующая или поддерживающая информационный процесс.   
К информационным можно относить любые системы, включающие в себя работу с информацией. В настоящее время основным помощником человека при работе с информацией является компьютер, поэтому именно его мы и будем рассматривать в качестве источника, способа изменения и хранения информационных систем. А в качестве информационных систем будем рассматривать программное обеспечение компьютера.   
В зависимости от предметной области информационные системы могут весьма значительно различаться по своим функциям, архитектуре, реализации. Однако можно выделить ряд свойств, которые являются общими.

* Информационные системы предназначены организации и поддержке информационного процесса, поэтому в основе любой из них лежит среда хранения и доступа к информации.
* Информационные системы ориентированы на конечного пользователя, не об­ладающего высокой квалификацией в области вычислительной техники. По­этому клиентские приложения информационной системы должны обладать простым, удобным, легко осваиваемым интерфейсом.

Таким образом, при разработке информационной системы приходится решать две основные задачи:

* разработка базы данных, предназначенной для хранения информации;
* разработка графического интерфейса пользователя клиентских приложений.

Подавляющее большинство информационных систем работает в режиме диалога с пользователем.   
В наиболее общем случае типовые программные компоненты, входящие в состав информационной системы, реализуют:

* диалоговый ввод-вывод;
* логику диалога;
* прикладную логику обработки данных;
* логику управления данными;
* операции манипулирования файлами и (или) базами данных.

**Классификация информационных систем**

Информационные системы классифицируются по разным признакам. 

**Классификация по масштабу**

По масштабу информационные системы подразделяются на следующие группы: 

* одиночные;
* групповые;
* корпоративные.

Одиночные информационные системы реализуются, как правило, на автономном персональном компьютере (сеть не используется). Такая система может содержать несколько простых приложений, связанных общим информационным фондом, и рассчитана на работу одного пользователя или группы пользователей, разделя­ющих по времени одно рабочее место. Подобные приложения создаются с помо­щью так называемых настольных, или локальных, систем управления базами дан­ных (СУБД). Среди локальных СУБД наиболее известными являются Clarion, Clipper, FoxPro, Paradox, dBase и Microsoft Access. 

**Групповые информационные системы**

Групповые информационные системы ориентированы на коллективное использование информации членами рабочей группы и чаще всего строятся на базе ло­кальной вычислительной сети. При разработке таких приложений используют­ся серверы баз данных (называемые также SQL (Structured Query Language – структурированный язык запросов)-серверами) для рабочих групп. Существует довольно большое количество различных SQL-серверов как коммер­ческих, так и свободно распространяемых. Среди них наиболее известны такие серверы баз данных, как Oracle, DB2, Microsoft SQL Server, InterBase, Sybase, Informix. 

**Корпоративные информационные системы**

Корпоративные информационные системы являются развитием систем для рабо­чих групп, они ориентированы на крупные компании и могут поддерживать тер­риториально разнесенные узлы или сети. В основном они имеют иерархическую структуру из нескольких уровней. Для таких систем характерна архитектура кли­ент-сервер со специализацией серверов или же многоуровневая архитектура. При разработке таких систем могут использоваться те же серверы баз данных, что и при разработке групповых информационных систем. Однако в крупных информа­ционных системах наибольшее распространение получили серверы Oracle, DB2 и Microsoft SQL Server. 

**Классификация по сфере применения**

По сфере применения информационные системы обычно подразделяются на четыре группы: 

* системы обработки транзакций (протоколов);
* системы поддержки принятия решений;
* информационно-справочные системы;
* офисные информационные системы.

