Архитектура параллельных вычислительных систем

к.т.н., доцент
Костичев Сергей Валентинович

Современные информационные технологии в вычислительных системах

snenv@mail.ru

Учебные вопросы:

1. Сервис-ориентированная архитектура

2. Виртуализация

3. Облачные вычисления и технологии

1. Сервис-ориентированная архитектура (SOA)

Концепция сервисно-ориентированной архитектуры

Интеграция информации - объединение разнородного по структуре и распределенного контента.

Включает в себя интеграцию

- программных приложений,
- данных,
- процессов,
- аппаратных средств
- рабочих мест.

Интеграция программных приложений - основа для интеграции всех других компонентов

Одна из важных задач интеграции программных средств - **борьба за повторную используемость (reusability) программного кода**:

1) процедуры и функции - сначала встроенные, а затем оформляемые как ***модули***.

   Модуль представляет собой единицу программного кода с хорошо описанным интерфейсом, отдельно компилирующуюся и отдельно хранящуюся.

   Каркас (framework) - развитая библиотека модулей, предоставляющая исчерпывающий набор средств для решения определенного класса технологических или прикладных задач**.**

   Применение каркаса превращает создание нового приложения в сборку программы из готовых элементов каркаса.

2) компонентная архитектура.

Компонент - это модуль, реализующий *стандартный* интерфейс. Стандартизация интерфейса обеспечивает взаимозаменяемость компонентов и взаимодействие компонента с промежуточным ПО - каркасами, инструментальными средствами разработки, контейнерами.

Контейнер представляет собой некоторое промежуточное ПО, которое поддерживает выполнение *компонентов*. Выполняет также интеграционную функцию.

В **технологиях распределенных приложений** (платформы Microsoft .NET и Java 2 Enterprise Edition - J2EE) **компонентная архитектура обеспечивает быструю и удобную интеграцию компонентов.**

Это относится только к компонентам, которые были разработаны в соответствии со спецификациями данной платформы.

Однако при интеграции с информационными системами партнеров, поставщиков, потребителей и т.д. **приведение всех компонентов системы к единой платформе невозможно**.

Необходимость решения этих проблем породила **концепцию сервисно-ориентированной архитектуры (SOA).**

Сервис - это программный компонент, реализующий законченную функцию предоставления или обработки данных.

Основным отличием сервиса от обычного компонента является ***стандартный*** и ***платформенно-независимый*** **интерфейс**.

Клиент, обращающийся к сервису, **не обязан ничего знать о подробностях реализации** сервиса.

Оформление процесса как сервиса предъявляет **требования к его интерфейсу**, но не к разработке процесса.

Платформа, реализующая SOA, должна удовлетворять следующим требованиям:

- обеспечивать спецификации интерфейса, которые приняты большинством разработчиков компонентов;
- использовать общепринятые протоколы для взаимодействия сервисов и клиентов;
- не использовать в интерфейсе какие-либо сложные и/или закрытые форматы представления информации;
- не требовать для своей поддержки дорогостоящего или ресурсоемкого ПО.

Сегодня **единственной общепринятой реализацией SOA являются Web-сервисы**.

Имеют статус открытых стандартов, поддерживаемых W3C (www.w3c.org) и OASIS (www.oasis-open.org) и всеми ведущими фирмами, работающими в сфере ИТ. Инициаторы - фирмы Microsoft и IBM.

Web-сервисы - это XML-приложения, осуществляющие связывание данных с программами, объектами, базами данных либо с деловыми операциями целиком.

Web-сервисы представляют собой оболочку, обеспечивающую стандартный способ взаимодействия с прикладными программными средами, такими как системы управления базами данных (СУБД), .NET, J2EE (Java2 Platform, Enterprise Edition).

Принципы Web-сервисов

- Создатель конкретного Web-сервиса определяет формат запросов к нему и формат ответов на данные запросы;
- С любого компьютера в Интернет можно сделать запрос к данному Web-сервису;
- Web-сервис выполняет заданную последовательность действий и отправляет обратно результат.

