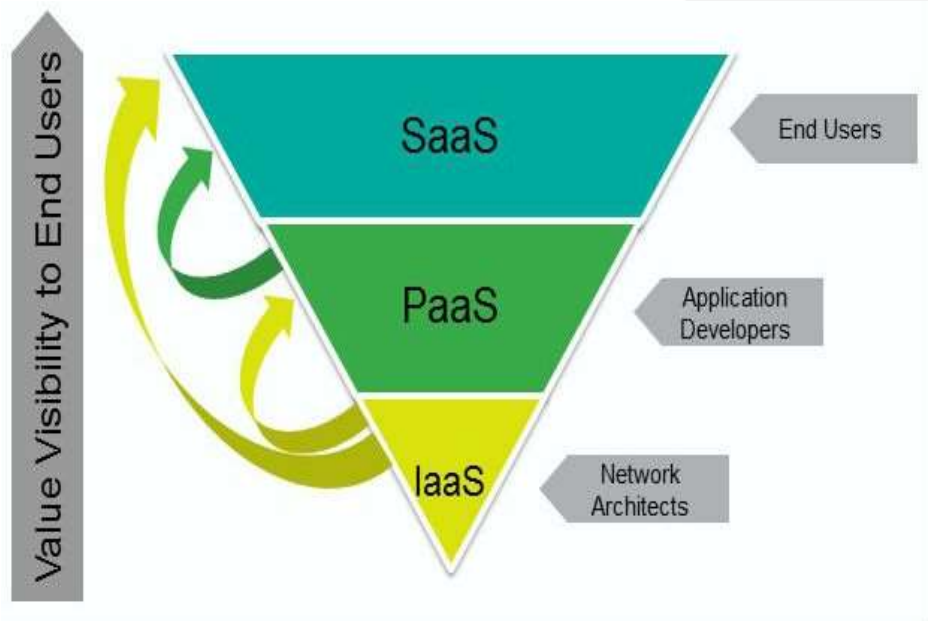
Exploring Cloud Computing Stack

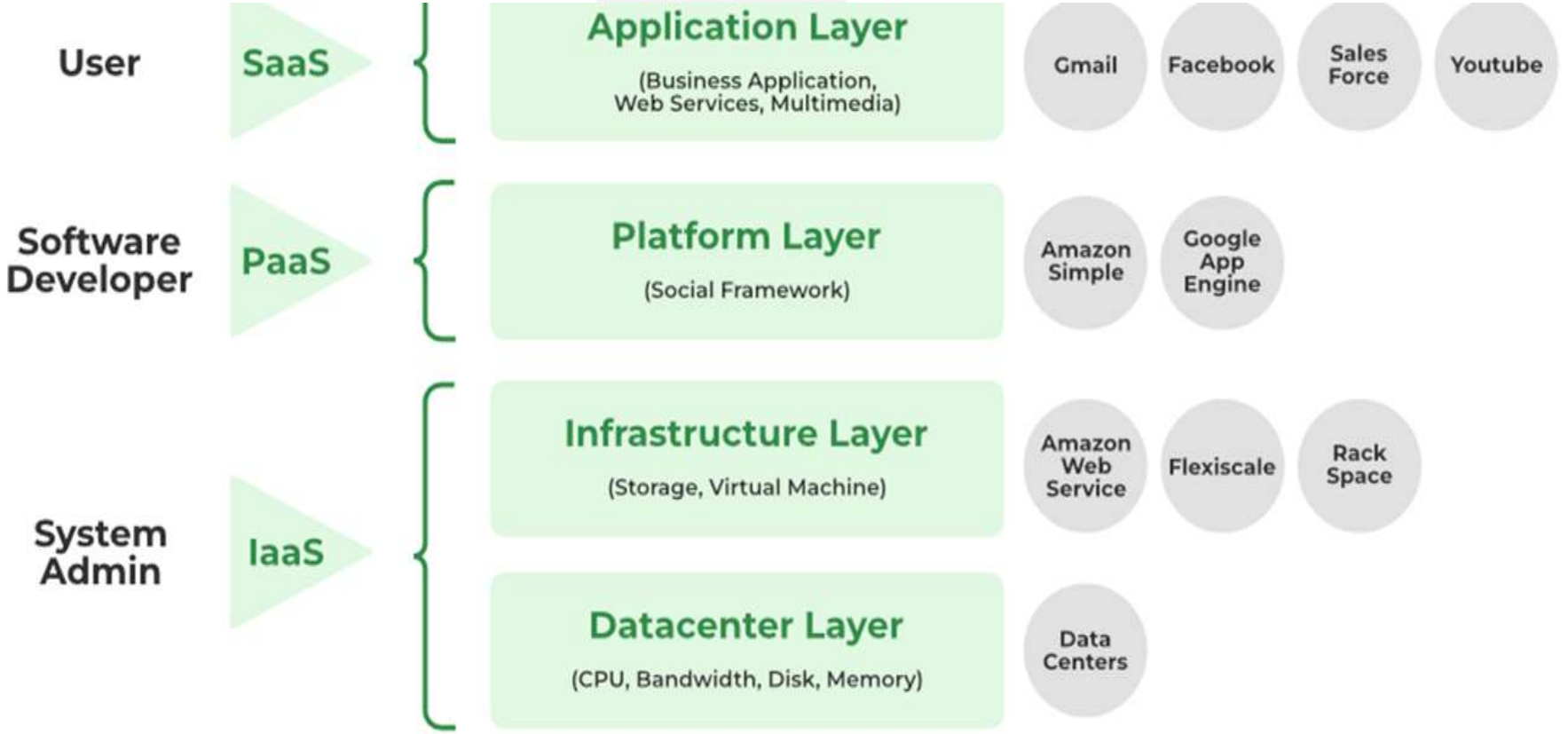
- Cloud computing has commonly been referred to as a “stack” because it typically encompasses many different types of services which have been built on top of each other under a “cloud”.
- Cloud computing has been defined by the NIST (National Institute of Standards and Technology) as a model that enables on-demand access to a shared resource pool consisting of servers, networks, applications, services and **cloud storage** which can be rapidly deployed with minimal management efforts.
- There are three different types of cloud computing services namely, Platform as a Service or PaaS, Software as a Service or SaaS and Infrastructure as a Service or IaaS.
- Almost everyone uses cloud services at least on customer bases.
- Nearly every online application uses cloud services in a direct or indirect manner. Companies are on the way to pushing every aspect of their business operations to the cloud and into the services-oriented methodology.
- If you use GitHub to manage your code lifecycle or you provision servers on the **cloud service providers** then you are using cloud services.

Exploring Cloud Computing Stack



- Users can make use of bulk resources which can be obtained quickly whenever there are demands. NIST has also stated some features as being essential for services to be regarded as **cloud hosting services**.
- On-demand services that end users can sign up for obtain.without.delays.
- Wide network access because users can access such services through multiple platforms like laptops, desktops and mobiles.
- Measured services because users will pay according to what they use; so, billing is offered as a utility service.
- Elasticity and scalability to cope with increases in demand.
- The SaaS applications have been designed for end users and they are offered across the Internet.
- The PaaS refers to sets of tools or services designed for coding and implementing the applications efficiently.
- IaaS refers to the software and hardware which powers it, such as, servers, networks, storage, and operating system.

Cloud Computing Stack / Layers



Cloud Service Model

Service Model	Main Access & Management Tool	Service content
 SaaS Consume	Web Browser	Cloud Applications Social networks, Office suites, CRM, Video processing
 PaaS Build On it	Cloud Development Environment	Cloud Platform Programming languages, Frameworks, PaaS Mashups editors, structured data
 IaaS Migrate On it	Virtual Infrastructure Manager	Cloud Infrastructure Compute Servers, Data Storage, Firewall, Load Balancer

Exploring SaaS

- Software as a Service or SaaS is software deployed over the web and a SaaS application can be licensed by a vendor to clients as on-demand service.
- This is made possible through subscription according to a pay-as-you-use model or free of cost where there are chances of generating revenues from channels like advertisements.
- SaaS offers internet access to commercial software and it managed from a central point.
- It is software that is offered through a “one-to-many” model and users do not have to be worried about patches and software upgrades.
- When businesses want to shift their operations to the cloud, they need to understand which applications should be shifted.
- For instance, “Vanilla” offerings wherein solutions are largely undifferentiated; for example, emails where competitors often use the same software as this basic technology is needed to conduct business but does not on.its.own.provide.a.competitive.advantage.

Exploring - SaaS

- Applications which demand Internet access like sales management software.
- Applications which involve interplay between the outside world and an organization like software for an email newsletter campaign.
- Software which sees frequent demand spikes like tax software or **billing software**
- Software which is needed for a short term like collaboration software for some project.
- SaaS should not be deployed for applications which need fast processing of real-time data or for applications where laws do not allow the data to be hosted externally or applications having an on-site solution that can cater to all organizational requirements.

Exploring - PaaS

- PaaS refers to a computing model which allows for the fast and easy creation of applications without buying or maintaining software and infrastructure for them.
- Unlike SaaS which is software delivered across the web, PaaS is the platform for creation of such software. It refers to:
- Services which develop, test, implement and maintain applications in an integrated development setting. Web-based interface creation tools for creating, altering, testing and deploying different user interfaces.
- Tools for handling billing Built-in scalability of software that includes failover and load balancing Integration with databases and web services through common standards Multitenant architecture where multiple users are using the same applications.
- Supporting development team collaborations; some PaaS solutions have project planning tools.
- PaaS is mainly used when many developers are working on one project and when outside parties have to communicate with development processes.
- It is useful for those that have existing data sources and are keen to build applications for leveraging that data. PaaS should not be sued where the application must be portable, or when proprietary languages can affect the development, or when application performance demands hardware and software customization etc.

Exploring - IaaS

- **Infrastructure as a service** or IaaS refers to cloud computing infrastructure in terms of servers and storage, operating systems and network. Instead of buying these, the client will buy the resources as an on-demand service.
- The public cloud refers to infrastructure which comprises shared resources that are deployed on self-service basis across the Internet, while private cloud refers to infrastructure which offers resources through a private network.
- Some providers even offer a combination of both these networks to produce a hybrid cloud. In IaaS, resources get distributed as services and it allows dynamic scaling. IaaS has variable costs because it follows a utility pricing model.
- You can use the IaaS when demands are changing, or for new businesses which do not have much capital for investing in hardware, or when an organization is growing very fast and scaling the hardware is challenging.
- IaaS is also beneficial when organizations face pressures to reduce capital costs and shift to operational costs and also for specific business needs or short-term infrastructure needs.
- IaaS should not be used where data storage outsourcing can become difficult with regulatory compliance or where on-site infrastructure can cater to an organization’s needs.

Exploring – IaaS, Paas,Saas

IaaS vs PaaS vs SaaS :

- The Three main **cloud computing stack** or **cloud software stack** is different from each other in many senses. But the main base at which they are differentiated is the control and the cost.
- When you have SaaS as your **cloud software stack** you lose a little control over the applications.
- This is because of the control of not only the applications but also of OS, storage as well as networking shifts to your vendor. Hence, if you are the owner of a small enterprise then SaaS as a **cloud technology stack** is the most suitable for you.
- Whereas with PaaS one gets the privilege of controlling their applications and data more than the vendor. The vendor is more responsible for managing OS, runtime, etc.
- Therefore, PaaS is better when it comes to cost. It is more suitable for enterprises that are into app development but do not keep their employees engaged in networking or running servers.
- A table that provides a concise comparison of Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) based on various key attributes:

Exploring – IaaS, PaaS,SaaS

Attribute	IaaS	PaaS	SaaS
• **Overview**	• Provides virtualized computing resources over the internet.	• Offers a platform for developers to build, deploy, and manage applications without dealing with the underlying infrastructure.	• Delivers software applications over the internet without the need for installation or maintenance.
• **Responsibility**	• Users manage applications, data, runtime, middleware, and OS.	• Platform provider manages runtime, middleware, OS, and infrastructure. Users focus on application development and data.	• Everything, including application, runtime, data, middleware, OS, and infrastructure, is managed by the service provider.
• **Scalability**	• Highly scalable with the ability to add or remove resources as needed.	• Offers automatic scalability based on demand. Users don't need to worry about infrastructure scaling.	• Scalability is entirely managed by the service provider. Users typically have limited control over scaling.
• **Flexibility**	• Users have control over the entire infrastructure stack, providing high flexibility.	• Offers a predefined platform, limiting flexibility compared to IaaS.	• Least flexibility as the entire application is provided as a service. Customization options are minimal.
• **Development Time**	• Longer development time as users need to manage various infrastructure components.	• Shorter development time since infrastructure management is abstracted. Developers focus on coding and application logic.	• Shortest development time as users only need to configure and use the software.
• **Example Providers**	• Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP).	• Heroku, Google App Engine, Microsoft Azure App Service.	• Salesforce, Google Workspace, Dropbox, Microsoft Office 365

Exploring – IaaS, Paas,Saas

Cloud Computing Components:

- **Compute:** Virtual machines, containers, and serverless computing.
- **Storage:** Object storage, block storage, and file storage solutions.
- **Networking:** Connects various components and enables communication.
- **Databases:** Managed database services for various data storage needs.
- **Security:** Identity and access management, encryption, and compliance features.
- **Management Tools:** Monitoring, logging, and resource management utilities.

Exploring – IaaS, Paas,Saas

Cloud Computing Statistics:

1. Market Trends:

- The global cloud computing market is projected to reach $1 trillion by 2026 (Source: Gartner).
- **Public cloud** spending is expected to grow at a CAGR of 16.7% from 2021 to 2026 (Source: IDC).

2. Adoption Rates:

- Approximately 94% of enterprises are using some form of cloud service (Source: Flexera).
- **Hybrid cloud** adoption increased to 58% in 2021, showing a rising trend (Source: Flexera).

3. Popular Cloud Providers:

- Amazon Web Services (AWS) leads the market with a 32% share, followed by Microsoft Azure (20%) and Google Cloud Platform (9%) (Source: Synergy Research Group).

