**Инструментальные средства разработки программного обеспечения**

**Инструментальные средства разработки программного обеспечения** – это программные инструменты, предназначенные для обеспечения полного цикла программного продукта.

Инструментальной средой разработки и сопровождения ПС будем называть совокупность элементов.

Для таких инструментальных сред характерно, во-первых, использование как программных, так и аппаратных инструментов, и, во-вторых, определенная ориентация либо на конкретный язык программирования, либо на конкретную предметную область.

**Инструментальная среда** не обязательно должна функционировать на том компьютере, на котором должно будет применяться разрабатываемое с помощью ее ПС. Часто такое совмещение бывает достаточно удобным, не нужно иметь дело с компьютерами разных типов, в разрабатываемую ПС можно включать компоненты самой инструментальной среды.

С точки зрения функций, которые инструменты выполняют при разработке ПС, их можно разбить на следующие четыре группы:

* редакторы,·
* анализаторы,·
* преобразователи,·
* инструменты, поддерживающие процесс выполнения программ.

Программы компиляторы транслируют текст программы с языка программирования в машинный код.

Программы компоновщики - производят компоновку программы из нескольких модулей, а также подключает нужные библиотеки и определяет ссылки между модулями.

Программы отладчики – предназначенные для анализа, выполнения программы и выявления ошибок при ее работе.

Программа для создания инсталляторов - используют дистрибутивы учитывающие особенности платформ на которых создается

Программы создания справочной системы – она позволяет организовать файлы с помощью нужной структуры

Специализированный набор инструментов - который используется при разработки только некоторых программ или изучения структуры построения кода программ сторонних производителей.

Различают **три основных класса инструментальных сред разработки и сопровождения ПС**

среды программирования,

рабочие места компьютерной технологии,·

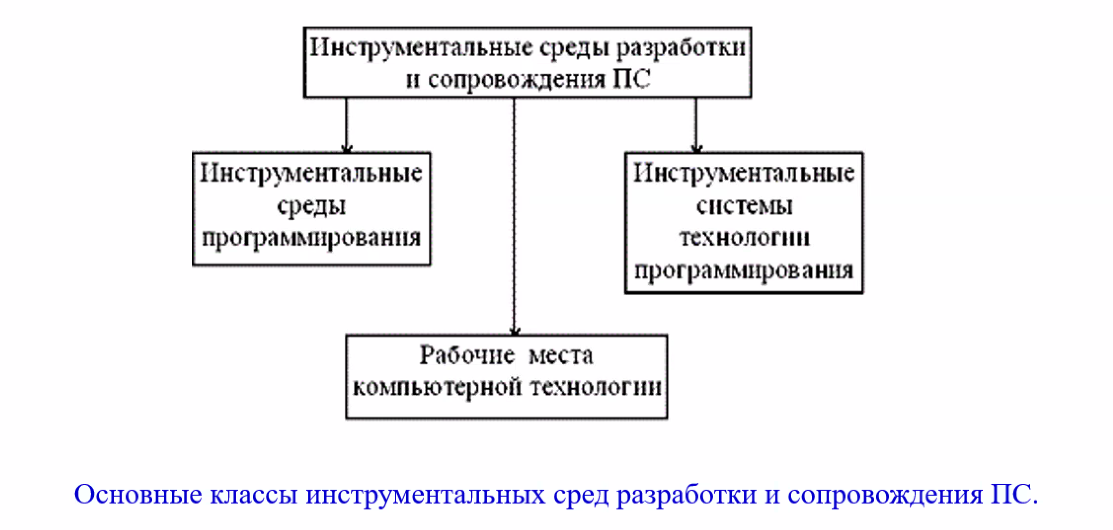
инструментальные системы технологии программирования.

Среда программирования предназначена

в основном для поддержки процессов программирования (кодирования), тестирования и отладки ПС.

Рабочее место компьютерной технологии ориентировано на поддержку ранних этапов разработки ПС (спецификаций) и автоматической генерации программ по спецификациям.

Инструментальная система технологии программирования предназначена для поддержки всех процессов разработки и сопровождения в течение всего жизненного цикла ПС и ориентирована на коллективную разработку больших программных систем с длительным жизненным циклом.



Инструментальные среды программирования содержат прежде всего

текстовый редактор, позволяющий конструировать программы на заданном языке программирования, инструменты, позволяющие компилировать или интерпретировать программы на этом языке, а также тестировать и отлаживать полученные программы.