Части Web-сервиса

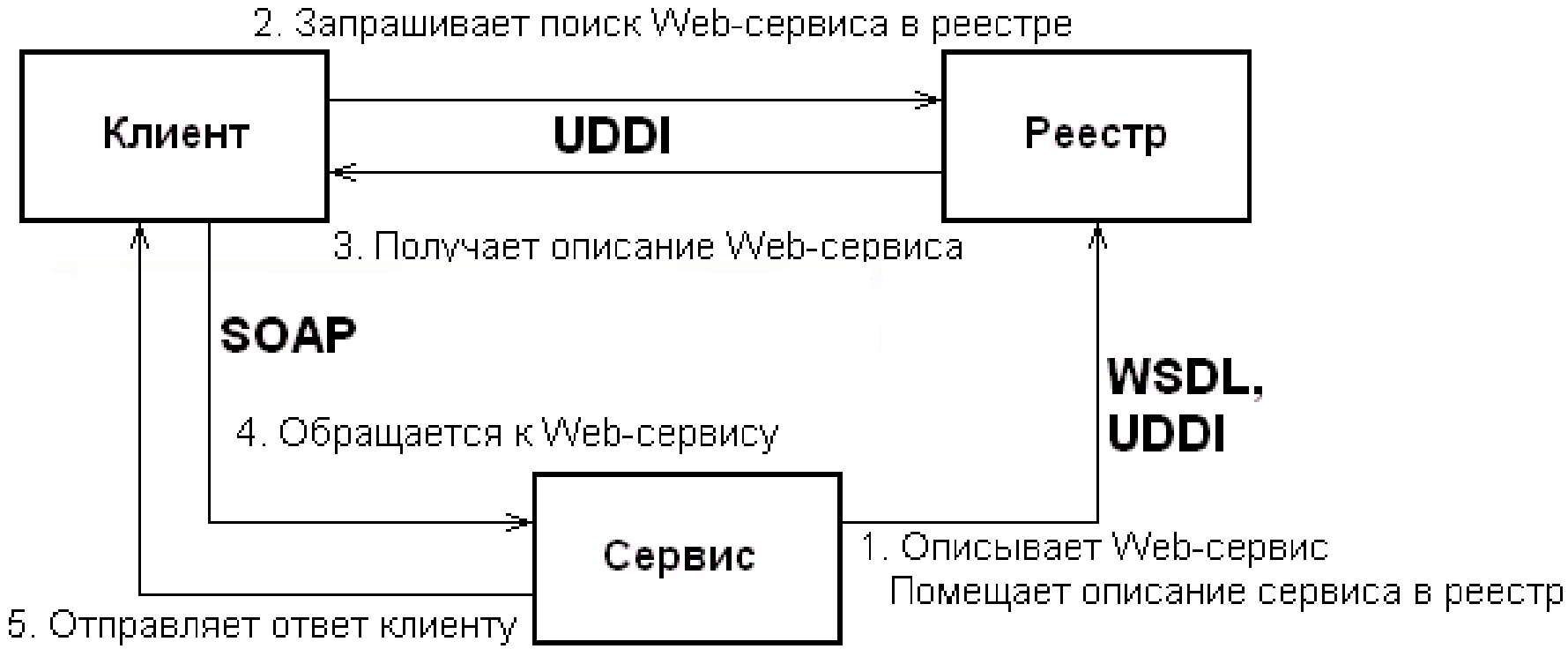
Платформа Web-сервисов разделяется на три части (с разработанными спецификациями):

- **коммуникационные протоколы**
  Простой протокол доступа к объектам Simple Object Access Protocol (**SOAP**, *www.w3.org/2000/xp*), который поддерживает связь между Web-сервисами

- **описания сервисов**
  Язык Web Services Description Language (*www.w3.org/TR/wsdl.htm,* **WSDL**), который дает формальное описание Web-сервисов.