4. Security Concerns:

- **66% of IT professionals view security as their top concern in cloud adoption (Source: Cybersecurity Insiders).**
- Cloud-based DDoS attacks increased by 967% in the last year (Source: Neustar).

Exploring – IaaS, Paas,Saas

- IaaS stack cloud computing gives you control on both applications as well as over the infrastructure.
- The best part of IaaS is that the vendor will spending on physical servers, networking, and storage. It is a bit costlier than the other cloud stack in cloud computing.
- Understanding the cloud computing stack is essential for making informed decisions in adopting cloud services.
- As the industry continues to evolve, staying updated on trends and statistics will be crucial for businesses aiming to leverage the benefits of cloud computing.

Large Business Model

- For large scale business, a hybrid cloud deployment model is often considered the best option.
- Hybrid cloud allows organizations to take advantage of the cost savings and scalability of public cloud resources while maintaining control over sensitive data and applications through the use of a private cloud.
- With a hybrid cloud, organizations can use the public cloud for non-sensitive data and applications that do not require a high level of security or compliance. This can include things like email, file sharing, and **web hosting**.
- At the same time, the organization can keep sensitive data and applications in a private cloud, which is generally considered to be more secure and can be used to meet compliance requirements.
- Hybrid cloud also allows organizations to take advantage of the many services that are available in the public cloud, such as data analytics, machine learning, and Internet of Things (IoT) services, while also being able to use their own resources and infrastructure.
- This allows organizations to be more agile and responsive to changing business needs.

Large- Business Model

- A hybrid cloud is also beneficial for large scale business because it can provide greater scalability and flexibility. Resources can be easily scaled up or down as needed, depending on the requirements of the organization.
- This allows businesses to only pay for the resources they need, instead of having to invest in expensive infrastructure that may not be fully utilized.
- A hybrid cloud deployment model offers large scale business the best of both worlds by providing the **cost savings and scalability of public cloud resources,** while also allowing for the control and security of a **private cloud** making it a good fit for large scale businesses with specific security and compliance requirements.
- A cloud service provider rents out the combination of technology, infrastructure, and expertise to other companies and individuals for the purpose of cloud computing, including online storage, compute, and networking over the Internet.
- Cloud service providers own and operate multiple data centers worldwide that house the physical infrastructure required for cloud computing. These include servers, hard drives, and cooling systems.
- Anyone, anywhere, and at any time can access this cloud infrastructure by connecting to these data centers and purchasing as much capacity as they require on a pay-as-you-go basis (usage-based pricing). CSPs deliver other benefits, too.

Connecting to Cloud Server

- After deploying a new Cloud Server you'll need to pick a method for connecting to it.
- Whether you chose password authentication or an SSH key.
- **OpenSSH**
- The primary way of connecting to a Linux server should be using an SSH client.
- For Unix-based systems, including macOS, Linux, and WSL (Windows Subsystem for Linux), OpenSSH is readily available and can be utilized directly through the local terminal.
- Using OpenSSH to log in is very simple, the single command to start a connection consists of 3 parts; the application command ssh, your username, and your remote host.
- Open a terminal and type in the command below to log in using the root account. Replace the remote-host with the public IP of your server.
- **ssh root@remote-host**

Connecting to Cloud Server

- If an SSH key has been configured for use, the system will attempt to authenticate using the key.
- If this is your first time connecting to the server, you will encounter a prompt regarding the host's authenticity.
- Confirm that the IP address displayed matches your server's IP, and then accept the prompt by typing "yes". Then press enter to continue.
- If the authentication was successful, you will be logged into the server straightaway.

```
❯
❯ ssh root@194.62.98.232
The authenticity of host '194.62.98.232 (194.62.98.232)' can't be established.
ED25519 key fingerprint is SHA256:AGgAxm0FYcVQRC7vkSvqW7xCE5jD/7daWKv66d5Cgi8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Connecting to Cloud Server

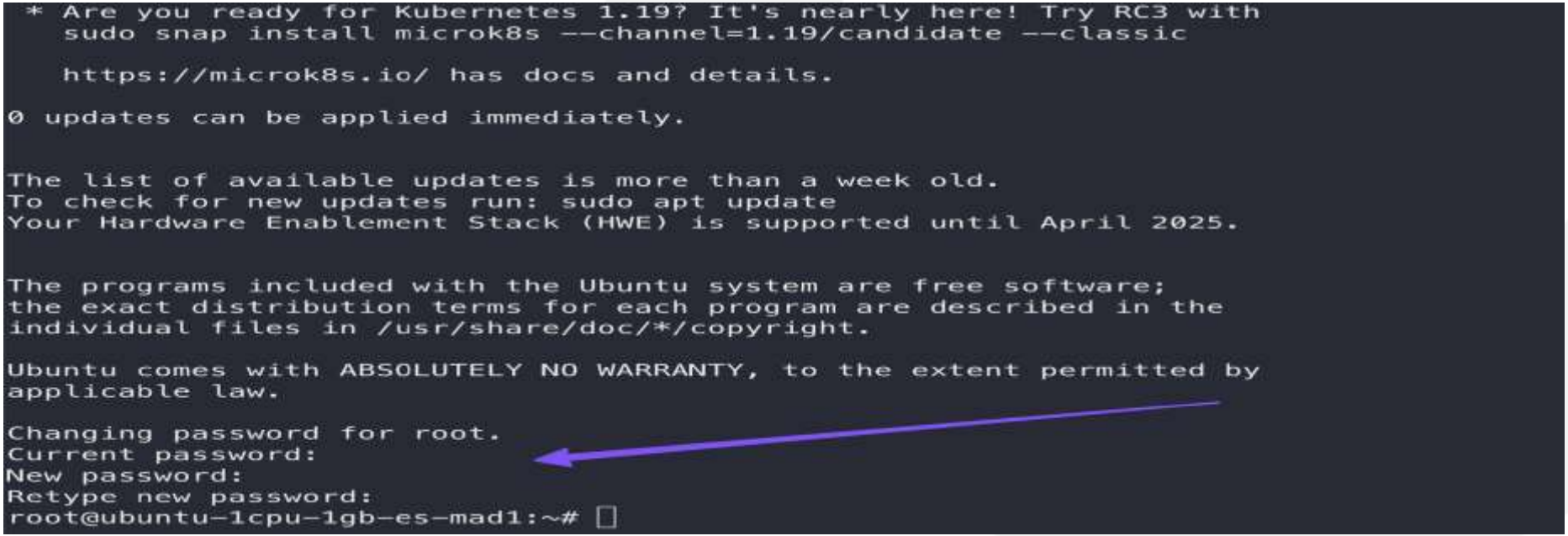

```
> ssh root@194.62.98.232
The authenticity of host '194.62.98.232 (194.62.98.232)' can't be established.
ED25519 key fingerprint is SHA256:AGgAxm0FYcVQRC7vkSvqW7xCE5jD/7daWKv66d5Cgi8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '194.62.98.232' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-53-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Feb 20 07:54:38 AM UTC 2024

  System load:             0.54248046875
  Usage of /:              11.7% of 24.53GB
  Memory usage:            33%
  Swap usage:              0%
  Processes:               90
  Users logged in:         0
  IPv4 address for eth0: 194.62.98.232
  IPv4 address for eth1: 10.7.1.128
  IPv6 address for eth2: 2a04:3542:8000:1000:607d:24ff:feef:129c