Кроме того, могут быть и другие инструменты, например, для статического или динамического анализа программ.

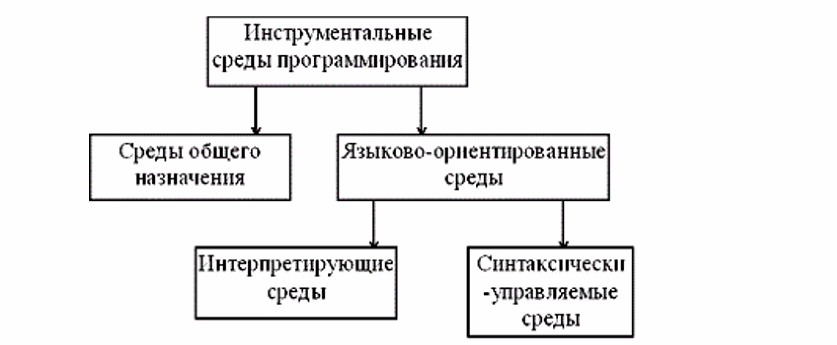
Взаимодействуют эти инструменты между собой через обычные файлы с помощью стандартных возможностей файловой системы.

Различают следующие **классы инструментальных сред программирования**:

среды общего назначения,·

языково-ориентированные среды.

**Инструментальные среды программирования общего назначения** содержат набор программных инструментов, поддерживающих разработку программ на разных языках программирования (например, текстовый редактор, редактор связей или интерпретатор языка целевого компьютера) и обычно представляют собой некоторое расширение возможностей используемой операционной системы. Для программирования в такой среде на каком-либо языке программирования потребуются дополнительные инструменты, ориентированные на этот язык (например, компилятор).



CASE - это абревиатура от английского Computer-Aided Software Engineering. Но без помощи (поддержки) компьютера ПС уже давно не разрабатываются (используется хотя бы компилятор).

Первоначально под CASE понималась инженерия ранних этапов разработки ПС (определение требований, разработка внешнего описания и архитектуры ПС) с использованием программной поддержки (программных инструментов).

Теперь под CASE может пониматься  и инженерия всего жизненного цикла ПС (включая и его сопровождение), но только в том случае, когда программы частично или полностью генерируются по документам, полученным на указанных ранних этапах разработки. В этом случае CASE-технология стала принципиально отличаться от ручной (традиционной) технологии разработки ПС: изменилось не только содержание технологических процессов, но и сама их совокупность.

Средства CASE технологий делятся на две группы:

1. Встроенные системы реализации – где все решения по проектированию и реализации привязаны к выбранной СУБД.
2. Независимые системы реализации – где все решения по проектированию ориентированы на унификацию начальных этапов жизненного цикла и средств их документирования.

В настоящее время компьютерную технологию разработки ПС можно характеризовать

- Использованием программной поддержки для разработки графических требований и графических спецификаций ПС,

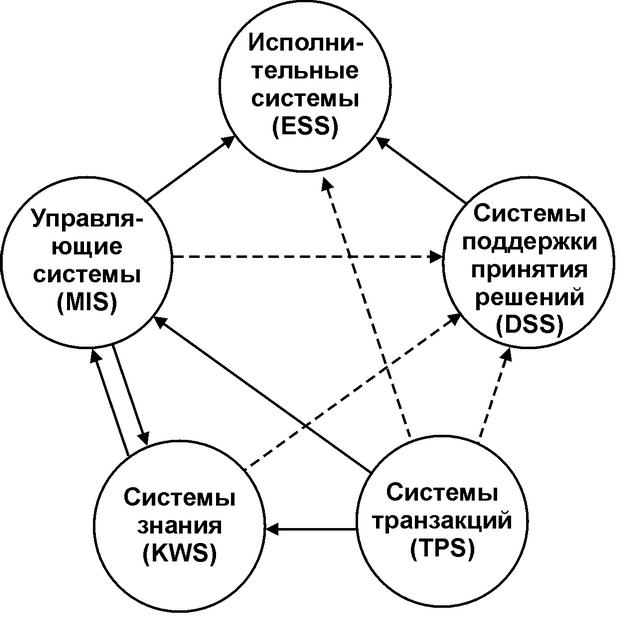
- автоматической генерации программ на каком-либо языке программирования или в машинном коде,

- программной поддержки прототипирования.