- **поиск услуг**
  Каталог Universal Description, Discovery, and Integration (**UDDI**, *www.uddi.org*), который представляет собой регистр описаний Web-сервисов.

Взаимодействие составных частей Web-сервисов



1) Кроме сервиса и его клиента в процессе применения Web-сервисов участвует реестр. **Реестр** - общедоступное хранилище описаний Web-сервисов. Реестры поддерживаются фирмами и организациями (например, IBM и Microsoft). Разработчик, желающий сделать свой сервис **общедоступным**, составляет описание своего сервиса на языке **WSDL** и помещает в реестр (публикует).
   Взаимодействие владельца сервиса с реестром определяется стандартом **UDDI**
2) Клиент, используя **UDDI**, запрашивает в реестре поиск Web-сервисов с нужными ему функциями
3) Клиент получает описания Web-сервисов, удовлетворяющих параметрам его запроса.
4) Выбрав подходящий Web-сервис, клиент обращается к нему по протоколу **SOAP**.
5) Web-сервис выполняет запрошенную клиентом функцию и отправляет результат клиенту по протоколу **SOAP**

Реестр не является обязательным компонентом применения и публикации Web-сервисов

Технология Web-сервисов

Все Web-сервисы базируются на применении открытых стандартов и протоколов ключевыми из которых являются следующие:

- **Язык XML** (Extensible Markup Language) - расширяемый язык разметки.
- **Язык WSDL** (Web Services Description Language) - язык описания Web-сервисов.
- **SOAP** (Simple Object Access Protocol) - простой протокол доступа к объектам.
- **Технология UDDI** (Universal Description, Discovery and Integration) - реестр Web-сервисов и механизм поиска.

Язык XML

Все стандарты Web-сервисов основаны на языке XML

XML — расширяемый язык разметки, представляющий собой свод общих синтаксических правил.
XML — текстовый формат, предназначенный для хранения структурированных данных, для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки.

Структура XML файла:

- *Заголовок* (определяет версию xml и кодировку <?xml version="1.0" encoding="UTF-8">)
- *Описание типа документа* (не обязательно, но его наличие будет означать валидность документа)
- *Дерево xml элементов* с одним корневым элементом
- *Комментарии*

Коммуникации: SOAP

Протокол SOAP на базе XML предназначен для передачи сообщений и удаленных вызовов процедур (remote procedure call, RPC)

Состав SOAP сообщения:

Конверт – определяет начало и конец сообщения

Заголовок – дополнительные атрибуты сообщения

Тело сообщения – содержит XML-данные сообщения

```
<SOAP:Envelope xmlns:SOAP=
    «http://schemas.xmlsoap.org/soap/envelope/»>
  <SOAP:Header>
    <!- content of header goes here ->
  </SOAP:Header>
  <SOAP:Body>
    <!- content of body goes here ->
  </SOAP:Body>
</SOAP:Envelope>
```

Описание: WSDL

WSDL – язык описания web-сервисов – это формат XML-схем, определяющий расширенную структуру описания интерфейсов web-сервисов.

WSDL – это общий способ представления передаваемых в сообщениях типов данных, указывающий действия, которые должны быть выполнены с данным сообщением и согласно которому сообщения привязываются к сетевым транспортам.

WSDL написан на XML

WSDL является XML-документом

WSDL используется для описания web-сервисов

WSDL также используется для определения расположения web-сервисов

Структура документа WSDL

<definitions>

<types>**определение типов.....**</types>

<message>**определение сообщения....**</message>

<portType>**определение порта.....** </portType>

<binding> **определение связей.....** </binding>

</definitions>

Документ WSDL может также содержать другие элементы, например элементы расширения и элемент service, который позволяет объединить вместе в одном отдельном документе WSDL определения нескольких web-сервисов.

Элементы, описывающие семантику:

<definition> - корневой элемент В этом элементе, как правило, определяются пространства имен, используемые в определении.

<types> - описываются все типы данных, используемые в документе

<message> - описывает каждое сообщение, с которым работает web-сервис: запрос, ответ, пересылку документов.