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@ubuntu-1cpu-1gb-es-mad1:~#
```

Connecting to Cloud Server

- If no SSH key was configured during the server deployment, the system will revert to password authentication.
- You'll instead be be prompted to enter your server's one-time password (OTP) during the login process.

- After successfully logging in, the system will prompt you to change your one-time password.
- You'll need to first enter the current OTP, then create a new secure and unique password. For verification, you'll be asked to enter your new password again to confirm it.

Connecting to Cloud Server



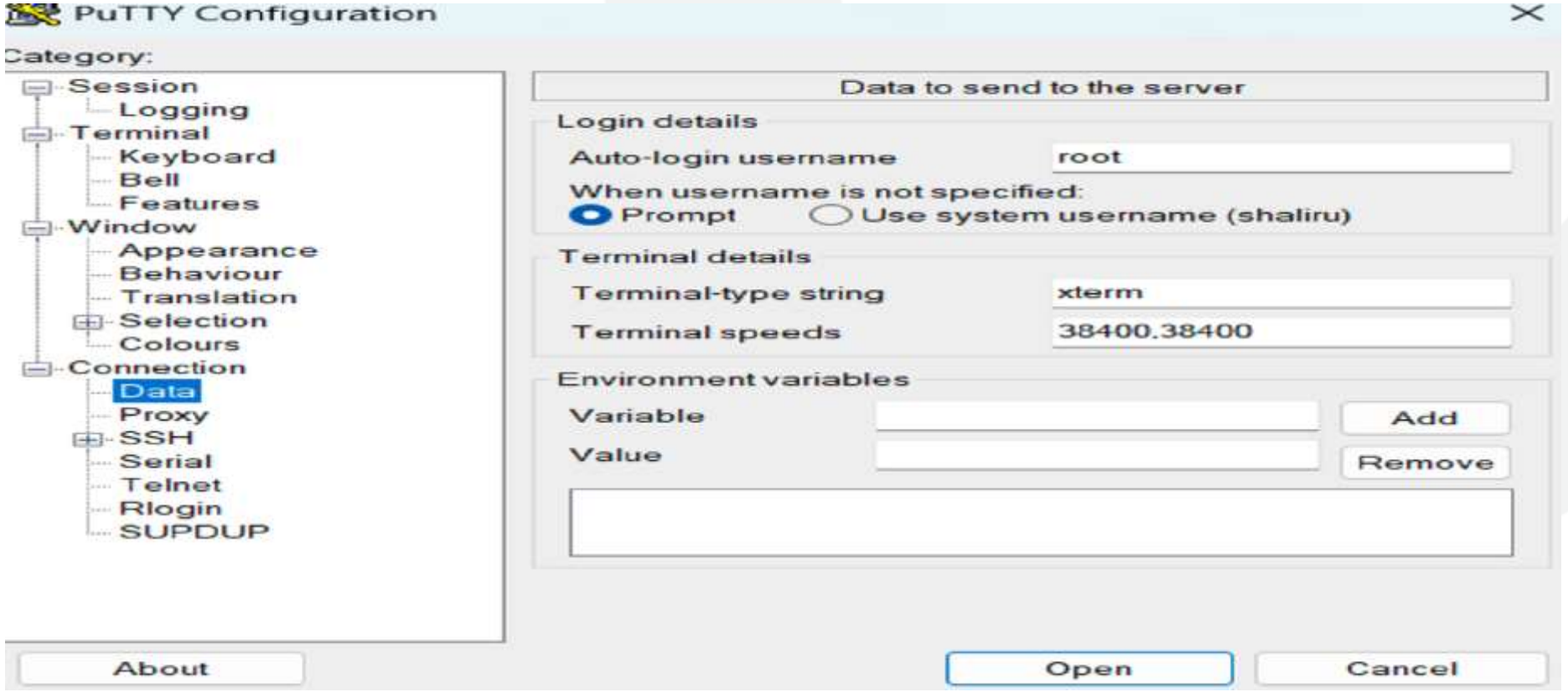
- While entering the password, note that characters will not be displayed on the screen; this is a standard security measure designed to protect your password visibility.

Connecting to Cloud Server

PuTTY

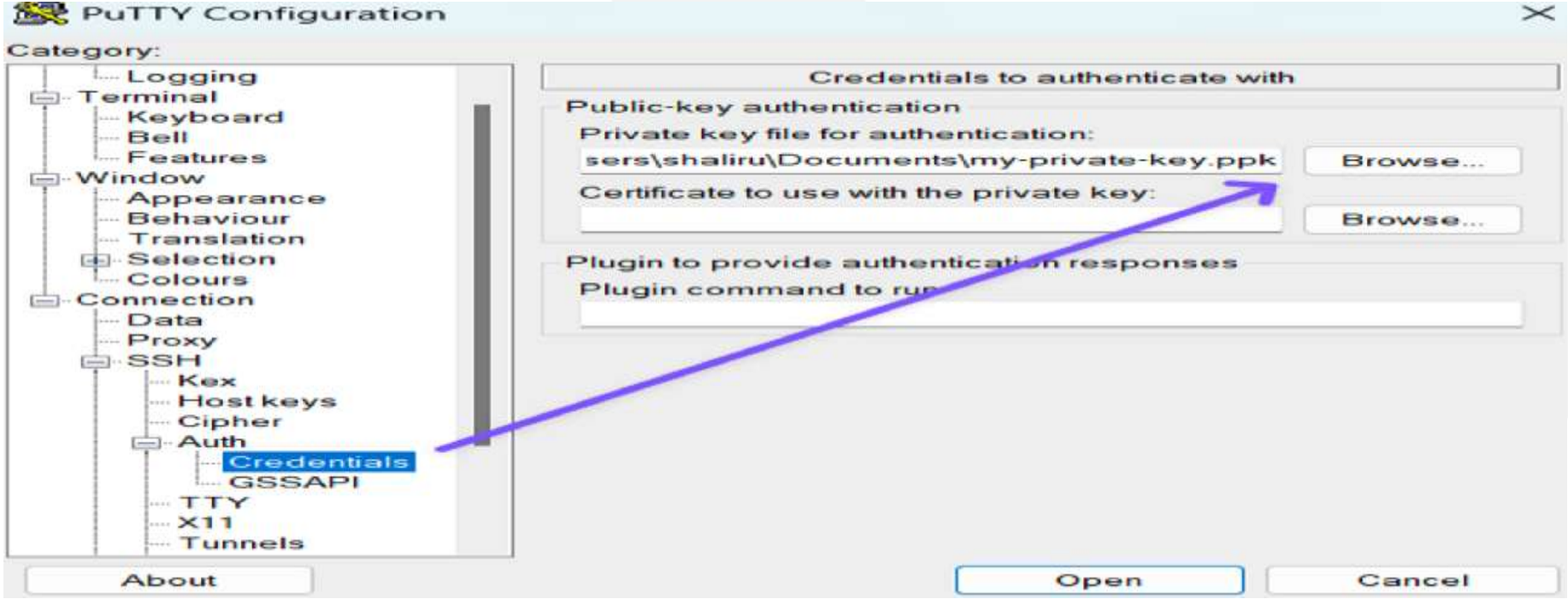
- Windows users do not have a built-in solution for SSH, but there are options for Windows as well.
- PuTTY is one of the most commonly used SSH clients for Windows. It's easy to get started with, but also offers a lot of features for advanced users.
- Open PuTTY and enter your server's public IP address in the Hostname field.
- Then, from the menu tree on the left-hand side, navigate to **Connection > Data** and enter the username that you want to use in the **Auto-login username** text box. For new Linux servers, this will usually be **root**.

Connecting to Cloud Server



Connecting to Cloud Server

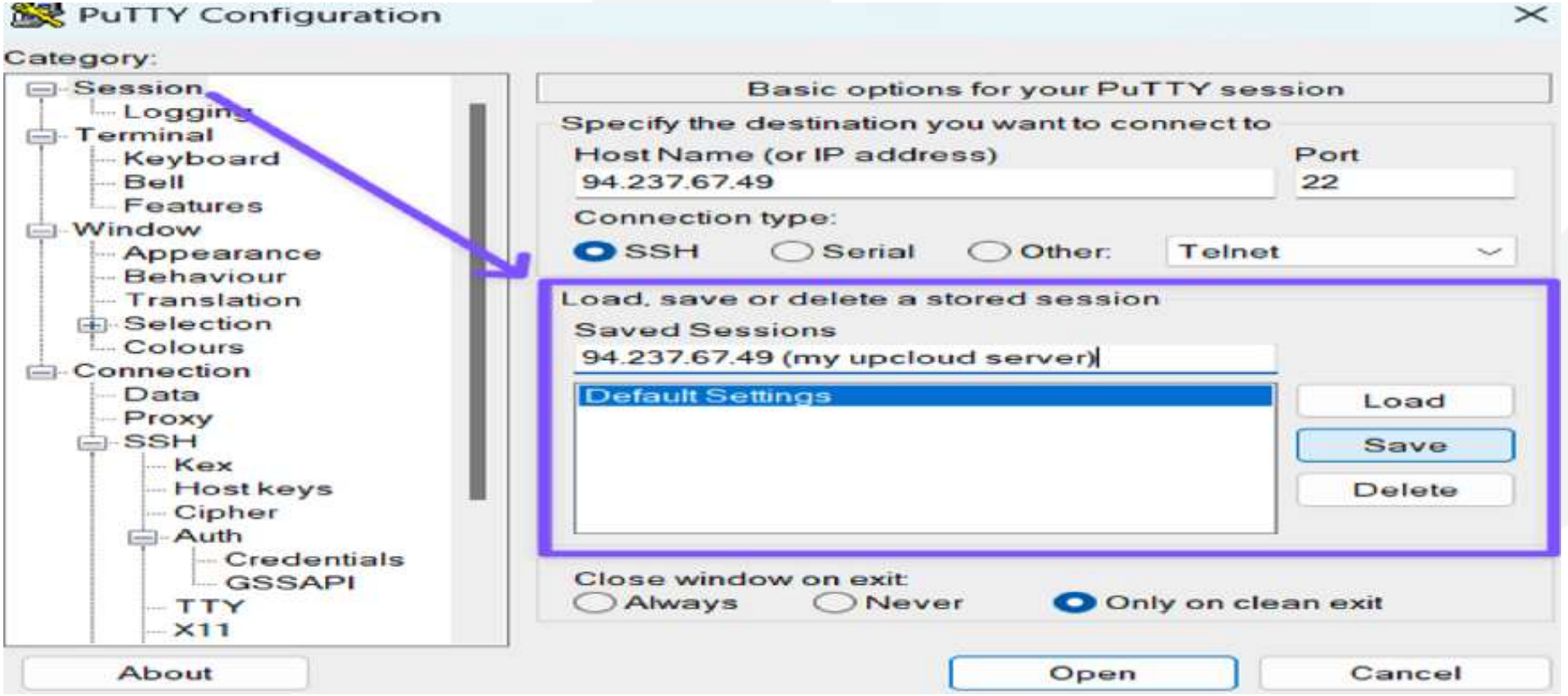
- Next, navigate to Connection > SSH > Auth > Credentials and click the Browse button to select the private key you used to deploy your cloud server.



Connecting to Cloud Server

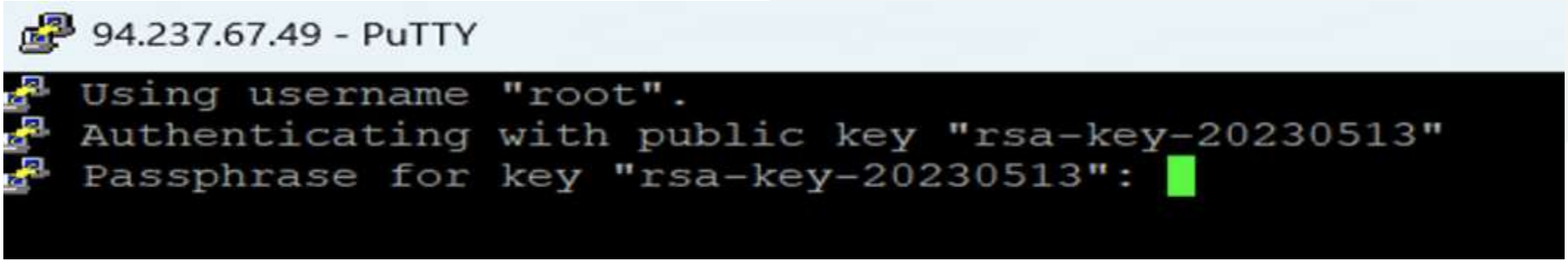
- Before proceeding, it’s a good idea to save this configuration, so you won’t have to repeat the steps each time you want to connect to your server in the future.
- To do this, navigate back to the Session screen, and under Saved Sessions, enter a name for the configuration.
- This can be anything you want, but it makes sense to give it a memorable name, such as the IP address or name of the server – or both. Click the Save button to save your changes.

Connecting to Cloud Server

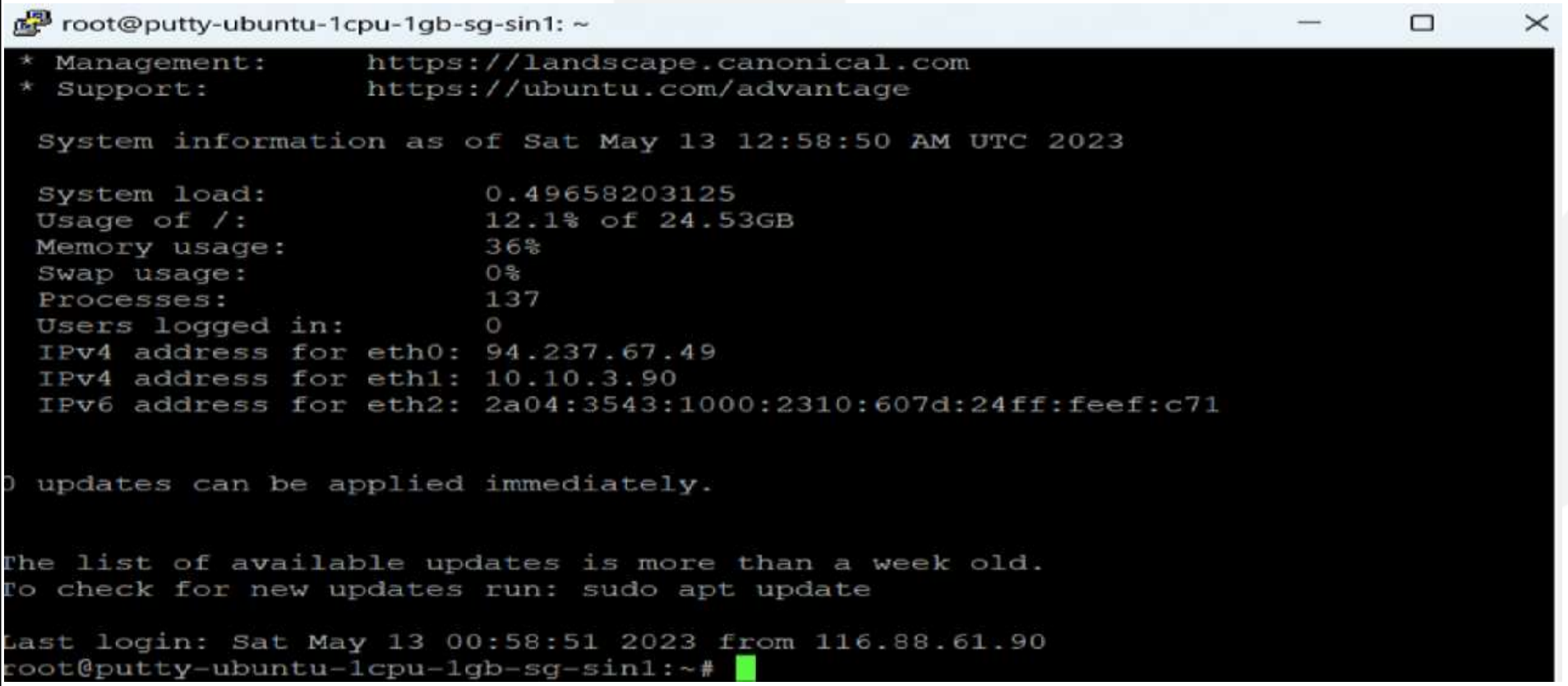


Connecting to Cloud Server

- Now you can load this configuration at any time, and it will have the IP address, username, and SSH key information pre-populated and ready to connect.
- Clicking Open will open a new terminal window prompting you to enter your key passphrase.
- Type it in and press enter to connect to your server.
- If you created your keypair without a passphrase, then you won't see this prompt. You will instead be connected to the server straightaway.



Connecting to Cloud Server



Connecting to Cloud Server

- If no SSH key was configured during the server deployment, simply enter your server's public IP address in the Hostname field, and click Open.

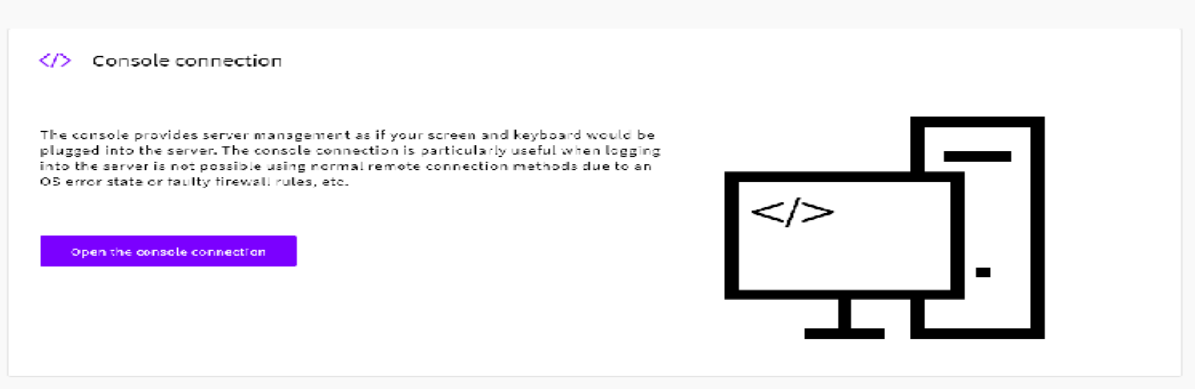
Remote desktop connection

- Windows servers employ their own remote desktop connection that allows you to operate your Windows Server just like your own desktop.
- Windows users should have the Remote Desktop Connect client installed by default. It is also available for macOS in the Mac App Store as well as on most Linux distributions through open-source alternatives such as Remmina.
- When connecting, simply enter your server IP address and authenticated it with the username Administrator and the password generated at deployment.
- Most clients support a full screen desktop experience and allow you to save user credentials for convenience, granted that your computer is sufficiently secure and not shared by other users.

Connecting to Cloud Server

Console connection

- The third option is to use the HTML5-based web console at your UpCloud Control Panel with no browser extensions or setup required.
- Although you probably want to use SSH primarily, this is a useful addition in case of faulty firewall rules, OS error states, or any other reason that prevents the usual access methods.
- Open your server settings and go to the Console tab. Then simply click the button on the left to Open the console connection. Log in using root or any other username and password stored on your Cloud Server.

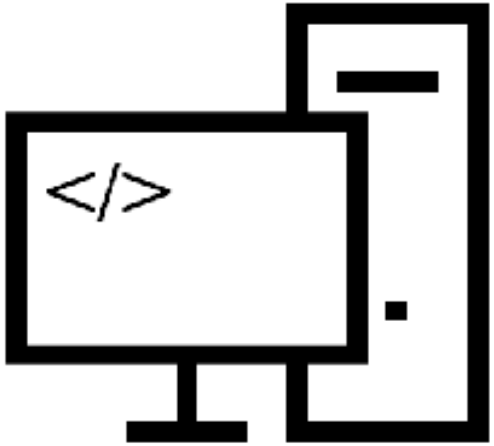


Connecting to Cloud Server

</> Console connection

The console provides server management as if your screen and keyboard would be plugged into the server. The console connection is particularly useful when logging into the server is not possible using normal remote connection methods due to an OS error state or faulty firewall rules, etc.

Open the console connection

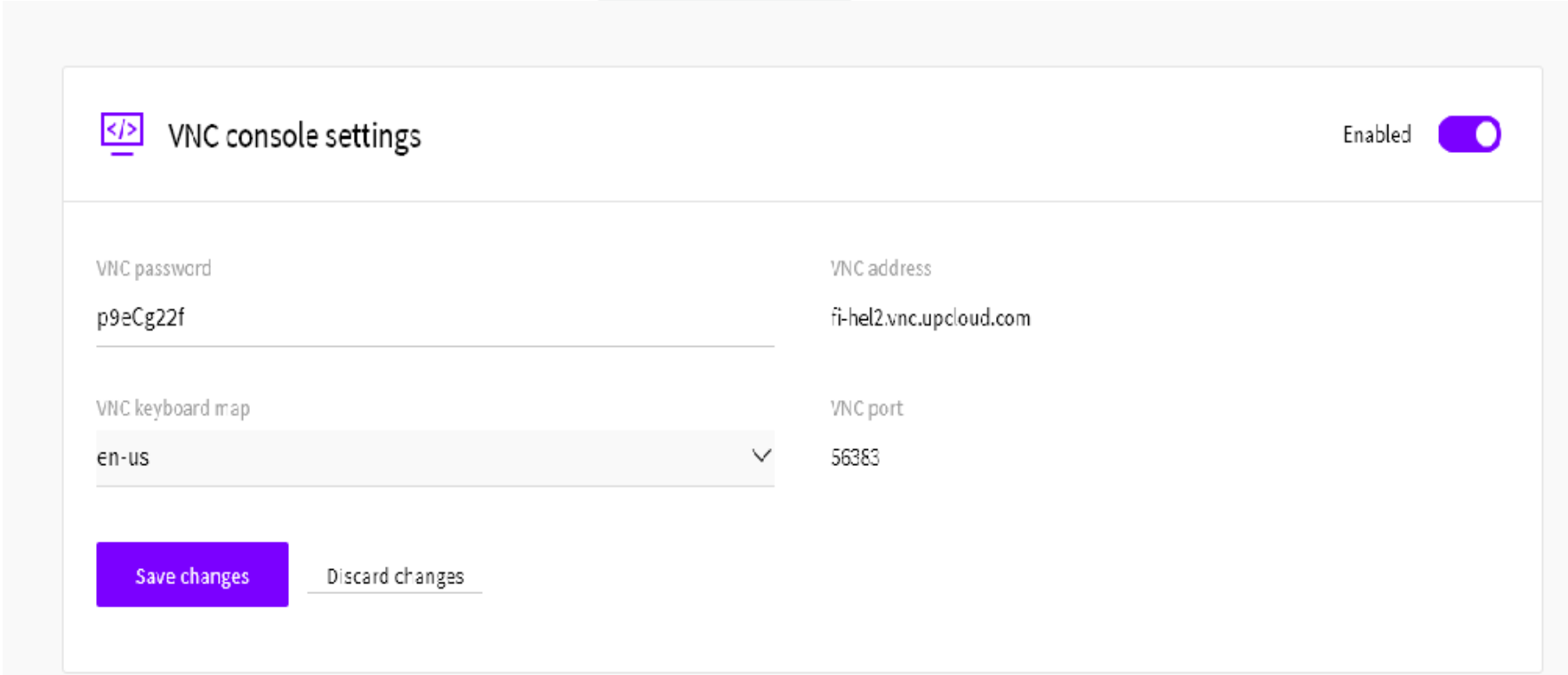


Connecting to Cloud Server

VNC Console

- UpCloud also offers optional VNC console access using your choice of VNC client. To enable a VNC connection, log in to your UpCloud Control Panel, open your server settings and go to the Console tab.
- By default the VNC service is disabled on new servers, click the toggle switch to enable the VNC connection.
- Underneath that, you will find the connection details for your server. Here is where you can set the password for VNC and change the keyboard mapping.
- Click the Save changes button to update the settings when done.
- Note that while enabling/disabling VNC console as well as changing the VNC password can be done with the Cloud Server running, changing the keyboard map will require the server to be powered down.
- Some VNC clients only ask for the hostname of your server without a second field for the port number, simply enter both the VNC address and VNC port number together separated by a colon (:) sign, for example, fi-hel2.vnc.upcloud.com:12345 to open a connection.

Connecting to Cloud Server



Connecting to Cloud Server

- Video Links :
  - https://youtu.be/HAz6UgFKlYY
  - https://www.google.com/search?sca_esv=43423dbf98504c44&rlz=1C1ONGR_enIN953IN953&q=Connecting+to+cloud+video+online&sa=X&ved=2ahUKEwjQvd6W3eKJAxWPqFYBHao9GLMQ1QJ6BAg0EAE&biw=1536&bih=730&dpr=1.25#

Service Oriented Architecture

- **Service-oriented architecture (SOA)** is a method of software development that uses software components called services to **create business applications.** Each service provides a business capability, and services can also communicate with each other **across platforms and languages.**

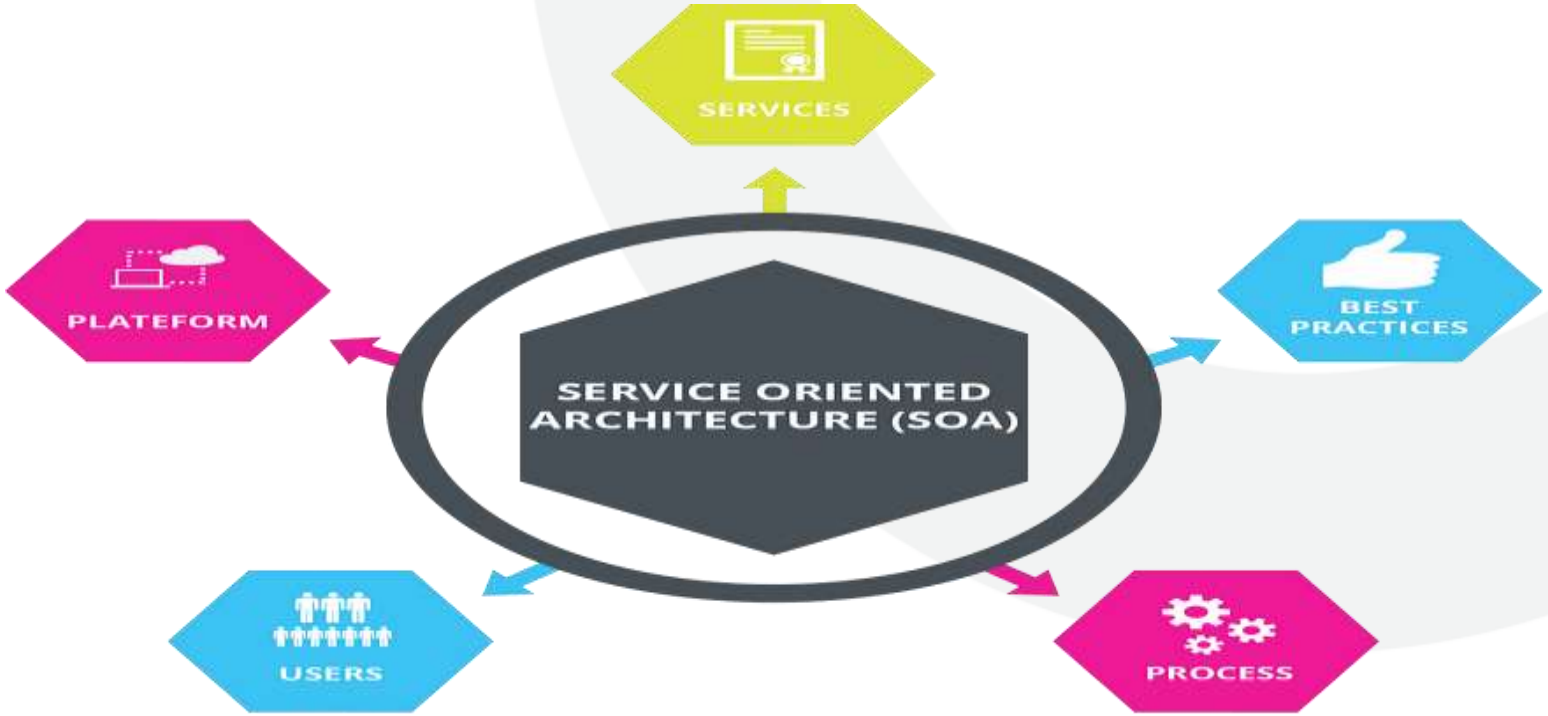
- Service Oriented Architecture (SOA) is a specification and a methodology for providing platform and language independent services for use in distributed applications.

- A service is a repeatable task within a business process, and a business task is a composition of services. SOA describes **a message passing taxonomy for a component based architecture** that provides services to clients upon demand.

- Clients access a component that complies with SOA by passing a message containing metadata to be acted upon in a standard format.

- The component acts on that message and returns a response that the client then uses for its own purpose. A common example of a message is an **XML file transported over a network protocol such as SOAP.**

Service Oriented Architecture



- This architecture does not contain executable links that require access to a specific API. The message presents data to the service, and the service responds.

- It is up to the client to determine if the service returned an appropriate result. An SOA is then seen as a method for creating an integrated process as a set of linked services. The component exposes itself as an “endpoint” (a term of art in SOA) to the client.

- The most commonly used message-passing format is an **Extensible Markup Language (XML)** document using **Simple Object Access Protocol (SOAP),** but many more are possible, including **Web Services Description Language (WSDL), Web Services Security (WSS), and Business Process Execution Language for Web Services (WS-BPEL).**

- WSDL is commonly used to describe the service interface, how to bind information, and the nature of the component’s service or endpoint.

- The **Service Component Definition Language (SCDL)** is used to define the service component that performs the service, providing the component service information that is not part of the Web service and that therefore wouldn’t be part of WSDL.

Service Oriented Architecture

- Usually service providers and service consumers do not pass messages directly to each other. Implementations of SOA employ middleware software to play the role of transaction manager (or broker) and translator.