### Интеграция информационных систем предприятия

#### Взаимосвязь информационных подсистем предприятия

Каким образом связаны информационные системы внутри предприятия? Обычный путь для российской компании средних размеров — начинать внедрение информационных технологий с автоматизации работы бухгалтерии, отдела кадров и документооборота. Данные этих систем наиболее формализованы, процессы легко автоматизируются. Широко распространенные пакеты "1C: Бухгалтерия", "Босс: Кадровик", "LanDocs", "LanStaff", "Salary" и др. позволяют наращивать себя любыми приложениями и, таким образом, интегрировать их в общую информационную систему предприятия. Рис. 5.1 показывает, каким образом модули информационной системы компании связаны друг с другом. Модуль TPS обслуживает основные производственные и вспомогательные процессы, и обычно это главный источник для других информационных модулей. ESS — главный получатель данных и внутренних систем и внешней среды.



**Рис. 5.1.**Взаимодействие модулей ИС

Другие системы также обмениваются данными. И здесь возникает один из самых трудных вопросов для руководителя — поиск оптимальной степени интеграции. Большой соблазн иметь абсолютно интегрированную систему, но такая интеграция чрезвычайно трудоемка, стоит немалых денег. И лучше даже не говорить, во что обходится сопровождение такой системы. Поэтому нужно взвесить потребности в интегрированных системах, поставив их на чашу весов против трудностей и дороговизны крупномасштабной ИС. Не существует стандартного уровня интеграции или централизации — каждый руководитель должен самостоятельно (или с помощью консалтинговой фирмы) решать эту непростую проблему.

Связи между DSS и совокупностью TPS, KWS, MIS намеренно показаны неопределенными. Иногда DSS тесно связана с другими подсистемами. Но это только в том случае, если предприятие отличается высокой степенью автоматизации всех процессов. Обычно подсистема DSS изолированы от основных производственных информационных систем и использует их данные и информационные потоки для работы своих аналитических систем.

В любом случае, нет рецептов на все случаи — все зависит от организационно-функциональной структуры конкретного предприятия, структуры его бизнеса, реальных инвестиционных возможностей и политики развития.

#### Сервис-ориентированная архитектура ИС

Интеграция разнородных и распределенных данных не в состоянии разрешить все вопросы управления предприятием. В соответствии с процессным подходом наибольшую ценность представляют не сами по себе данные, а использование информации в тех или иных бизнес-процессах компании. В самых современных ИС принято рассматривать за "атомарную" единицу не данные в "чистом" виде, а некоторый сервис, соответствующий какому-то элементарному бизнес-процессу. В частности, такой сервис может просто выдавать какие-то данные, являясь аналогом "атомарной" единицы классических ИС.

В настоящее время при формировании информационной инфраструктуры предприятия, при проектировании и реализации КИС всё чаще применяется сервис-ориентированная архитектура (Service-Oriented Architecture — SOA). Это такая архитектура ИС, в которой система строится из набора гетерогенных слабосвязанных компонентов (сервисов). SOA понимается как парадигма организации и использования распределенного множества функций, которые могут контролироваться различными владельцами. Базовыми понятиями в такой архитектуре являются "информационная услуга" и "композитное приложение".

Информационная услуга (сервис) — это атомарная прикладная функция автоматизированной системы, пригодная для использования при разработке приложений, реализующих прикладную логику автоматизируемых процессов как в самой системе, так и для использования в приложениях других автоматизированных систем.

Сервис обычно характеризуется следующими свойствами:

* возможность многократного применения;
* услуга может быть определена одним или несколькими технологически независимыми интерфейсами;
* выделенные услуги слабо связаны между собой и каждая из них может быть вызвана посредством коммуникационных протоколов, обеспечивающих возможность взаимодействия услуг между собой.

Композитное (составное) приложение — программное решение для конкретной прикладной проблемы, связывающее прикладную логику процесса с источниками данных и информационных услуг, хранящихся на гетерогенном множестве базовых информационных систем. Обычно композитные приложения ассоциированы с процессами деятельности и могут объединять различные этапы процессов, представляя их пользователю через единый интерфейс.