<binding> - осуществляет «привязку" web-сервиса и определяет для каждого сообщения конкретизацию сетевого протокола, способы кодировки и упаковки послания

<portType> -- самый важный элемент WSDL.

Он определяет **web-сервис**, выполняемые им **операции** и допустимые **сообщения.**

Порт определяет точку монтирования к web-сервису. Его можно сравнить с библиотекой функций (модулем, классом), а каждую операцию можно сравнить с функцией в традиционном языке программирования.

Типы операций

- Однонаправленный (One-way) Операция может принимать сообщение, но не будет возвращать ответ
- Запрос-ответ (Request-response) Операция может принимать запрос и возвратит ответ. Самый распространенный
- Вопрос-ответ (Solicit-response) Операция может послать запрос и будет ждать ответ
- Извещение (Notification) Операция может послать сообщение, но не будет ожидать ответ

Поиск: UDDI

UDDI (реестр web -сервисов и механизм поиска) — инструмент для расположения описаний web - сервисов, который регистрирует и публикует информацию о web-сервисах.

UDDI – это каталог web -сервисов. Подобен каталогу «Желтые страницы» (Универсальное Описание, Обнаружение и Интеграция) - проект, который помогает бизнес-партнерам находить друг друга

Свойства UDDI

- UDDI -- платформонезависимая структура описания, поиска и интегрирования бизнес-сервисов в Internet.
- UDDI предназначен для хранения информации о web-сервисе
- В UDDI интерфейс web-сервиса описывается с помощью WSDL
- UDDI имеет связь с SOAP
- UDDI встроен в платформу Microsoft .NET

UDDI основан на общем наборе индустриальных стандартов, таких как HTTP, XML, XML Schema и SOAP. UDDI предоставляет инфраструктуру для основанных на web -сервисах программных сред, как для публично доступных сервисов, так и сервисов, разработанных для функционирования в локальных сетях компаний.

Цели UDDI

Основной целью UDDI реестра является представление данных и метаданных о web-сервисах.

Реестр UDDI, независимо от области применения (внутри организации или как публично доступный сервис), предоставляет стандартные механизмы для классификации, каталогизации и управления web-сервисами.

Коммерческие организации и другие провайдеры сервисов могут использовать UDDI реестр для представления информации о своих сервисах, независимо от типа использования сервиса.

Проблемы решаемые через UDDI

- **найти реализации сервисов**, которые основаны на общем абстрактном определении интерфейса;
- **найти поставщиков сервисов**, которые классифицированы в соответствии с известным классификатором или системой идентификации;
- **определить уровень безопасности и транспортные протоколы**, поддерживаемые данным сервисом;
- **найти сервисы** по ключевому слову;
- **запомнить техническую информацию о сервисе** и обновлять эту информацию в процессе работы приложения.

Базовые структуры UDDI

businessEntity (бизнес-сущность):

эта структура охватывает информацию о бизнесе или компании и используется компанией для описания и публикации информации о себе и о предлагаемых услугах. **businessEntity** - структура данных высшего уровня. В businessEntity выражаются описания услуги и техническая информация.

businessService (сервис информация):

эта структура обозначает услуги или бизнес-процессы, обеспечиваемые businessEntity. Обычно она содержит уникальный ключ, используемый для представления этой услуги, а также понятное человеку имя службы.

SOA – выводы

SOA – это компонентная модель, состоящая из отдельных функциональных модулей приложений, называемых **сервисами**, имеющих определенные согласно некоторым общим правилам интерфейсы и механизм взаимодействия между собой.

SOA – это архитектура приложений, в рамках которой все функции приложения являются **независимыми сервисами с четко определенными интерфейсами**, которые можно вызывать в нужном порядке с целью формирования бизнес-процессов.

SOA – это не технология, а **способ проектирования и организации информационной архитектуры** и бизнес-функциональности

2. Виртуализация

Обзор

В широком смысле, **виртуализация** - сокрытие настоящей реализации какого-либо процесса или объекта от истинного его представления для того, кто им пользуется. Иными словами, происходит отделение представления от реализации чего-либо.