**Системы обработки транзакций**, в свою очередь, по оперативности обработки данных разделяются на пакетные информационные системы и оперативные инфор­мационные системы. В информационных системах организационного управления преобладает режим оперативной обработки транзакций (OnLine Transaction Pro­cessing, OLTP) для отражения актуального состояния предметной области в лю­бой момент времени, а пакетная обработка занимает весьма ограниченную часть. Для систем OLTP характерен регулярный (возможно, интенсивный) поток довольно простых транзакций, играющих роль заказов, платежей, запросов и т.п. Важными требованиями для них являются: 

* высокая производительность обработки транзакций;

гарантированная доставка информации при удаленном доступе к БД по телекоммуникациям.   
**Системы поддержки принятия решений** (Decision Support System, DSS) представляют собой другой тип информационных систем, в которых с помощью довольно сложных запросов производится отбор и анализ данных в различных разрезах: временных, географических, по другим показателям.   
Обширный класс **информационно-справочных систем** основан на гипертекстовых документах и мультимедиа. Наибольшее развитие такие информационные системы получили в Интернете.   
Класс **офисных информационных систем** нацелен на перевод бумажных документов в электронный вид, автоматизацию делопроизводства и управление документооборотом. 

**Классификация по способу организации**

По способу организации групповые и корпоративные информационные системы подразделяются на следующие классы : 

* системы на основе архитектуры файл-сервер;
* системы на основе архитектуры клиент-сервер;
* системы на основе многоуровневой архитектуры;
* системы на основе Интернет/интранет-технологий.

В любой информационной системе можно выделить необходимые функциональные компоненты (табл. 1), которые помогают понять ограничения различных архитектур информационных систем. Рассмотрим более подробно особенности вариантов построения информационных приложений.   
Таблица 1.1. Типовые функциональные компоненты информационной системы

|  |  |  |
| --- | --- | --- |
| **Обозначение** | **Наименование** | **Характеристика** |
| **PS** | Presentation Services (средства представления) | Обслуживает пользовательский ввод и отображает то, что сообщает ему компонент логики представления (PL), с использованием соответствующей программной поддержки |
| **PL** | Presentation Logic (логика представления) | Управляет взаимодействием между пользователем и ЭВМ. Обрабатывает действия пользователя при выборе команды в меню, щелчке на кнопке или выборе пункта в списке |
| **BL** | Business Logic (прикладная логика) | Набор правил для принятия решений, вычислений и операций, которые должно выполнить приложение |
| **DL** | Data Logic (логика управления данными) | Операции с базой данных (реализуемые SQL-операторами), которые нужно выполнить для реализации прикладной логики управления данными |
| **DS** | Data Services (операции c базой данных) | Действия СУБД, реализующие логику управления данными, такие как манипулирование данными, определение данных, фиксация или откат транзакций и т. п. СУБД обычно компилирует SQL-предложения |
| **FS** | File Services (файловые операции) | Дисковые операции чтения и записи данных для СУБД и других компонентов. Обычно являются функциями операционной системы (ОС) |

**Архитектура файл-сервер**

В архитектуре файл-сервер сетевое разделение компонентов диалога PS и PL отсутствует, а компьютер используется для функций отображения, что облегчает построение графического интерфейса. Файл-сервер только извлекает данные из файлов, так что дополнительные пользователи и приложения лишь незначительно увеличивают нагрузку на центральный процессор.   
Объектами разработки в файл-серверном приложении являются компоненты приложения, определяющие логику диалога PL, а также логику обработки BL и управления данными DL. Разработанное приложение реализуется либо в виде законченного загрузочного модуля, либо в виде специального кода для интерпретации.   
Однако такая архитектура имеет существенный недостаток: при выполнении некоторых запросов к базе данных клиенту могут передаваться большие объемы данных, загружая сеть и приводя к непредсказуемости времени реакции. Значительный сетевой трафик особенно сказывается при организации удаленного доступа к базам данных на файл-сервере через низкоскоростные каналы связи. Одним из вариантов устранения данного недостатка является удаленное управление файл-серверным приложением в сети. При этом в локальной сети размещается сервер приложений, совмещенный с телекоммуникационным сервером (обычно называемым сервером доступа), в среде которого выполняются обычные файл-серверные приложения. Особенность состоит в том, что диалоговый ввод-вывод поступает от удаленных клиентов через телекоммуникации. Приложения не должны быть слишком сложными, иначе велика веро­ятность перегрузки сервера или же нужна очень мощная платформа для сервера приложений. 