- **Middleware can discover and list available services**, as well as potential service consumers, often in the form of a registry, because SOA describes a distributed architecture security and trust services are built directly into many of these products to protect communication.

- Middleware products also can be where the logic of business processes reside; they can be general purpose applications, industry-specific, private, or public services.

- Middleware services manage lookup requests. The **Universal Description Discovery and Integration (UDDI)** protocol is the one most commonly used to broadcast and discover available Web services, often passing data in the form of an Electronic Business **using eXtensible Markup Language (ebXML) documents.**

- Service consumers find a Web service in a broker registry and bind their service requests to that specific service; if the broker supports several Web services, it can bind to any of the ones that are useful.

- The defining concepts of **Service-Oriented Architecture vary from company to company,** there are six key parameters that over reach the broad concept of Service-Oriented Architecture.

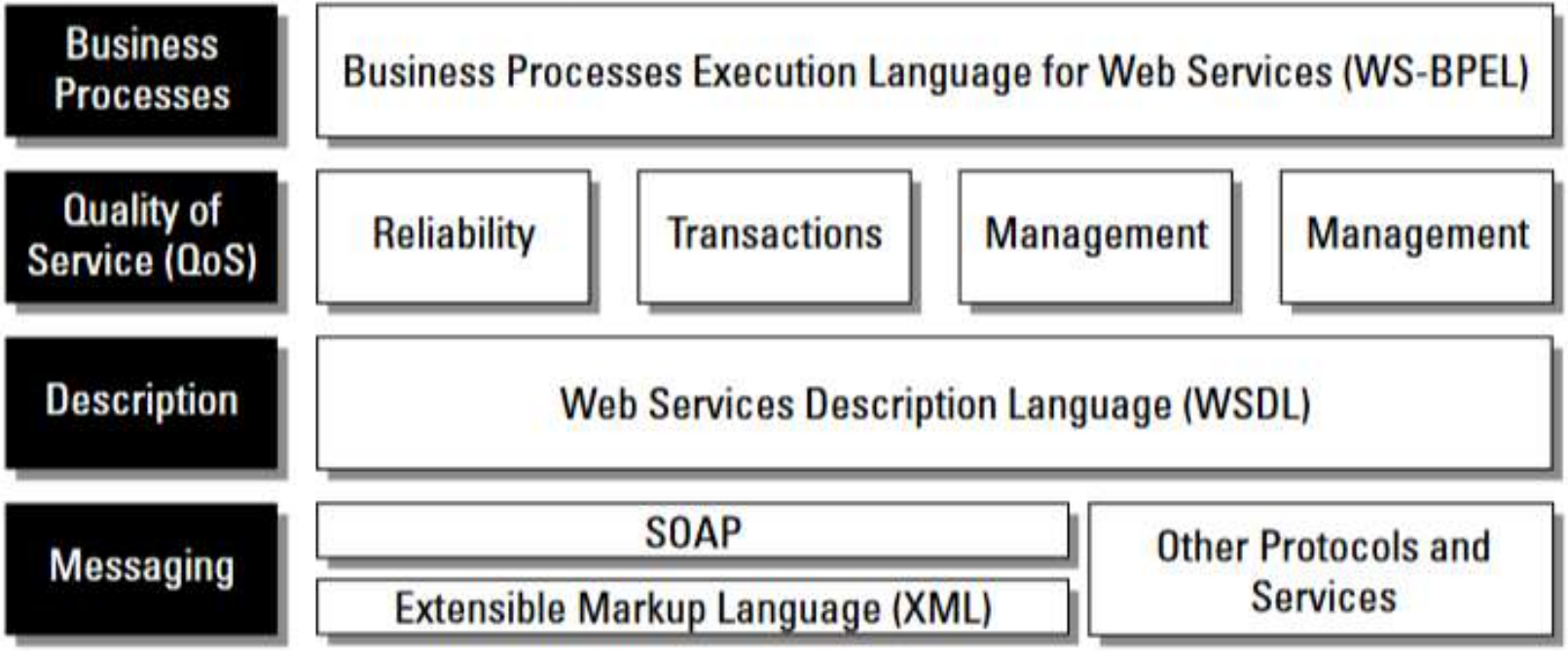
Core values include:

- Business value
- Strategic goals
- Intrinsic inter-operability
- Shared services
- Flexibility
- Evolutionary refinement


01
PEOPLE
Empower decision Makers
02
PROCESS
Align IT with business operations
SOA
Service Oriented Architecture
03
PLATFORM
Increase Operational efficiency
04
PRACTICE
Employ best practice methology

SOA-Protocol Stack

A protocol stack for SOA showing the relationship of each protocol to its function



- A protocol stack for an SOA architecture and how those different protocols execute the functions required in the Service Oriented Architecture.
- In the figure, the box labeled Other Services could include Common Object Request Broker Architecture (CORBA),Representational State Transfer (REST), Remote Procedure Calls (RPC).
- Distributed Common Object Model (DCOM), Jini, Data Distribution Service (DDS), Windows Communication Foundation (WCF), and other technologies and protocols. It is this flexibility and neutrality that makes SOA so singularly useful in designing complex applications.
- This architecture does not contain executable links that require access to a specific API. The message presents data to the service, and the service responds. It is up to the client to determine if the service returned an appropriate result.
- An SOA is then seen as a method for creating an integrated process as a set of linked services. The component exposes itself as an “endpoint” (a term of art in SOA) to the client.
- The most commonly used message-passing format is an Extensible Markup Language (XML) document using Simple Object Access Protocol (SOAP), but many more are possible, including WebServices Description Language (WSDL), Web Services Security (WSS), and Business Process Execution Language for Web Services (WS-BPEL). WSDL is commonly used to describe the service.

- Interface, how to bind information, and the nature of the component's service or endpoint. The Service Component Definition Language (SCDL) is used to define the service component that performs the service, providing the component service information that is not part of the Web service and that therefore wouldn't be part of WSDL.

- Note whatever protocol is used to negotiate a transaction, the formal definition of the transaction is referred to as the "contract." Indeed, the notion of a contract implies a certain level of service that is available to clients and that may be part of any paid service in SOA.

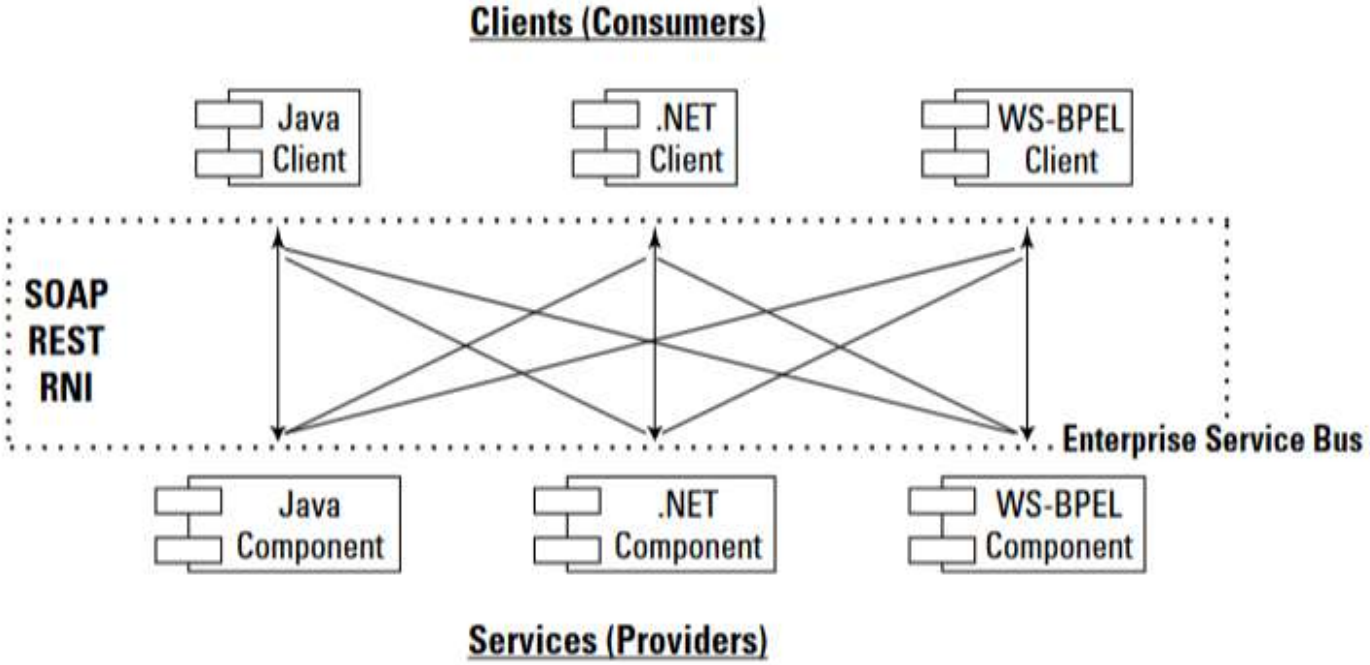
- Figure shows a protocol stack for an SOA architecture and how those different protocols execute the functions required in the Service Oriented Architecture.

- SOA provides the framework needed to allow clients of any type to engage in a request response mechanism with a service. The specification of the manner in which messages are passed in SOA.

- SOA requires the use of an orchestrator or broker service to ensure that messages are correctly transacted. SOA makes no other demands on either the client (consumer) or the components (provider) of the service; it is concerned only with the interface or action boundary between the two.

SOA-Client Construction

SOA allows for different component and client construction, as well as access to each using different protocols.

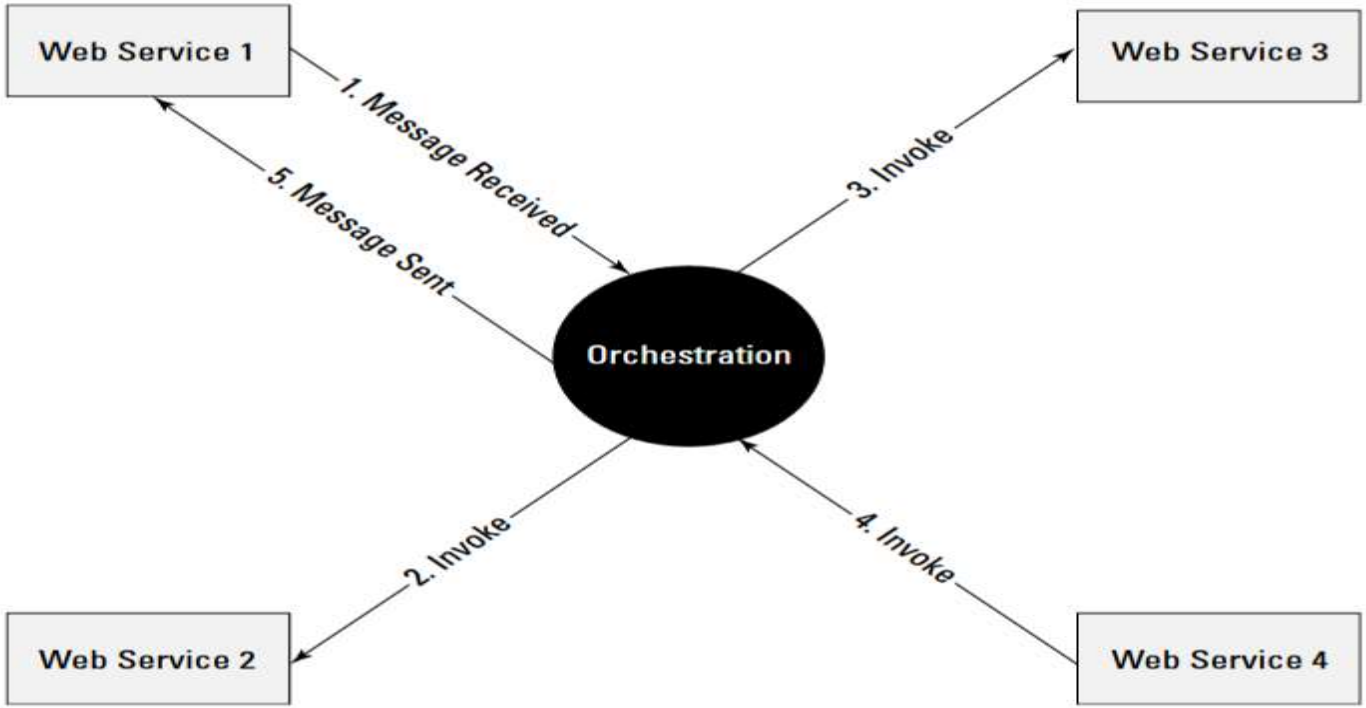


SOA- Choreography

- Figure shows how components of different types can communicate using different protocols as part of SOA.
- When you combine Web services to create business processes, the integration must be managed.
- Two main methods are used to combine Web services: orchestration and choreography.
- In orchestration, a middleware service centrally coordinates all the different Web service operations, and all services send messages and receive messages from the orchestrator.
- A compound business process that uses **choreography has no central coordination function**. In choreography, each Web service that is part of a business process is aware of when to process a message and with what client or component it needs to interact with.
- **Choreography** is a collaborative effort where the logic of the business process is pushed out to the members who are responsible for determining which operations to execute and when to execute them, the structure of the messages to be passed and their timing, and other factors.

Orchestration

An orchestrated business process uses a central controlling service or element, referred to as the orchestrator, conductor, or less frequently, the coordinator.

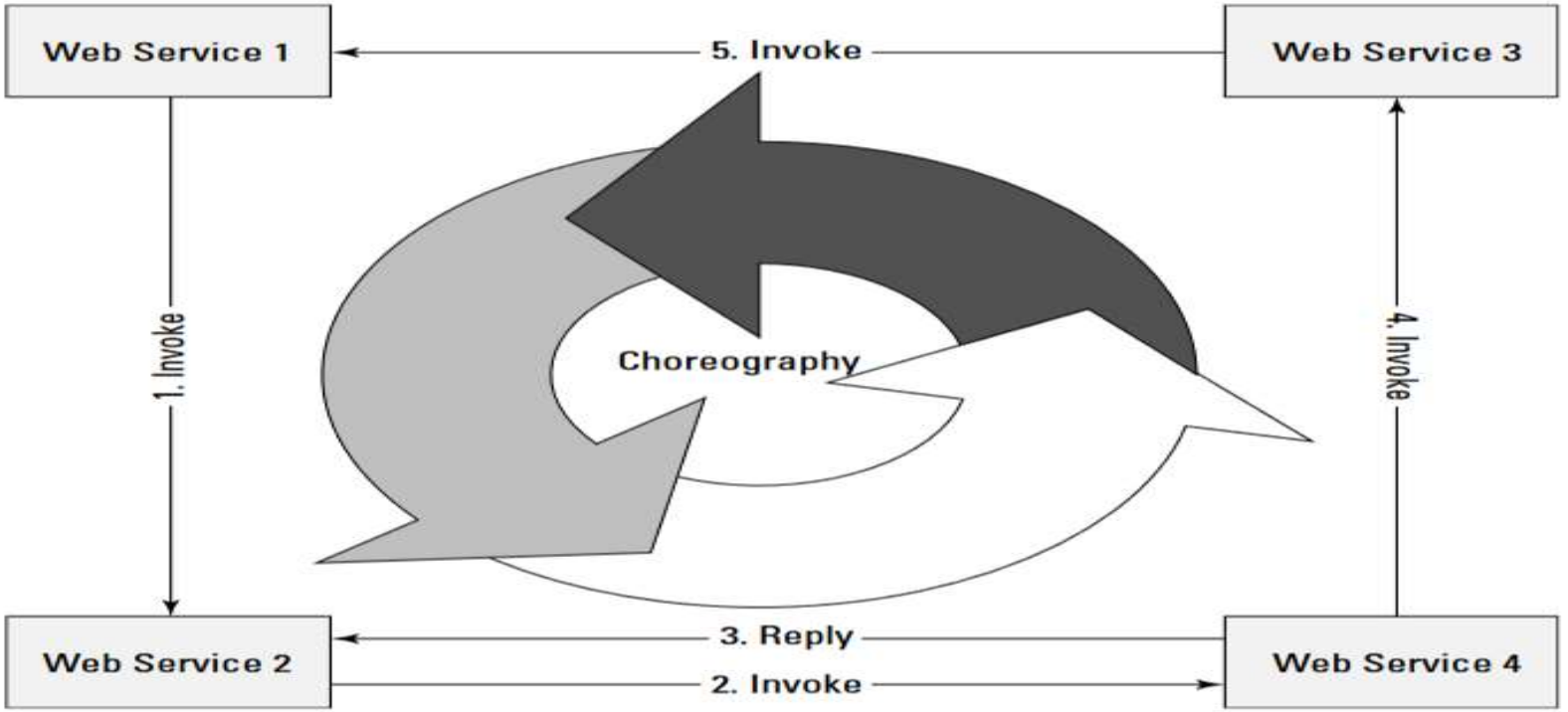


SOA- Choreography

- Most mature SOA implementations favour orchestration over choreography for a number of reasons.
- With orchestration a single central service manages the various processes, and changes to the business logic can be made in that one location.
- The integration of Web services into the architecture is easier than with choreography because these services don't need to know anything about the business process.
- Centralizing the business logic also makes it easier to put error handling mechanisms in place and to account for, manage, and analyze events that occur outside the business process that relate to a part of the process.
- Event handling is part of event-driven SOA or SOA 2.0, which extends Service Oriented Architecture to include both random and scheduled events that are triggered by a business process outside of a business process.
- One way of performing orchestration is through the use of an Enterprise Service Bus or ESB. An ESB provides a middleware software layer for event management with a messaging infrastructure.

SOA- Choreography

With choreography, business process execution is a cooperative affair.



Event-driven SOA or SOA 2.0

- Event-driven SOA or SOA 2.0 is an extension of the Service Oriented Architecture to respond to events that occur as a result of business processes or perhaps cause and influence a business process.
- For example, in a business process, sales at a certain Web site are processed. If the business process recognizes the rate at which sales are occurring, it could perform an analysis to determine what events might influence the buying decision.
- This is the sort of analysis that event-driven SOA is meant to address. SOA 2.0 can allow low-level events to trigger a business process, correlate events with information contained in the SOA design, inhibit a business process if the appropriate events don't appear, or invoke a reaction or response based on a trigger.
- To perform these tasks in SOA 2.0, a **Causal Vector Engine (CVE)** with some built-in artificial intelligence must be added to the SOA design.
- Events are analyzed in terms of event sequences, event relationships, and event timing to establish whether a certain condition has occurred.
- The CVE then determines how to react to the condition using a set of rules that are built into the system. Many CVE systems display events in a console in different contexts so that an observer can

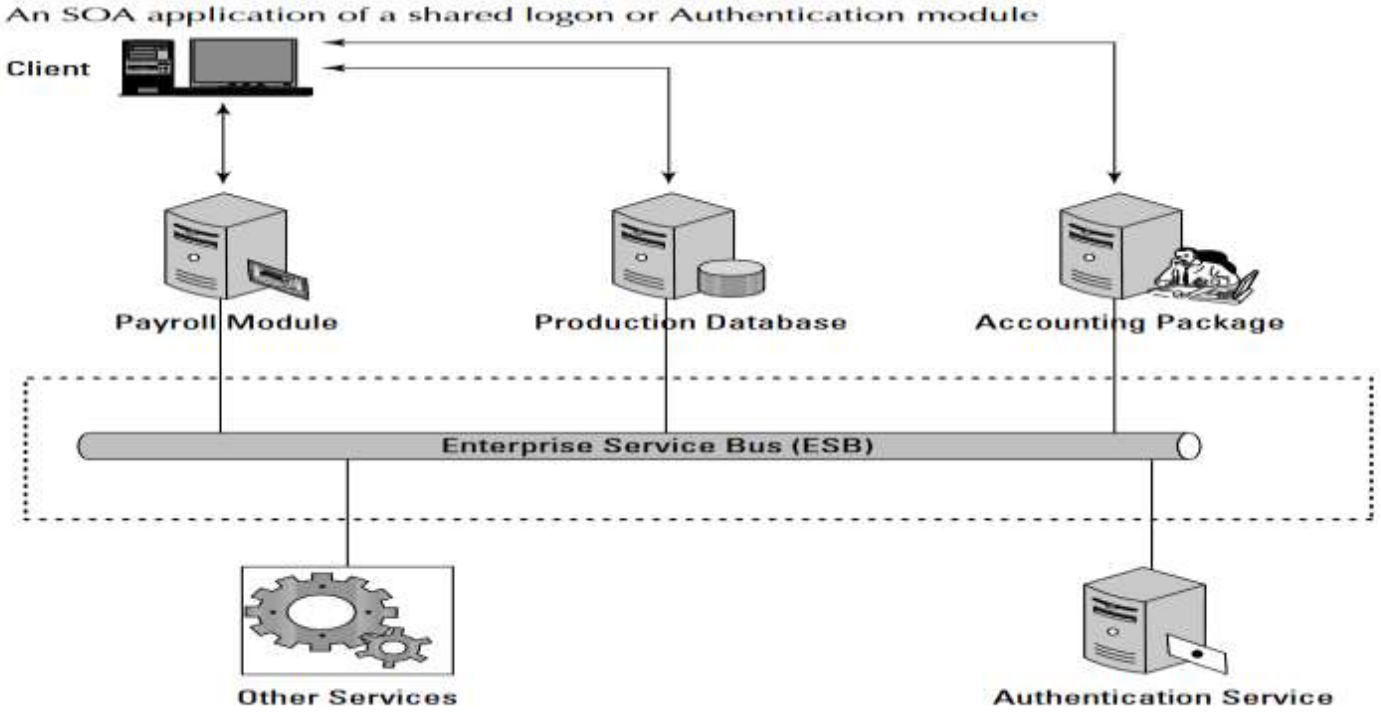
Event-driven SOA or SOA 2.0

- The CVE then determines how to react to the condition using a set of rules that are built into the system. Many CVE systems display events in a console in different contexts so that an observer can analyze the display and take appropriate actions.
- A CVE application may include the ability to query event data in the same way that a stock ticker or trading application can query trading data.
- The CVE application provides the same kind of heartbeat and correlation functionality that a stock trading application does.
- From the standpoint of the service requestor or consumer (client), the client simply needs to know the form required to initiate the action of the provider (service) and how to interpret the results returned from the service provider.