В IT **виртуализация** - абстракция вычислительных ресурсов и предоставление пользователю системы, которая «инкапсулирует» (скрывает в себе) собственную реализацию.

Проще говоря, пользователь работает с удобным для себя представлением объекта, и для него не имеет значения, как объект устроен в действительности.

Виды виртуализации

виртуализация платформ

- Полная эмуляция
- Частичная эмуляция
- Частичная виртуализация
- Паравиртуализация
- Виртуализация уровня ОС
- Виртуализация уровня приложений.

виртуализация ресурсов

- Объединение, агрегация и концентрация компонентов
- Кластеризация компьютеров и распределенные вычисления
- Разделение ресурсов
- Инкапсуляция

- **виртуализация платформ**
  Продуктом этого вида виртуализации являются **виртуальные машины** - некие программные абстракции, запускаемые на платформе реальных аппаратно-программных систем.

- **виртуализация ресурсов**
  Данный вид виртуализации преследует своей целью комбинирование или упрощение представления аппаратных ресурсов для пользователя и получение неких пользовательских абстракций оборудования, пространств имен, сетей и т.п.

Виртуализация платформ

Под виртуализацией платформ понимают **создание программных систем на основе существующих аппаратно-программных комплексов**, зависящих или независящих от них.

Система, предоставляющая аппаратные ресурсы и программное обеспечение, называется хостовой (**host**), а симулируемые ей системы – гостевыми (**guest**).

Есть несколько видов виртуализации платформ.

Виды виртуализации платформ

Полная эмуляция (симуляция).

- Виртуальная машина **полностью** виртуализует все аппаратное обеспечение при сохранении гостевой ОС в неизменном виде.
- Такой подход позволяет эмулировать различные аппаратные архитектуры.

Например, можно запускать виртуальные машины с гостевыми системами для x86-процессоров на платформах с другой архитектурой (например, на RISC-серверах компании Sun).

Примеры продуктов для создания эмуляторов: Bochs, PearPC, QEMU (без ускорения), Hercules Emulator.

Частичная эмуляция (нативная виртуализация).

- Виртуальная машина виртуализует **лишь необходимое количество аппаратного обеспечения**, чтобы она могла быть запущена изолированно.
- Такой подход позволяет запускать гостевые ОС, разработанные **только для той же архитектуры**, что и у хоста.
- Таким образом, несколько экземпляров гостевых систем могут быть запущены одновременно.
- Такой подход позволяет существенно увеличить быстродействие гостевых систем по сравнению с полной эмуляцией и широко используется в настоящее время.
- К **минусам** данного вида виртуализации можно отнести зависимость виртуальных машин от архитектуры аппаратной платформы.

Примеры продуктов для нативной виртуализации: VMware Workstation, VMware Server, VMware ESX Server, Virtual Iron, Virtual PC, VirtualBox, Parallels Desktop и другие.

Частичная виртуализация, а также «виртуализация адресного пространства» («address space virtualization»).

- Виртуальная машина симулирует несколько экземпляров аппаратного окружения (но не всего), в частности, пространства адресов.
- При таком виде виртуализации пользователем не создаются виртуальные машины, а происходит изоляция каких-либо процессов на уровне ОС.

Паравиртуализация.

- Аппаратное обеспечение не симулируется, используется специальный программный интерфейс (API) для взаимодействия с гостевой операционной системой.
- Гостевые ОС подготавливаются для исполнения в виртуализированной среде. Код, касающийся виртуализации, локализуется непосредственно в ОС, для чего их ядро незначительно модифицируется.
- Модификация ядра - недостаток метода (возможна, если гостевые ОС имеют открытые исходные коды).
- ОС взаимодействует с программой гипервизора, который предоставляет ей гостевой API.
- Достоинство: паравиртуализация предлагает производительность почти как у реальной не виртуализированной системы

Виртуализация уровня ОС.