**Архитектура клиент-сервер**

Архитектура клиент-сервер предназначена для разрешения проблем файл-серверных приложений путем разделения компонентов приложения и размещения их там, где они будут функционировать наиболее эффективно. Особенностью архитектуры клиент-сервер является наличие выделенных серверов баз данных, понимающих запросы на языке структурированных запросов (Structured Query Language, SQL) и выполняющих поиск, сортировку и агрегирование информации.   
Отличительная черта серверов БД — наличие справочника данных, в котором записаны структура БД, ограничения целостности данных, форматы и даже серверные процедуры обработки данных по вызову или по событиям в программе. Объектами разработки в таких приложениях, помимо диалога и логики обработки, являются, прежде всего, реляционная модель данных и связанный с ней набор SQL-операторов для типовых запросов к базе данных.   
Большинство конфигураций клиент-сервер использует двухуровневую модель, в которой клиент обращается к услугам сервера. Предполагается, что диалоговые компоненты PS и PL размещаются на клиенте, что позволяет реализовать графический интерфейс. Компоненты управления данными DS и FS размещаются на сервере, а диалог (PS, PL) и логика (BL, DL) — на клиенте. В двухуровневом опре­делении архитектуры клиент-сервер используется именно этот вариант: приложение работает на клиенте, СУБД — на сервере.   
Поскольку эта схема предъявляет наименьшие требования к серверу, она обладает наилучшей масштабируемостью. Однако сложные приложения, активно взаимодействующие с БД, могут жестко загрузить как клиент, так и сеть. Результаты SQL-запроса должны вернуться клиенту для обработки, потому что там реализована логика принятия решения. Такая схема приводит к дополнительному усложнению администрирования приложений, разбросанных по различным клиентским узлам.   
Для сокращения нагрузки на сеть и упрощения администрирования приложений компонент BL можно разместить на сервере. При этом вся логика принятия решений оформляется в виде хранимых процедур и выполняется на сервере БД.   
**Хранимая процедура** — процедура с SQL-операторами для доступа к БД, вызываемая по имени с передачей требуемых параметров и выполняемая на сервере БД. Хранимые процедуры могут компилироваться, что повышает скорость их выполнения и сокращает нагрузку на сервер.   
Хранимые процедуры улучшают целостность приложений и БД, гарантируют актуальность коллективных операций и вычислений. Улучшается сопровождение таких процедур, а также безопасность (нет прямого доступа к данным).   
Создание архитектуры клиент-сервер возможно и на основе многотерминаль­ной системы. В этом случае в многозадачной среде сервера приложений вы­полняются программы пользователей, а клиентские узлы вырождены и пред­ставлены терминалами. Подобная схема информационной системы характерна для Unix.   
Двухуровневые схемы архитектуры клиент-сервер могут привести к некоторым проблемам в сложных информационных приложениях с множеством пользовате­лей и запутанной логикой. Решением этих проблем может стать применение мно­гоуровневой архитектуры. 

**Многоуровневая архитектура**

Многоуровневая архитектура стала развитием архитектуры клиент-сервер и в своей классической форме состоит из трех уровней: 

* нижний уровень представляет собой приложения клиентов, выделенные для выполнения функций и логики представлений PS и PL и имеющие программный интерфейс для вызова приложения на среднем уровне;
* средний уровень представляет собой сервер приложений, на котором выполняется прикладная логика BL и с которого логика обработки данных DL выполняет операции с базой данных DS;
* верхний уровень представляет собой удаленный специализированный сервер базы данных, выделенный для услуг обработки данных DS и файловых операций FS (без использования хранимых процедур).