- The nature of the component's processing is unknown, the location where the processing is done is unknown, and the various operating systems and applications involved are unknown.

Event-driven SOA or SOA 2.0

- The client is responsible for validating that the service returned the results that were expected. The SOA component is essentially a black box to the client.
- SOA makes no demands of the component other than to conform to the rules of a standard end point.
- This level of abstraction offers operational advantages to Web service providers in that components can be continually upgraded, replaced, or moved to improve efficiencies without disrupting the clients that depend on those services, and the Quality of Service for that service can be accurately measured and delivered.
- SOA provides the rules so each application can access the authentication module in its own way, as required.
- What you gain with SOA is the ability to add significant capabilities with a fraction of the cost or effort and to federate applications if you desire.
- What you lose with SOA is the ability to perform fundamental customization of the service itself when that service is provided by a third party.

The Enterprise Service Bus



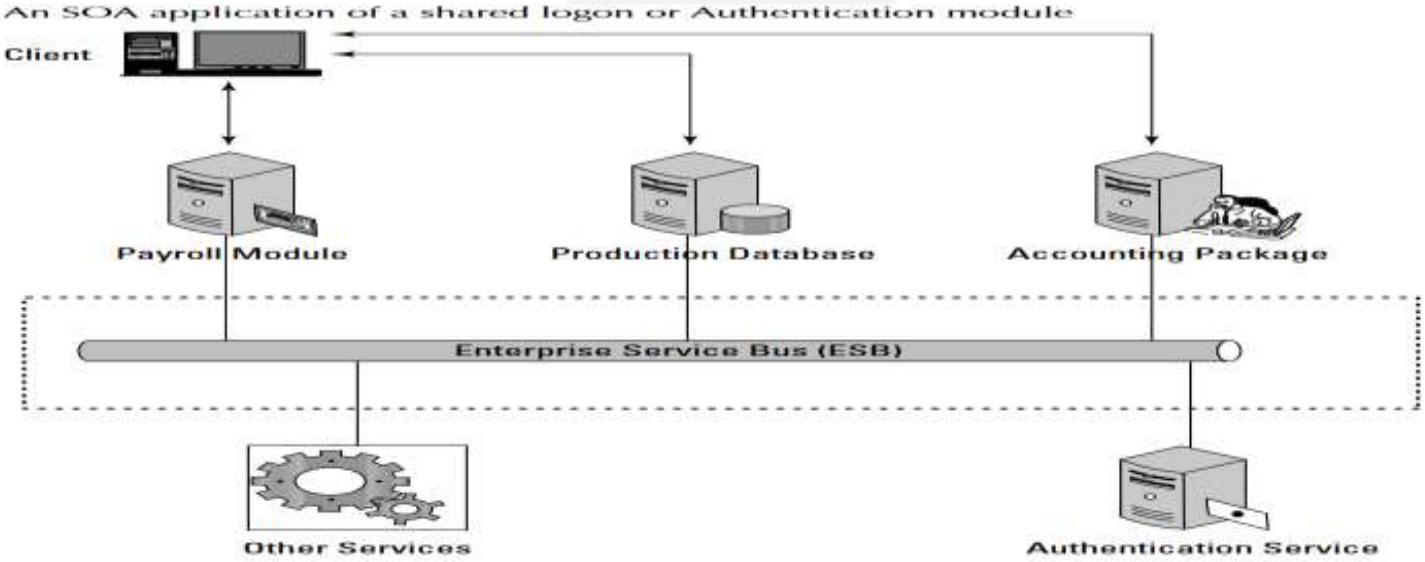
The Enterprise Service Bus

- An ESB is not a physical bus in the sense of a network; rather, it is an architectural pattern comprised of a set of network services that manage transactions in a Service Oriented Architecture.
- An ESB as a set of services that separate clients from components on transactional basis and that the use of the word bus in the name indicates a high degree of connectivity or fabric quality to the system; that is, the system is loosely coupled.
- Messages flow from client to component through the ESB, which manages these transactions, even though the location of the services comprising the ESB may vary widely.
- An ESB is necessary but not essential to a Service Oriented Architecture because typical business processes can span a vast number of messages and events, and distributed processing is an inherently unreliable method of transport.
- An ESB therefore plays the role of a transaction broker in SOA, ensuring that messages get to where they where supposed to go and are acted upon properly.

The Enterprise Service Bus

- The service bus performs the function of mediation: message translation, registration, routing, logging, auditing, and managing transactional integrity.

- Transactional integrity is similar to ACID in a database system—atomicity, consistency, isolation, and durability, the essence of which is that transactions succeed or they fail and are rolled back.



The Enterprise Service Bus

- An ESB may be part of a network operating system or may be implemented using a set of middleware products. An ESB creates a virtual environment layered on top of an enterprise messaging system where services are advertised and accessed.

- Think of an ESB as a message transaction system. IBM's WebSphere ESB 7.0 is an ESB based on open standards such as Java EE, EJB, WS-Addressing, WS-Policy, and Kerberos security, and it runs on the WebSphere Application Server.

- It is interoperable with Open SCA. WebSphere ESB contains both a Service Federation Management tool and an integrated Registry and Repository function.

The Enterprise Service Bus

Typical features are found in ESBs, among others:

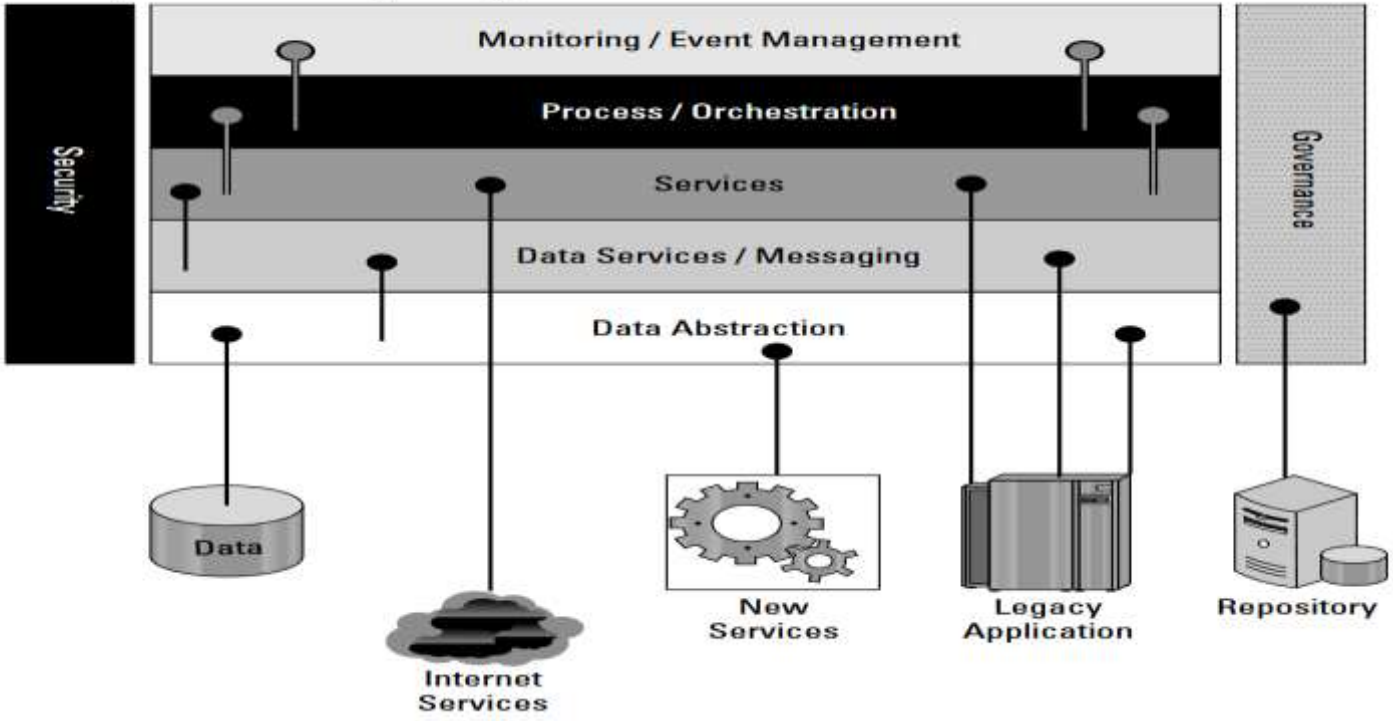
- Monitoring services aid in managing events.
- Process management services manage message transactions.
- Data repositories or registries store business logic and aid in governance of business processes.
- Data services pass messages between clients and services.
- Data abstraction services translate messages from one format to another, as required.
- Governance is a service that monitors compliance of your operations with governmental regulation, which can vary from state to state and from country to country.
- Security services validate clients and services and allow messages to pass from one to the other.

The Enterprise Service Bus

- The difference between a repository and a registry in the context of a Service Oriented Architecture is subtle. Repositories and registries are both data stores, but a repository stores references to the components of the SOA, their source code, and linking information that are used to provide SOA services.
- An SOA registry contains references to rules, descriptions, and definitions of the services that is, the metadata of the components.
- A repository serves the role that a name server does in a network operating system infrastructure, while the registry plays the role of a directory service (domain).
- The service broker uses the rules contained in the SOA registry to perform its function as translator and delivery agent. For developers, the registry serves as the central location to store component descriptions that allow composite applications to be created and the place in which services may be published for general use.
- These services in an SOA also include the provider interfaces and standard sets of network protocols that were mentioned previously. Developers may also choose to create a Business Process Orchestration module to coordinate the access and transactional integrity of multiple business applications that integrate into a larger platform,

SOA-Network Infrastructure

This figure shows a network services model infrastructure for an SOA, which is based on the SOA meta-model of the Linthicum Group, 2007. A slightly different version of this diagram appears in *Networking Bible* by Barrie Sosinsky, Wiley, 2009.



Service Catalogs

- Finding any particular service and locating the service's requirement in a large SOA implementa tion can involve a large amount of network system overhead.
- To aid in locating services, SOA infrastructure often includes a catalog service. This service stores information on the following, among other things:
- What services are available, both internal and external
- How to use a service
- Which applications are related to a particular service (dependencies)
- How services relate to one another
- Who owns the service and how a service is modified
- The event history of a service, including service levels, outages, and so on
- The nature of service contracts

Service Catalogs

- Service catalogs are dynamic and under constant modification. Catalog servers have these features:
- They can be standalone catalog servers serving a single site.
- They serve the role of a global catalog service where two or more catalog servers are merged to include several sites. A global service usually requires some sort of synchronization or update to maintain a unified data store across the servers involved.
- They can be part of a federated catalog service where two or more global catalog servers have access to one another's information through a trusted query relationship.
- Catalog services have an enormous impact on large system performance and eventually become essential as a SOA internetwork system grows.
- An internetwork is a network that is constructed through the consolidation of separate networks, in the same manner that the Internet has been built.

SOA Communications

- Message passing in SOA requires the use of two different protocol types: **the data interchange format and the network protocol that carries the message.**
- A client (or customer) connected to an **ESB communicates over a network protocol such as HTTP,** Representational State Transfer (REST), or Java Message Service (JMS) to a component (or service).
- Messages are most often in the form of the **eXtensible Markup Language (XML)** or in a variant such as the Simple Object Access Protocol (SOAP).
- SOAP is a messaging format used in **Web services that use XML as the message format** while relying on Application layer protocols such as HTTP and Remote Procedure Calls (RPC) for message negotiation and transmission.
- The software used to write clients and components can be written **in Java, .NET, Web Service Business Process Execution Language (WS-BPEL)**, or another form of executable code; the services that they message can be written in the same or another language.
- What is required is the ability to transport and translate a message into a form that both parties can understand.

SOA Communications

- An ESB may require a variety of combinations in order to support communications between a service consumer and a service provider.
- For example, in WebSphere ESB, you might see the following combinations:
- XML/JMS (Java Message Service)
- SOAP/JMS
- SOAP/HTTP
- Text/JMS
- Bytes/JMS