- Виртуализация физического сервера на уровне ОС в целях создания нескольких защищенных виртуализованных серверов на одном физическом.

Виртуализация уровня приложений.

- Приложение помещается в контейнер с необходимыми элементами для своей работы: файлами реестра, конфигурационными файлами, пользовательскими и системными объектами. В результате получается приложение, не требующее установки на аналогичной платформе.

Виды виртуализации

виртуализация платформ

- Полная эмуляция
- Частичная эмуляция
- Частичная виртуализация
- Паравиртуализация
- Виртуализация уровня ОС
- Виртуализация уровня приложений.

виртуализация ресурсов

- Объединение, агрегация и концентрация компонентов
- Кластеризация компьютеров и распределенные вычисления
- Разделение ресурсов
- Инкапсуляция

Виды виртуализации ресурсов

Объединение, агрегация и концентрация компонентов.

- Организация нескольких физических или логических объектов в пулы ресурсов (группы), представляющих удобные интерфейсы пользователю

Кластеризация компьютеров и распределенные вычисления (grid computing).

- Включает в себя техники, применяемые при объединении множества отдельных компьютеров в глобальные системы (метакомпьютеры), совместно решающие общую задачу.

Разделение ресурсов (partitioning).

- Происходит разделение какого-либо одного большого ресурса на несколько однотипных объектов, удобных для использования. В сетях хранения данных это называется зонированием ресурсов («zoning»).

Инкапсуляция.

- это процесс создания системы, предоставляющей пользователю удобный интерфейс для работы с ней и скрывающей подробности сложности своей реализации. *Например,* использование центральным процессором кэша для ускорения вычислений не отражается на его внешних интерфейсах.

3. Облачные вычисления и технологии (Cloud Computing)

Определения

Облачные вычисления – это доступ по требованию к виртуализированным компьютерным ресурсам, размещенным за пределами вашего собственного центра обработки данных, разделяемым другими, простым в использовании и доступным через Интернет.

Облачные вычисления – это модель, обеспечивающая удобный сетевой доступ по требованию к общей массе настраиваемых вычислительных ресурсов (таких как сети, серверы, приложения, системы хранения, услуги), которые могут быть быстро предоставлены, выпущены с минимальными усилиями руководства или взаимодействием с поставщиком услуг.

Облачные вычисления – технология распределённой обработки данных, в которой компьютерные ресурсы и мощности предоставляются пользователю как Интернет-сервис (*Википедия*).

Предпосылки развития

- расширение пропускной способности Интернета
- появление Salesforce.com в 1999 году. Данная компания стала 1-й компанией, предоставившей доступ к своему приложению через сайт
- разработка облачного web-сервиса компанией Amazon в 2002 году
- В 2006г. Amazon запустила сервис под названием Elastic Compute cloud (EC2), как web-сервис который позволял его пользователям запускать свои собственные приложения
- создание компанией Google, платформы Google Apps для web - приложений
- развитие аппаратного обеспечения : создание многоядерных процессоров и увеличения емкости накопителей информации

Вывод: основой для создания и быстрого развития облачных вычислений послужили крупные интернет сервисы (Google, Amazon**) и технический прогресс. Это говорит о том что появление облачных вычислений было всего лишь делом времени.**

Каким образом технический прогресс позволил облачным системам стать доступнее?

- **Развитие многоядерных процессоров** привело к:
  - увеличению производительности;
  - снижению стоимости оборудования;
  - снижению энергопотребления облачной системы.
- **Увеличение емкостей носителей информации, снижение стоимости хранения** 1 Мб информации позволило:
  - безгранично увеличить объемы хранимой информации;
  - снизить стоимость обслуживания хранилищ информации.
- **Развитие многопоточного программирования** привело к:
- эффективному использованию вычислительных ресурсов многопроцессорных систем;
- гибкому распределению вычислительных мощностей облаков.