Использование такого подхода при построении архитектуры сложных интегрированных информационных систем позволяет:

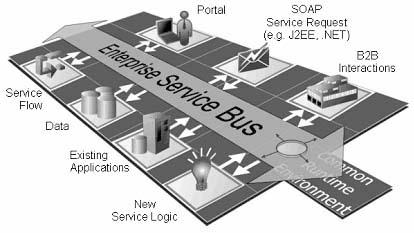
* создать систему корпоративных композитных приложений, основанных на системе корпоративных Web-сервисов;
* организовать интеграцию приложений на базе автоматизации бизнес-процессов;
* использовать различные транспортные протоколы и стандарты форматирования сообщений, средства обеспечения безопасности, надежной и своевременной доставки сообщений;
* существенно повысить скорость разработки прикладных приложений и снизить затраты на эти цели.

Благодаря упрощению среды управления и взаимодействия снижается потребность в кодировании новых программ. Повторное использование сервисов сокращает затраты времени на разработку; рационализация унаследованных процессов помогает уменьшить общее число процессов, требующих эксклюзивных методов управления. Благодаря использованию простых протоколов, значительно сокращаются трудозатраты на поддержку приложений.

Обязательным условием построения и внедрения архитектуры системы на основе SOA является использование единой инфраструктуры описания сервисов (репозитория сервисов), разрешенных протоколов доступа и обмена сообщениями, форматов сообщений.

Упомянутая инфраструктура образует так называемую интеграционную шину (Enterprise Service Bus — ESB), являющуюся одним из центральных компонентов системы. Она устанавливает единые правила публикации сервисов, управления и информационного взаимодействия между приложениями различных систем, входящих в состав интегрированной системы. Это упрощает управление приложениями и их поддержку, а также снижает риск фрагментации приложений и процессов.

Основные компоненты архитектуры информационной системы, построенной на основе концепции SOA и ESB, представлены на рис. 5.2.



**Рис. 5.2.**Структура построения ESB и компоненты концепции SOA

Каждая из служб взаимодействует не с остальными службами напрямую, а только с шиной. ИШ образует однородную среду информационного взаимодействия и является фундаментом для интеграции информационных систем, функционирующих в различных учреждениях и ведомствах. ИШ определяет кем, где, каким образом и в каком порядке должны обрабатываться запросы.

Если сервис (информационный ресурс) не поддерживает эти правила, необходимо создавать промежуточный модуль-адаптер, который предоставляет системе необходимый интерфейс и обеспечивает взаимодействие с ресурсом.

По данным Gartner Group ("Predicts 2007: SOA Advances", 17 ноября 2006): "К 2008 году SOA станет господствующей архитектурой построения ИТ-систем, что приведет к окончанию 40-летней эры господства архитектуры монолитных приложений". Отметим, что этот прогноз в большой степени оправдался.

Изменение и совершенствование бизнес-процессов в компаниях занимает годы. По усредненным данным Gartner Group: 80 % ИТ-бюджета — это расходы на сопровождение систем, из них 35 % — затраты на интеграцию приложений, 60 % стоимости внедрения корпоративной ИС составляют расходы на интеграцию, 50 % ИТ-бюджета потрачено на обеспечение интерфейсов систем. Использование SOA архитектуры позволяет эффективно организовать оперативную адаптацию ИТ-систем под требования бизнеса, что дает стратегическое преимущество компании, заключающееся в:

* повышение скорости адаптации бизнеса к быстроменяющимся требованиям рынка (Agility);
* расширении взаимодействия гетерогенных корпоративных информационных систем при сохранение сделанных в них инвестиций;
* сокращение расходов на ИТ-системы на основе повторного использования их функциональных компонентов;
* повышение производительности труда клиентов, партнеров и сотрудников (на основе архитектуры Web 2.0).

С точки зрения бизнеса SOA можно представить как набор гибких служб и процессов, которые бизнес предлагает своим заказчикам, партнерам или внутри своей собственной организации. В данном контексте эти же службы можно по-разному комбинировать и оснащать, поддерживая изменения или развитие бизнес-требований и моделей с течением времени.

Основные бизнес-цели внедрения SOA-решений состоят в ликвидации:

* фрагментированности и дублирование данных;
* дублирования реализаций бизнес-функций, процедур, процессов негибкой архитектуры.