- The Web Service Description Language (WSDL) is one of the most commonly used XML protocol for messaging in Web services, and it finds use in Service Oriented Architectures.

SOA Communications

- Version 1.1 of WSDL is a W3C standard, but the current version WSDL 2.0 (formerly version 1.2) has yet to be ratified by the W3C.
- The significant difference between 1.1 and 2.0 is that version 2.0 has more support for RESTful (e.g. Web 2.0) application, but much less support in the current set of software development tools.
- The most common **transport for WSDL is SOAP, and the WSDL** file usually contains both XML data and an XML schema.
- REST offers some very different capabilities than SOAP. With REST, each URL is an object that you can query and manipulate.
- You use HTML commands such as GET, POST, PUT, and DELETE to work with REST objects. SOAP uses a different approach to working with Web data, exposing.
- Web objects through an API and transferring data using XML. The REST approach offers light weight access using standard HTTP command, is easier to implement than SOAP, and comes with less overhead.

SOA Communications

- **SOAP is often more precise and provides a more error-free consumption model**. SOAP often comes with more sophisticated development tools.
- All major Web services use REST, but many Web services, especially newer ones, combine REST with SOAP to derive the benefits that both offer.

WSDL are essential objects to support message transfer, including these:

- The service object, a container where the service resides.
- The port or endpoint, which is the unique address of the service.
- The binding, which is the description of the interface (e.g. RPC) and the transport (e.g. SOAP).
- The portType, or interface that defines the capabilities of the Web service, and what operations are to be performed, as well as the messages that must be sent to support the operation.
- The operation that is to be performed on the message.

SOA Communications

- The message content, which is the data and metadata that the service operation is performed on. Each message may consist of one or more parts, and each part must include typing information.

- The types used to describe the data, usually as part of the XML schema that accompanies the WSDL.

- A set of primers on WSDL may be found on the W3C Web site. The latest version, “Web Services Description
- Language (WSDL) Version 2.0 Part 0: Primer” may be found at http://www.w3.org/2002/ws/desc/wsdl20-primer. n

The code that follows is the WSDL 2.0 document that is created and analyzed in the W3C WSDL

- **primer referenced in the accompanying tip.**
- <?xml version=”1.0” encoding=”utf-8” ?>
- <description
- xmlns=”http://www.w3.org/ns/wsdl”
- targetNamespace= “http://greath.example.com/2004/wsdl/resSvc”
- xmlns:tns= “http://greath.example.com/2004/wsdl/resSvc”
- xmlns:ghns = “http://greath.example.com/2004/schemas/resSvc”
- xmlns:wsoap= “http://www.w3.org/ns/wsdl/soap”
- xmlns:soap=”http://www.w3.org/2003/05/soap-envelope”
- xmlns:wsdlx= “http://www.w3.org/ns/wsdl-extensions”>

SOA Communications

- <documentation>
- This document describes the GreatH Web service. Additional
- application-level requirements for use of this service --
- beyond what WSDL 2.0 is able to describe -- are available
- at http://greath.example.com/2004/reservation-documentation.html
- </documentation>
- <types> <xs:schema
- xmlns:xs=”http://www.w3.org/2001/XMLSchema”
- targetNamespace=”http://greath.example.com/2004/schemas/
- resSvc”
- xmlns=”http://greath.example.com/2004/schemas/resSvc”>
- <xs:element name=”checkAvailability”
- type=”tCheckAvailability”/>
- <xs:complexType name=”tCheckAvailability”>
- <xs:sequence>
- <xs:element name=”checkInDate” type=”xs:date”/>
- <xs:element name=”checkOutDate” type=”xs:date”/>
- <xs:element name=”roomType” type=”xs:string”/>
- </xs:sequence> </xs:complexType>

SOA Communications

- <xs:element name=”checkAvailabilityResponse” type=”xs:double”/>
- <xs:element name=”invalidDataError” type=”xs:string”/>
- </xs:schema>
- </types>
- <interface name = “reservationInterface” >
- <fault name = “invalidDataFault”
- element = “ghns:invalidDataError”/>
- <operation name=”opCheckAvailability”
- pattern=”http://www.w3.org/ns/wsdl/in-out”
- style=”http://www.w3.org/ns/wsdl/style/iri”
- wsdlx:safe = “true”>
- <input messageLabel=”In”
- element=”ghns:checkAvailability” />
- <output messageLabel=”Out”
- element=”ghns:checkAvailabilityResponse” />
- <outfault ref=”tns:invalidDataFault” messageLabel=”Out”/>
- </operation>

SOA Communications

- </interface>
- <binding name=”reservationSOAPBinding”
- interface=”tns:reservationInterface”
- type=”http://www.w3.org/ns/wsdl/soap”
- wsoap:protocol=”http://www.w3.org/2003/05/soap/bindings/HTTP/”>
- <fault ref=”tns:invalidDataFault”
- wsoap:code=”soap:Sender”/>
- <operation ref=”tns:opCheckAvailability”
- wsoap:mep=”http://www.w3.org/2003/05/soap/mep/soap-response”/>
- </binding>
- <service name=”reservationService”
- interface=”tns:reservationInterface”>
- <endpoint name=”reservationEndpoint”
- binding=”tns:reservationSOAPBinding”
- address =”http://greath.example.com/2004/
- reservation”/>
- </service>
- </description>

SOA Communications

- The XML document sets up the **namespace, defines the interface, specifies the binding, names the service, provides the documentations for the service,** and then supplies a schema that may be used to validate the document. In the message descriptions, the message types are declared.

- **XML schema can be separate files**, but what we see here is normal for WSDL, an inline schema that is part of the WSDL document. For a much more complete step-by-step description, refer to the W3C tutorial.

- A WSDL file contains essential message data for a transaction, but it doesn't capture the full scope of a Service Oriented Architecture design.

- Additional requirements need to be specified. The functional requirements for message passing between client and service in SOA are embodied in the concept of a service contract.

- A service contract codifies the relationship between the data to be processed, the metadata that accompanies that data, the intended service, and the manner in which the service will act upon that message.

SOA Communications

- **Header:** The header contains the name of the service, service version, owner of the service, and perhaps a responsibility assignment. This is often defined in terms of a RACI matrix where the various roles and responsibilities for processes are spelled out in terms of a set of tasks or deliverables.

- The acronym designates the Responsible party or service, the Accountable decision maker, the Consulted party, and person(s) or service(s) that must be Informed on the use of the service.

- **Service Type**: Examples of service types include data, business, integration, presentation, and process types.

- **Functional Specification:** This category includes the functional requirements, what service operations or actions and methods must be performed, and the manner in which a service is invoked or initiated. Invocation usually includes the URL and the nature of the service interface.

- **Transaction attributes**: A message may define a transaction that may need to be managed or tracked or be part of or include another transaction operated at a specific Quality of Service and under a specific Service Level Agreement (SLA).
- Security parameters also are part of a transaction's attributes, as are the role the message plays in a process and the terms or semantics used to describe the interaction of the message with a service's interface.

Business Process Execution Language

- If a message represents an **atomic transaction in a Service Oriented Architecture,** the next level of abstraction up is the grouping and managing of sets of transactions to form useful work and to execute a business process.
- An example of an execution language is the Business Process Execution Language (BPEL) or alternatively as the **Web Service Business Process Execution Language.**
- (WS-BPEL), a language standard for Web service interactions. The standard is maintained by the Organization for the Advancement of Structured Information Standards (OASIS) through their Web Services Business Process Execution Language Technical Committee.
- **BPEL is a meta-language comprised of two functions:** executable commands for Web services and clients, and internal or abstract code for executing the internal business logic that processes require.
- A meta-language is any language whose statements refer to statements in another language referred to as the object language.
- BPEL is often used to compose, orchestrate, and coordinate business processes with Web services in the SOA model, and it has commands to manage asynchronous communications.

Business Process Execution Language