- **Развитие технологий виртуализации** привело к:
  - созданию ПО позволяющего создавать виртуальную инфраструктуру не зависимо от количества предоставленных аппаратных ресурсов;
  - легкости масштабирования, наращивания систем;
  - уменьшению расходов на администрирование облачных систем;
  - доступности виртуальной инфраструктуры через сеть Интернет.
- **Увеличение пропускной способности** привело к:
  - увеличению скорости работы с облачными системами;
  - сниженюе стоимости Интернет трафика для работы с большими объемами информации;
  - проникновению облачных вычислений в массы.

Достоинства облачных вычислений

- доступность – облака доступны всем, из любой точки, где есть Интернет
- низкая стоимость – основные факторы снизившие стоимость использования облаков следующие:
  - снижение расходов на обслуживания виртуальной инфраструктуры;
  - оплата фактического использования ресурсов;
  - использование облака на правах аренды ;
  - развитие аппаратной части вычислительных систем, в связи с чем снижение стоимости оборудования.
- гибкость — неограниченность вычислительных ресурсов (память, процессор, диски).
- надежность – надежность «облаков», особенно находящихся в специально оборудованных ЦОД, очень высокая.
- безопасность – «облачные» сервисы имеют достаточно высокую безопасность при должном ее обеспечении.
- большие вычислительные мощности – вы как пользователь «облачной» системы можете использовать все ее вычислительные способности.

Недостатки облачных вычислений

- постоянное соединение с сетью – для получения доступа к услугам «облака» необходимо постоянное соединение с сетью Интернет.
- конфиденциальность – конфиденциальность данных хранимых на публичных «облаках» в настоящее вызывает много споров, так как в настоящее время нет технологии которая бы гарантировала 100% конфиденциальность хранимых данных.
- надежность – с уверенностью можно сказать что если вы потеряли информацию хранимую в “облаке”, то вы ее потеряли навсегда.
- безопасность – “облако” само по себе является достаточно надежной системой, однако при проникновении на него злоумышленник получает доступ к огромному хранилищу данных. Еще один минус - использование систем виртуализации, в которых в качестве гипервизора используются ядра стандартных ОС таких, как Linux, Windows и др., что позволяет использовать вирусы.
- дороговизна оборудования – для построения собственного облака компании необходимо выделить значительные материальные ресурсы.

Характеристики

- Масштабируемость
- Гибкость (неограниченность вычислительных ресурсов)
- Самообслуживание
- Повсеместный доступ
- Полная виртуализация
- Сравнительное постоянство
- Измеряемые услуги
- Многоарендность
- Масштабируемость отдельных приложений
- Надежность (особенно в ЦОД)

Виды услуг предоставляемые облачными системами

- **все как услуга (Everything as a Service);**
  пользователю предоставлено все от программно аппаратной части и до управлением бизнес процессами

- **инфраструктура как услуга (Infrastructure as a service);**
  Пользователю предоставляется компьютерная инфраструктура, обычно виртуальные платформы (компьютеры) связанные в сеть

- **платформа как услуга (Platform as a service);**
  Пользователю предоставляется компьютерная платформа, с установленной операционной системой возможно и с ПО

- **программное обеспечение как услуга (Software as a service);**
  Позиционируется как «программное обеспечение по требованию

- **аппаратное обеспечение как услуга (Hardware as a Service);**
  Пользователю услуги предоставляется оборудование, на правах аренды

- **рабочее место как услуга (Workplace as a Service);**
  Используется компанией для организации рабочих мест своих сотрудников

- **данные как услуга (Data as a Service);**
  Пользователю предоставляется дисковое пространство

- **безопасность как сервис (Security as a Service).**
  Пользователю предоставляется возможность быстро развертывать, продукты позволяющие обеспечить безопасное использование веб-технологий, безопасность электронной переписки, безопасность локальной системы. Это позволяет экономить на развертывании и поддержании своей собственной системы безопасности.

Виды облаков

- Частные
- Публичные
- Гибридные

- Внутренние
- Внешние

- **Частное облако** — это безопасная ИТ-инфраструктура, контролируемая и эксплуатируемая в интересах одной организации. Организация может управлять частным облаком самостоятельно или поручить эту задачу внешнему подрядчику.