Становление и развитие SOA происходило на базе практических требований бизнеса, заключавшимхся, прежде всего, в разумной экономии программных и технологических средств и затрат на реализацию и сопровождение информационной инфраструктуры:

* обеспечивать преемственность инвестиций в IT, сохранение существующих информационных систем и их совместное эффективное использование для повышения ROI от IT-вложений;
* обеспечивать реализацию различных типов интеграции:
  + пользовательская интеграция (User Integration) — обеспечение взаимодействия информационной системы с конкретным персонифицированным пользователем;
  + интеграция приложений (Application Connectivity) — обеспечение взаимодействия приложений;
  + интеграция процессов (Process Integration) — интеграция процессов в соответствии с бизнес-логикой деятельности предприятия;
  + информационная интеграция (Information Integration) — интеграция с целью обеспечения доступности информации и данных;
  + интеграция новых приложений (Build to Integrate) — интеграция новых приложений и сервисов в существующие информационные системы.
* обеспечивать поэтапность внедрения вновь созданных и миграции существующих информационных систем;
* иметь стандартизованную технологическую обеспеченность реализации и инструментарий разработки, совокупно предоставляющие наилучшие возможности повторного использования приложений, внедрения новых и миграции существующих информационных систем;
* позволять реализацию различных моделей построения информационных систем, в особенности таких как портальные решения, grid-системы и on-demand-системы.

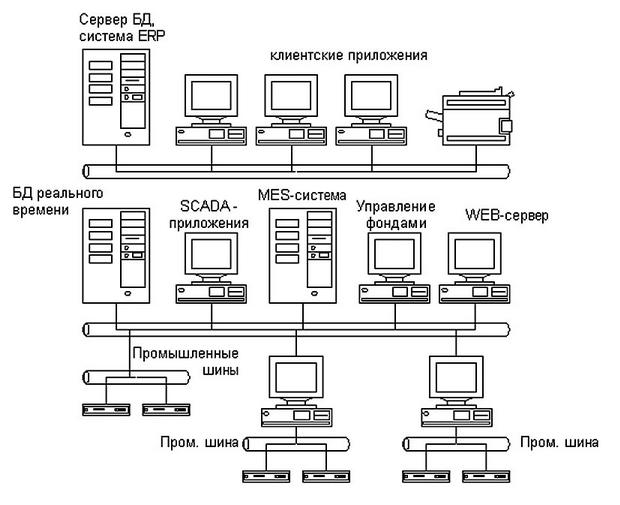
Сегодняшний уровень развития SOA позволяет утверждать, что все указанные требования в той или иной мере выполняются. Рост рынка продуктов для SOA-решений — 100 % в год. В 2007 году SOA была использована как основа создания 50 % новых, критичных для бизнеса приложений и бизнес-процессов; к 2012 году этот показатель вырос до 85 %. Более 80 % приложений, введенных в промышленное использование в 2010 году, будут частично или полностью перепроектированы к 2014 году, чтобы быть использованы в построении композитных приложений в SOA-архитектуре.

К 2014 более 80 % всех программных инфраструктурных продуктов будут включать корпоративную шину сервисов или требовать ее использования. Среди исполнительных директоров компаний 58 % считают, что в период до 2015года в числе главных стратегических преимуществ компаний новые модели ведения бизнеса имеют бoльшее значение, чем выпуск новых продуктов и услуг. По данным Forrester ("The State of SOA in Financial Services", январь 2014 года) "Большинство финансовых компаний будут использовать SOA к концу 2014 г. В настоящее время более 60 % европейских финансовых компаний или уже используют SOA или на последней стадии внедрения".

1. **Варианты интеграционных решений**

Многообразие применяемых технологий и систем, разнообразие форматов данных, циркулирующих в информационных потоках, обилие аналитических и отчётных форм сделали чрезвычайно актуальной задачу интеграции указанных выше технологических и информационных объектов и сущностей, а также физические и виртуальные пространства их взаимодействия в единую информационно-управленческую среду (рис. 5.3)

Интеграция — это не просто механическое объединение модулей информационной системы. При разработке плана интеграции исходят прежде всего из стратегических целей развития предприятия, возможного изменения бизнес-логики, в соответствии с которой выстраиваются бизнес-процессы и осуществляется их информационное сопровождение. Интеграция может производиться на уровне форматов и баз данных, программно-аппаратных и сетевых устройств, пользовательских интерфейсов, форм и шаблонов документооборота, программных приложений и т.д. Выгоды от такой интеграции очевидны.