- BPEL uses XML with specific support for messaging protocols such as SOAP, WSDL, UDDI, WS-Reliable Messaging, WS-Addressing, WS-Coordination, and WS-Transactions.
- BPEL also builds on IBM's Web Services Flow Language (WSFL) and Microsoft's XLANG for data transport.
- The former is a system of directed graphs, while the latter is a block-structured language adding to additional verbs and nouns specific for business processes to BPEL, which were combined to form BPEL4WS and are being merged with BPEL.
- A version of BPEL to support human interaction is called BPEL4People, and it falls under the WS-HumanTask specifications of OASIS.
- BPEL was designed to interact with WSDL and define business processes using an XML language. BPEL does not have a graphical component.
- A business process has an internal or executable view and an external or abstract view in BPEL.

Business Process Execution Language

- One process may interact with other processes, but the goal is to minimize the number of specific extensions added to BPEL to support any particular business process.
- Data functions in BPEL support process data and control flow, manage process instances, provide for logic and branching structures, and allow for process orchestration.
- Because transactions are long-lived and asynchronous, BPEL includes techniques for error handling and scopes transactions. As much as possible, BPEL uses Web services for standards and to assemble and decompose processes.

SOA Communications

- SOA was created by the industry to solve a problem: **how to make disparate, diverse,** and distributed services talk to disparate and diverse clients.
- The final result of an SOA project isn't the access of services, per se; it is the creation of a business process.
- In a complex business project, **the developers juggle many clients and many services**, which can make visualization of the overall system difficult.
- To address this problem, various modeling tools have been developed to support SOA development and **optimization, system and process management, change and life-cycle management.**
- Several methodologies have been developed to model SOAs.
- Working in a software package to model your business processes is similar in approach to designing and **optimizing a relational database in entity-relationship, object-role modeling package, or another Computer Aided System Engineering (CASE) tool for data storage** and equally as valuable.

Business Process Modeling

- **Unified Modeling Language (UML):** The UML standard is the work of the Object Management Group (http://www.omg.org/).
- **UML creates graphical representations of software systems** in the form of a set of diagram types. Elements in a **UML architectural blueprint include actors, business processes, logic modules (components), program routines, database schemas, software components, and activities.**
- UML diagrams are separated into **seven structural types and four behavior types**; structure types model the components of the system, while behavior types model states, actions, and events.
- UML is widely used in the **industry for software system modeling.** A developed system model can be reduced automatically to code.
- **XML Metadata Interchange (XMI)**: XMI is another standard of the Object Management Group (OMG) and is used to exchange metadata using the Extensible Markup Language (XML).
- Metadata is structured into a **meta model** that fits into the OMG's Meta-Object Facility. UML models often use XMI as their interchange format, although they can be used by other languages.

Business Process Modeling

- XMI files are not generally interchangeable between the different modeling languages that can use them. XMI has been codified as an international standard by **ISO, as IS0/IEC 19503:2005.**

- **Systems Modeling Language (SysML) is an open-source extension** of the part of the UML system dealing with profiles. It is smaller, more focused, and easier to learn and work with than UML itself. SysML reuses 7 of UML 2.0's 13 diagrams.

- The effort to develop SysML was rolled into OMG in 2008, but remains open source. **SysML can use XMI and is developing toward support of ISO 10303,** which is the Standard for the Exchange of Product Model Data (STEP) AP-233.

- **Business Process Modeling Notation (BPMN)** is a methodology for representing business processes as a set of connected visual objects that illustrate workflow in a Business Process Diagram (BPD).

- Originally developed in the **Business Process Management Initiative (BPMI),** it was incorporated into the Open Management Group in 2005.

- A BPD can be reduced to the OASIS standard **WS-Business Process Execution Language**, which is an executable language for information transfer between different Web services.

Business Process Modeling

- **Service-Oriented Modeling Framework (SOMF):** This framework was proposed by Michael Bell and combines a modeling language with a graphical display of the various SOA components so the system can be viewed as a map of objects and associated relationships.

- SOMF software allows developers to **create an action plan to implement their business processes** and can be valuable in system and architecture optimization, tracing message pathways, positioning software assets correctly, and providing a language for describing through abstraction and generalization how the processes operate.

- **SOMF soft-ware** not only allows you to determine what needs to be done, but also allows you to run "what-if" scenarios to see how changes will impact your SOA system.

- **SOMA reduces services** to a set of service objects and breaks down relationships into three components: the services themselves, the service components that make use of those services, and the information flows required to interact between them.

- Flows consist both of processes as well as their internal composition. In SOMA, domains and functional groups are identified, variables that **affect processes are analyzed, and component development** and object oriented analysis are used to model specific cases. SOMA is meant to provide information on service and service boundaries, service granularity, and asset analysis.

Business Process Modeling

In SOMF, the modeling is based on the elements of the service life cycle:

- **Conceptualization:** Defining the processes that will be supported
- **Discovery:** Determining how components are exposed
- **Analysis:** Identifying the relationships services have to clients
- **Integration:** Implementing the messaging infrastructure that connects clients to components
- **Logical design**: Building the system and process logic and determining how orchestration or choreography will be performed
- **Architecture design:** Creating the system component architecture and reducing design to software components with specified interfaces
- **System implementation:** Building and testing the system components

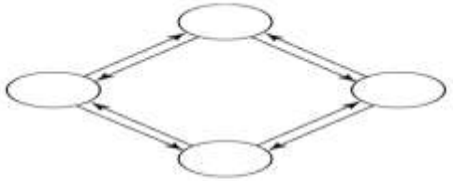
Four modelling topologies for message passing are used in SOMF:

- **Circular topology:** A circular topology is one where message passing is carried out in a circular fashion.
- There is no orchestrator in this system, and each component providing a service is responsible for knowing which message to act on and where to send a message next. That is, the choreography of the system is maintained at the component.
- **Hierarchical topology:** In a hierarchical topology, services are arranged in a tree pattern with parent/child relationships.
- Messages from one service to another must traverse up the branch of the tree and down another branch from the root until the matching service is found. Hierarchical topologies offer the advantage of a well-defined set of relationships, a central location (the root) where logic may reside, and a clearly stated address space.
- The disadvantage of a **hierarchical topology is that the overhead of passing a message** from one service to another **isn't optimized.**

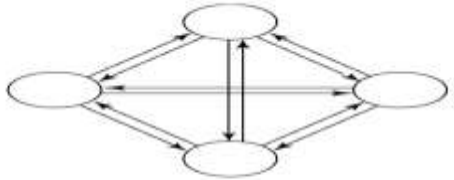
Business Process Modeling

- **Network topology:** A network topology has a many-to-many relationship between services and their clients.
- The advantage of a network topology is that the overhead associated with **message passing has been minimized** but there is considerable overhead built into the system in order to maintain the many links needed.
- **Star topology:** In a star topology, services are designed to connect through a central service. The star model is favored in orchestration processes and is useful for services that use broadcasting or multicasting services, publish and subscribe, and other related systems.

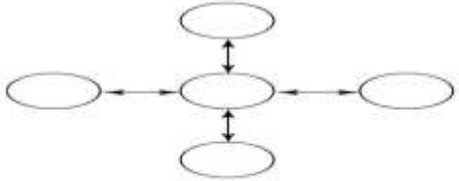