- **Публичное облако** — это ИТ-инфраструктура, используемая одновременно множеством компаний и сервисов. Пользователи данных облаков не имеют возможности управлять и обслуживать данное облако, вся ответственность по этим вопросам возложена на владельца данного облака.

- **Гибридное облако** — это ИТ- инфраструктура использующая лучшие качества публичного и частного облака, при решении поставленной задачи. Часто такой тип облаков используется, когда организация имеет сезонные периоды активности

Определения действующих лиц в Cloud Computing

Актор	Определение
Облачный Потребитель Cloud Consumer	Лицо или организация, поддерживающая бизнес-отношения и использующая услуги Облачных Провайдеров.
Облачный Провайдер Cloud Provider	Лицо, организация или сущность, отвечающая за доступность облачной услуги для Облачных Потребителей.
Облачный Аудитор Cloud Auditor	Участник, который выполняет независимую оценку (assessment) облачных услуг, обслуживания информационных систем, производительности и безопасности реализации облака.
Облачный Брокер Cloud Broker	Сущность, управляющая использованием, производительностью и предоставлением облачных услуг, а также устанавливающая отношения между Облачными Провайдерами и Облачными Потребителями.
Облачный Оператор Связи Cloud Carrier	Посредник, предоставляющий услуги подключения и транспорт (услуги связи) облачных услуг от Облачных Провайдеров к Облачным Потребителям.

Взаимодействие между действующими лицами в СС

Сценарии использования

Сценарий 1: участие облачного брокера во взаимодействии потребителя с провайдером

Сценарий 2: участие облачного оператора в предоставлении провайдером услуг

Сценарий 3: участие облачного аудитора в оценке предоставления провайдером услуг

Концептуальная диаграмма действующих лиц в СС

Облачный провайдер

Безопасность (Security) и Приватность (Privacy):

- аутентификация и авторизация
- доступность
- конфиденциальность
- управление идентификацией
- мониторинг безопасности и обработка инцидентов
- управление политиками безопасности
- сбор, обработка, передача, использование и хранение персональных данных и информации

Возможные уязвимости CC(Cloud Computing)

- Неправомерное и нечестное использование облачных технологий
- Небезопасные программные интерфейсы (API)
- Внутренние нарушители
- Уязвимости в облачных технологиях
- Потеря или утечка данных
- Кража персональных данных и неправомерный доступ к сервису
- Прочие уязвимости

Руководство безопасностью

- всеобъемлющий контроль над управлением изменениями, образами и инцидентами
- доступ к отчетам по инцидентам для компаний-пользователей среды и к данным журналов регистрации и аудита
- наблюдаемость
- возможность аудита силами сторонних организаций

Люди и идентификационная информация

- несанкционированный доступ должен быть заблокирован
- контроль над каждым привилегированным пользователем
- объединенное управление идентификационной информацией
- быстрая регистрация новых пользователей
- возможность единого входа в систему

Вопросы?

Достоинства облачных вычислений

- Экономическое преимущество для пользователя
- Увеличение производительности
- Увеличение эффективности IT инфраструктуры
- Уменьшение количества проблем с обслуживанием
- Уменьшение затрат на приобретаемое ПО
- Постоянное обновление программ
- Увеличение доступных вычислительных мощностей
- Неограниченный объем хранимых данных
- Независимость от ОС
- Улучшение совместимости форматов документов
- Простота совместной работы группы пользователей
- Повсеместный доступ к документам
- Всегда самая последняя и свежая версия
- Доступность с различных устройств
- Дружелюбие к природе
- Устойчивость данных к потере или краже

Недостатки облачных вычислений

- Постоянное соединение с сетью Интернет
- Необходимость быстрого доступа в Интернет
- Повышенные временные затраты
- Недоступность программ или отдельных свойств
- Безопасность данных может быть под угрозой
- Безвозвратная потеря данных