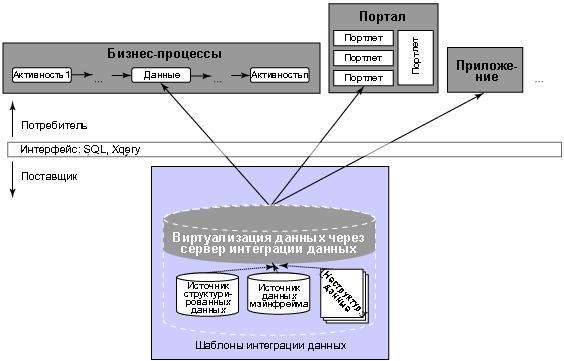
**Рис. 5.3.**Общая схема аппаратно-комммуникационной реализации интегрированной системы управления предприятием

Подход к разработке и внедрению КИС, основанный на интеграции приложений, позволяет:

* сохранить ранее сделанные инвестиции;
* сократить временные и финансовые затраты на поддержку и развитие информационного пространства компании;
* использовать для решения конкретных задач наиболее эффективные системы отдельных производителей;
* легко расширять и развивать отдельные возможности существующих информационных систем с уже накопленными в них данными.

##### Интеграция на уровне данных

Одной из главных проблем интеграции данных является обилие форматов и типов (неструктурированные, частично-структурированные, жёстко-структурированные) данных, а также лавинообразное нарастание их объёмов. Циркулирование разнородных массивов данных и информации в сетях различных служб предприятия создает множество проблем с их сбором, структурированием, обработкой, анализом, хранением, архивированием и передачей пользователю для принятия делового решения. На рисунке 5.4 показана традиционная схема интеграции данных.



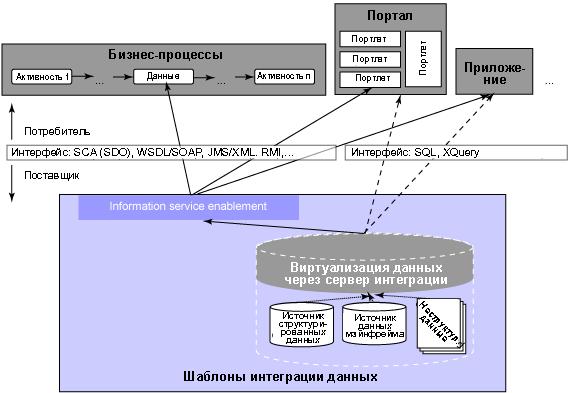
**Рис. 5.4.**Традиционная схема интеграции данных

Для их интеграции в настоящее время обычно используют стандартные интерфейсы и протоколы, например, SQL и JDBC/ODBC, применяют различные инструменты реляционных баз данных (Relational Database — RD), сквозных репозиториев — баз данных с "надстройкой", содержащей информацию об артефактах и объектах проектирования, надмножество словарей метаданных (Transparent Repository — TR) и современных хранилищ и фабрик данных (Data Warehouse, Data Factory — DW, DF).

Последний вид технологий интеграции применяется, как правило, в крупных компаниях и производственных объединениях. Такие технологии создают удобную для пользователя единую среду для хранения и использования данных. Ниже будет подробнее рассказано о системах коллективного использования информации.

##### Интеграция на уровне физических, программных и пользовательских интерфейсов

Этот вид интеграции начинался как один из видов "лоскутной интеграции", когда предпринимались попытки объединить разрозненные программные приложения, написанные в разное время разными разработчиками, в подобие единого целого. Приложения объединялись по принципу "каждый с каждым", что, в конечном счёте, усложняло их взаимодействие и создавало массу проблем. Кроме того, всё сложнее становилось использовать унаследованные (Legacy Software) и встроенные (Embedded System) системы.



**Рис. 5.5.**Организация доступа к интегрированным данным через открытые интерфейсы

Такой подход хорош для небольшого количества приложений. При большом их числе он практически не работает и не позволяет строить качественно новые запросы к агрегированным данным, т.е. существенного выигрыша от объединения данных нет. В настоящее время проблема интеграции на уровне интерфейсов решается на базе использования информационных подсистем, реализованных стандартными программными приложениями с открытыми интерфейсами (Open Application Programming Interface).

Подобные унифицированные интерфейсы разрабатываются, например, на базе семейства международных стандартов POSIX. В этом случае степень интегрируемости можно характеризовать некоторым числовым показателем (метрикой) который можно, условно говоря, вычислить, перемножив показатель "качества" и "показатель открытости" программного интерфейса. Показателем качества могут выступать такие характеристики, как "совместимость", "надёжность", "переносимость", "понятность", "удобство использования" и пр. В результате мы получим индекс, который (в известной степени) характеризует способность приложения быть частью какого-то другого, глобального композитного приложения.