Подобную концепцию обработки данных пропагандируют, в частности, фирмы Oracle, Sun, Borland и др.   
Трехуровневая архитектура позволяет еще больше сбалансировать нагрузку на разные узлы и сеть, а также способствует специализации инструментов для разработки приложений и устраняет недостатки двухуровневой модели клиент-сервер.   
Централизация логики приложения упрощает администрирование и сопровождение. Четко разделяются платформы и инструменты для реализации интерфейса и прикладной логики, что позволяет с наибольшей отдачей реализовывать их специалистам узкого профиля. Наконец, изменения прикладной логики не затрагивают интерфейса, и наоборот. Но поскольку границы между компонентами PL, BL и DL размыты, прикладная логика может реализовываться на всех трех уровнях. Сервер приложений с помощью монитора транзакций обеспечивает интерфейс с клиентами и другими серверами, может управлять транзакциями и гарантировать целостность распределенной базы данных. Средства удаленного вызова процедур наиболее соответствуют идее распределенных вычислений: они обеспечивают из любого узла сети вызов прикладной процедуры, расположенной на другом узле, передачу параметров, удаленную обработку и возврат результатов. С ростом систем клиент-сервер необходимость трех уровней становится все более очевидной. Продукты для трехуровневой архитектуры, так называемые мониторы транзакций, являются относительно новыми. Эти инструменты в основном ориентированы на среду Unix, однако прикладные серверы можно строить на базе Microsoft Windows NT с вызовом удаленных процедур для организации связи клиентов с сервером приложений. На практике в локальной сети могут использоваться смешанные архитектуры (двухуровневые и трехуровневые) с одним и тем же сервером базы данных. С учетом глобальных связей архитектура может иметь больше трех уровней.   
Таким образом, многоуровневая архитектура распределенных приложений позволяет повысить эффективность работы корпоративной информационной системы и оптимизировать распределение ее программно-аппаратных ресурсов. Но пока на российском рынке по-прежнему доминирует архитектура клиент-сервер. 

**Интернет/интранет-технологии**

В развитии Интернет/интранет-технологий основной акцент пока что делается на разработке инструментальных программных средств. В то же время наблюдается отсутствие развитых средств разработки приложений, работающих с базами данных. Компромиссным решением для создания удобных и простых в использовании и сопровождении информационных систем, эффективно работающих с базами данных, стало объединение Интернет/интранет-технологий с многоуровневой архитектурой. При этом структура информационного приложения приобретает следующий вид:   
браузер — сервер приложений — сервер баз данных — сервер динамических страниц — веб-сервер.   
Благодаря интеграции Интернет/интранет-технологий и архитектуры клиент-сервер, процесс внедрения и сопровождения корпоративной информационной системы существенно упрощается при сохранении достаточно высокой эффективности и простоты совместного использования информации.

## Требования, предъявляемые к информационным системам

Информационная система должна соответствовать требованиям гибкости, надежности, эффективности и безопасности. 

### Гибкость

Гибкость, способность к адаптации и дальнейшему развитию подразумевает возможность приспособления информационной системы к новым условиям, новым потребностям предприятия. Выполнение этих условий возможно, если на этапе разработки информационной системы использовались общепринятые средства и методы документирования, так что по прошествии определенного времени сохранится возможность разобраться в структуре системы и внести в нее соответствующие изменения, даже если все разработчики или их часть по каким-либо причинам не смогут продолжить работу.   
Любая информационная система рано или поздно морально устареет, и станет вопрос о ее модернизации или полной замене. Разработчики информационных систем, как правило, не являются специалистами в прикладной области, для которой разрабатывается система. Участие в модернизации или создании новой системы той же группы проектировщиков существенно сократит сроки модернизации. Вместе с тем возникает риск применения устаревших решений при модернизации системы. Рекомендация в таком случае одна — внимательнее относиться к подбору разработчиков информационных систем. 