В настоящее время всё чаще применяется следующий алгоритм: отделяют слой обработки данных от привязанных к ним форм визуализации и реализуют прикладную бизнес-логику на одном из языков третьего поколения (3GL), оформив программный доступ к прикладным функциям в виде хорошо документированного программного интерфейса (рис. 5.5).

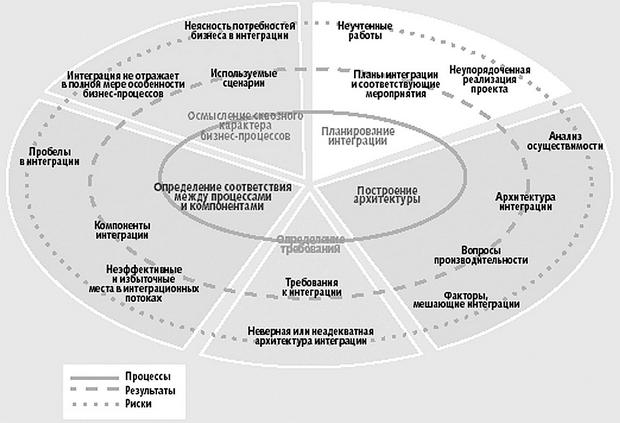
##### Интеграция на функционально-прикладном и организационном уровнях

Этот вид интеграции предполагает объединение ряда однотипных или схожих функций в макрофункции с перераспределением потоков данных и управления, а также ресурсов и механизмов для исполнения. Это часто влечёт за собой перестройку организационных структур, бизнес-процессов и, соответственно, схему их информационного и документационного обеспечения.

Выгоды от такой интеграции очевидны — процессы становятся более прозрачными, управляемыми, менее затратными, уменьшается количество обслуживающего персонала, число ошибок при формировании документов и т.д. Однако интеграция такого вида влечёт за собой существенную перестройку или полный реинжиниринг сети процессов, что связано с крупными рисками. Чаще всего такая интеграция проводится в том случае, когда предприятие готовится к внедрению КИС на базе известного решения, которое требует привести бизнес-процессы к требуемому стандарту, или перестраивает свою деятельность в связи со сменой устремлений, открытием филиалов в других странах, освоением новых сегментов рынка и т.д.

##### Интеграция на уровне корпоративных программных приложений

Интеграция на уровне приложений (Enterprise Application Integration — EAI,) подразумевает совместное использование исполняемого кода, а не только внутренних данных интегрируемых приложений. Программы разбиваются на компоненты, которые интегрируются с помощью стандартизованных программных интерфейсов и специального связующего ПО.

[](https://intuit.ru/EDI/15_10_17_5/1508019676-18691/tutorial/1320/objects/5/files/5_6.jpg)  
**Рис. 5.6.**Схема применения методологии EIM

При таком подходе из этих компонентов создается универсальное программное ядро или платформа, с помощью которых используют все приложения. Для каждого приложения создается только один интерфейс для связи с этим ядром, что существенно облегчает задачу интеграции. Полученную в результате систему легче поддерживать и расширять. Повторное использование функций в рамках имеющейся среды позволяет значительно снизить время и стоимость разработки приложений. В этом случае анализ внутренней конструкции приложений — обязательный этап в оценке степени интегрируемости тех приложений, которые предполагается связывать в рамках того или иного проекта. Этот анализ усложняется тем, что обычно разработчики приложений, являющихся законченными программными продуктами, как правило, не показывают деталей внутренней конструкции приложений.

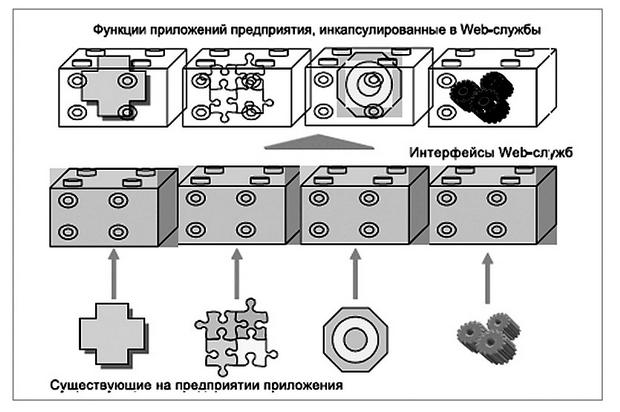
В связи с этим технология интеграции в настоящее время рассматривает не просто интеграцию приложений, но их интеграцию на базе интеграции бизнес-процессов – в этом случае следует говорить об интеграции на уровне всего предприятия (Enterprise Integration Metodology — EIM). Схема такой объединенной методологии показана на рисунке 5.6.