### Надежность

Надежность информационной системы подразумевает ее функционирование без искажения информации, потери данных по «техническим причинам». Требование надежности обеспечивается созданием резервных копий хранимой информации, выполнения операций протоколирования, поддержанием качества каналов связи и физических носителей информации, использованием современных программных и аппаратных средств. Сюда же следует отнести защиту от случайных потерь информации в силу недостаточной квалификации персонала. 

### Эффективность

Система является эффективной, если с учетом выделенных ей ресурсов она позволяет решать возложенные на нее задачи в минимальные сроки. В любом случае оценка эффективности будет производиться заказчиком, исходя из вложенных в разработку средств и соответствия представленной информационной системы его ожиданиям.   
Негативной оценки эффективности информационной системы со стороны заказчика можно избежать, если представители заказчика будут привлекаться к проектированию системы на всех его стадиях. Такой подход позволяет многим конечным пользователям уже на этапе проектирования адаптироваться к изменениям условий работы, которые иначе были бы приняты враждебно.   
Активное сотрудничество с заказчиком с ранних этапов проектирования позволяет уточнить потребности заказчика. Часто встречается ситуация, когда заказчик чего-то хочет, но сам не знает чего именно. Чем раньше будут учтены дополнения заказчика, тем с меньшими затратами и в более короткие сроки система будет создана.   
Кроме того, заказчик, не являясь специалистом в области разработки информационных систем, может не знать о новых информационных технологиях. Контакты с заказчиком во время разработки для него информационной системы могут подтолкнуть заказчика к модернизации его аппаратных средств, применению новых методов ведения бизнеса, что отвечает потребностям, как заказчика, так и проектировщика. Заказчик получает рост эффективности своего предприятия, проектировщик — расширение возможностей, применяемых при проектировании информационной системы.   
Эффективность системы обеспечивается оптимизацией данных и методов их обработки, применением оригинальных разработок, идей, методов проектирования.   
Не следует забывать и о том, что работать с системой придется обычным людям, являющимся специалистами в своей предметной области, но зачастую обладающим весьма средними навыками в работе с компьютерами. Интерфейс информационных систем должен быть им интуитивно понятен. В свою очередь, разработчик-программист должен понимать характер выполняемых конечным пользователем операций. Рекомендациями в этом случае могут служить повышение эффективности управления разработкой информационных систем, улучшение информированности разработчиков о предметной области. 

### Безопасность

Под безопасностью, прежде всего, подразумевается свойство системы, в силу которого посторонние лица не имеют доступа к информационным ресурсам организации, кроме тех, которые для них предназначены. Защита информации от постороннего доступа обеспечивается управлением доступа к ресурсам системы, использованием современных программных средств защиты информации. В крупных организациях целесообразно создавать подразделения, основным направлением деятельности которых было бы обеспечение информационной безопасности, в менее крупных организациях назначать сотрудника, ответственного за данный участок работы.   
Помимо злого умысла, при обеспечении безопасности информационных систем приходится сталкиваться еще с несколькими факторами. В частности, современные информационные системы являются достаточно сложными программными продуктами. При их проектировании с высокой вероятностью возможны ошибки, вызванные большим объемом программного кода, несовершенством компиляторов, человеческим фактором, несовместимостью с используемыми программами сторонних разработчиков в случае модификации этих программ и т.п. Поэтому за фазой разработки информационной системы неизбежно следует фаза ее сопровождения в процессе эксплуатации, в которой происходит выявление скрытых ошибок и их исправление.   
Требование безопасности обеспечивается современными средствами разработки информационных систем, современной аппаратурой, методами защиты информации, применением паролей и протоколированием, постоянным мониторингом состояния безопасности операционных систем и средств их защиты.

**Страницы:**[**1**](http://mirea.kremlina.ru/information-systems/4-lekciya-1.html)**2**