Методология EIM реализуется современными технологиями и инструментами, среди которых можно, например, указать рассмотренную выше технологию интеграции на базе сервис-ориентированных архитектур (SOA). Архитектура ИС в таком случае строится из набора гетерогенных слабосвязанных компонентов (сервисов) и понимается как парадигма организации и использования распределенного множества функций, которые могут контролироваться различными владельцами. Базовыми понятиями в такой архитектуре являются "информационная услуга" и "композитное приложение".

##### Интеграция при помощи Web-сервисов

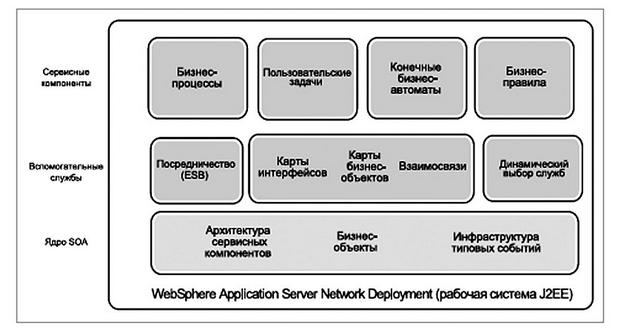
Самый современный и быстро развивающийся подход к интеграции приложений. Он основан на обеспечении стандартного для Web-служб интерфейса доступа к приложениям и данным (рис.5.7).

Например, используя стандартный протокол доступа к объектам SOAP (Simple Object Access Protocol), браузер пользователя может сравнить данные на нескольких сайтах и представить клиенту сравнительный отчет. Другой пример — сотрудники территориально распределенного предприятия могут одновременно использовать корпоративные приложения, доступ к которым осуществляется через соответствующие Web-сервисы (портальное решение).

  
**Рис. 5.7.**Схема доступа с использованием Web-служб

Web-сервисы напоминают подход EAI, но с одним важным отличием — в большинстве случаев EAI-решения разрабатываются как частные для связи конкретных продуктов. Соответственно, подключить к существующему EAI-решению еще одну систему — достаточно трудная и долговременная задача. Web-сервисы существенно более унифицированы и стандартизованы. Поскольку Web-сервисы основаны на общих для W3C-консорциума стандартах, они могут работать всюду, где используется всемирная паутина (WWW). Результаты построения КИС на основе Web-интеграции:

* возможность осуществлять оперативное управление распределенной компанией и ведение консолидированного управленческого учета по нескольким филиалам;
* возможность осуществлять планомерное развитие общекорпоративной информационной системы, интегрируя в нее функциональные компоненты, исходя из приоритетов развития бизнеса компании и потребностей функциональных подразделений, т.е. возможность синхронизировать развитие системы с развитием бизнеса;
* возможность при необходимости заменить любой функциональный компонент другим, более соответствующим текущим бизнес-потребностям;
* возможность инвестировать в развитие информационных технологий не сразу, а поэтапно, на каждом этапе соотнося вложенные средства с полученным бизнес-эффектом, а также снижать общую стоимость автоматизированного рабочего места, включая затраты на создание системы, поддержку рабочих мест и обучение пользователей;
* резкое снижение времени сбора информации, необходимой для принятия управленческих и деловых решений, сокращение времени и трудозатрат на ведение учетных операций, на формирование промежуточных отчетов, на сверку информации между подразделениями и ликвидация противоречивости и несовместимости данных от различных служб;
* cохранение инвестиций в имеющиеся системы и оборудование, в обучение персонала.

[](https://intuit.ru/EDI/15_10_17_5/1508019676-18691/tutorial/1320/objects/5/files/5_8.jpg)  
**Рис. 5.8.**Архитектурная модель WebSphere Application Server

В настоящее время крупные разработчики программных продуктов предлагают консолидированные решения, которые содержат не только конкретные инструменты для разработки и внедрения изначально интегрированных корпоративных приложений, но и реализуют интегрированную среду разработки таких приложений. Примером такого решения может служить программный продукт IBM WebSphere (рис. 5.8).