The four different message-passing topologies used in SOME are shown above. The lines and arrows indicate message-passing pathways and relationships.



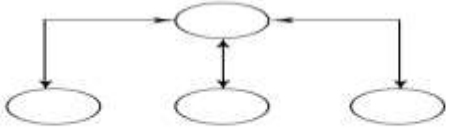
Circular topology



Network topology

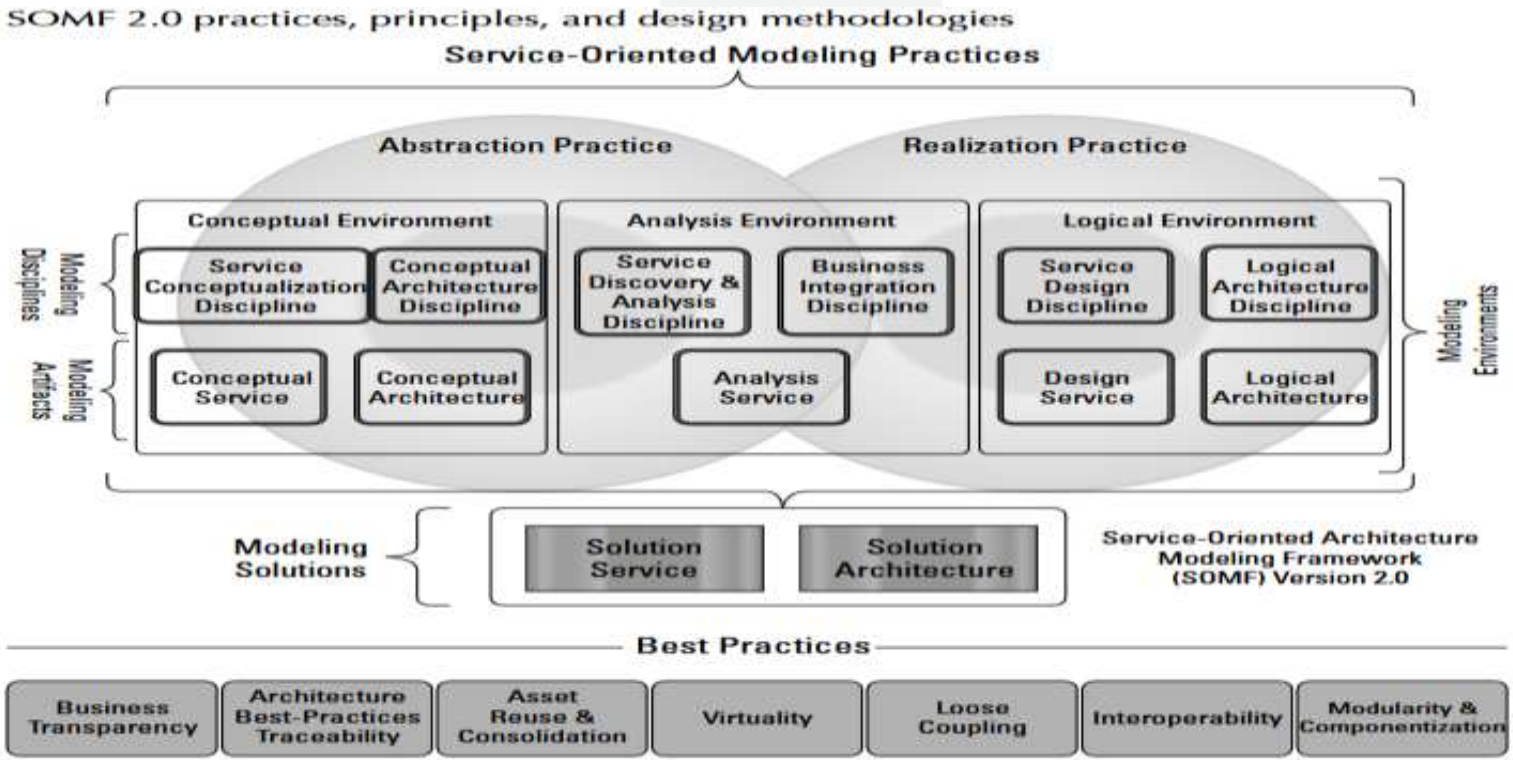


Star topology



Hierarchical topology

Business Process Modeling



- A service in SOMF that is granular and very narrow in scope is referred to as an atomic service. An atomic service cannot be decomposed into smaller services that provide a useful function.
- An example of an atomic service is a customer lookup that contains a customer ID and the customer's name and address. An address that doesn't contain the customer's name or lists a customer name without an ID wouldn't have much value, so this service stands alone.
- A service that provides ID, name, and address might be part of a set of services that describe a bank account, customer purchase record, or any number of services. A collection of services that work together is referred to as a composite service.
- The Enterprise Service Bus described previously in this chapter is another example of a composite service; it contains functions for message routing, message interchange and translation, and process orchestration.
- A composite service is usually organized as a hierarchical topology and is multifunctional or a coarse-grained entity. SOA describes a distributed collection of services performing business process functions.

Business Process Modeling

- A collection of composite services that would form a process module is referred to as a service cluster. Service clusters may be composed of both atomic services and composite services.
- Large functions such as payroll modules would normally be composite SOA services.
- In an SOMF model, each of these three service types (atomic, composite, and clusters) appears as a specific shape, and connections are made between them that generalize, specify, expand, or contract the services they provide.
- Services are typed, granular services are identified, and then services may be aggregated, decomposed, unified, intersected, or subtracted from other services to suit the needs of the business process being modeled.
- The SOMF modeling notation has a symbol for each analysis that relates one service to another. As you build a business process, you add services to the model and connect them in ways that make sense for your workflow.
- When the model is complete and optimized, it is reduced to a conceptualized service that relates the business process to the specific implementation chosen. Some modeling technologies allow for the reduction of the model to executable code.

Managing and Monitoring SOA

- Software for monitoring and managing an SOA infrastructure plays an important role in large SOA deployments.
- While SOA offers a logical design and reusable components, it does not make the task of network management any easier. If anything, SOA management requires proactive over sight because you can't wait for a particular application to fail before taking corrective action.
- Tools for managing SOAs tend to be multifaceted and run constantly. There are a number of network management frameworks products and suites, notably these:
- HP Software and Solutions OpenView SOA Manager (https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto &cp=1-10^36657_4000_100)
- IBM Tivoli Framework Composite Application Manager for SOA (ITCAM; see http:// www-01.ibm.com/software/tivoli/solutions/),Oracle BPEL Process Manager (http://www.oracle.com/technology/bpel/index.html)
- These products have SOA tools for network management. IBM's product specializes in change management and SOA lifecycle development, and it integrates with a WebSphere and other Tivoli systems.

Managing and Monitoring SOA

- HP SOA Manager provides dynamic mapping, monitoring, and optimization of SOA services such as Web services, software assets, and virtual services.
- These framework products create a central console with a variety of management views. Oracle's BPEL Process Manager and WebSphere are process managers for creating an Enterprise Service Bus.
- The SOA management software technology is dynamic, with many small vendors' products some of which have been purchased and rolled into larger systems.
- Oracle's recent acquisition of AmberPoint's SOA Management System is an example of this trend.
- BMC Software's AppSight **(http://www.bmc.com/products/product-listing/BMC-AppSight.html)** is an automated SOA problem-resolution package, as is Tidal Software's Intersperse package, which has root cause analysis services.

Managing and Monitoring SOA

- The CA Wily SOA Solution (http://www.ca.com/us/eitm/solution.aspx?id=8254) is a monitoring and discovery service that can map SOA transactions and dependencies and discover components such as ESBs, Web portals, and various Web services. iTKO's LISA **(http://www.itko.com/products/index. jsp).**

- Enterprise SOA Testing platform specializes in testing Web service components that are used in SOA. Another example of an SOA transaction manager is OpTier's CoreFirst (http://www. optier.com/corefirst_overview.aspx.

- Configuration and change management present a particular challenge in the area of In addition to the fact that elements of an SOA infrastructure can be highly distributed and therefore require good discovery mechanisms, these environments also are highly virtualized.

- As workloads vary, solutions often provision virtual servers as needed and move these virtual servers' processing across physical servers. Virtualization will continue to challenge SOA management software well into the future.

Relating SOA and Cloud Computing

- Cloud computing is still in its infancy, and although Web services can implement a Service Oriented Architecture, it is not a requirement.
- Applications of those types have less of a need for the flexibility and loose coupling that SOA provides.
- As cloud applications become more diverse in scope, SOA offers an architectural blueprint for accessing diverse optimized services through a loosely coupled standardized method that provides an ability to evolve that is difficult to implement in any other way.
- SOA is loosely coupled because the service is separated from the messaging. If a component doesn't provide the capabilities required, it is an easy task to switch to a different component, and switching requires almost no programming.
- Developers lose some of the ability to customize modules, but gain a significant advantage in simplifying their applications. Taken as a whole, applications that rely on SOA components can be very complex and appear to be tightly coupled, when in reality they are not.

SOA Security

- Any system that sends hundreds or thousands of messages across an internetwork as SOA does is subject to attack in all the traditional ways that network traffic is hijacked, spoofed, redirected, or blocked.
- Because SOA eliminates the use of application boundaries, the traditional methods where security is at the application level aren't likely to be effective.
- Cisco has a family of products that enforce rules and policies for the transmission of XML messaging that they have named Application Oriented Networking (AON; http://www.cisco.com/en/US/products/ps6480/).
- A similar policy based XML security service may be found in Citrix's NetScaler 9.0 (http://www.citrix.com/English/ps2/products/product.asp?contentID=21679) Web application delivery appliance.
- To address SOA security, a set of OASIS standards (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security) was created, which includes the following:

SOA Security

- **Security Assertion Markup Language (SAML)** is an XML standard that provides for data authentication and authorization between client and service.
- The SAML technology is used as part of Single Sign-on Systems (SSO) and allows a user logging into a system from a Web browser to have access to distributed SOA resources.
- **WS-Security (WSS)** is an extension of SOA that enforces security by applying tokens such as Kerberos, SAML, or X.509 to messages.Through the use of XML Signature and XML Encryption, WSS aims to offer client/service security.
- **WS-SecureConversion** is a Web services protocol for creating and sharing security context. WS-SecureConversion is meant to operate in systems where WS-Security, WS-Trust, and WS-Policy are in use, and it attaches a security context token to communications such as SOAP used to transport messages in an SOA enterprise.
- **WS-SecurityPolicy** provides a set of network policies that extend WS-Security, WS-Trust, and WS-SecureConversion so messages complying to a policy must be signed and encrypted. The SecurityPolicy is part of a general WS-Policy framework.

- **WS-Trust extends WS-Security** to provide a mechanism to issue, renew, and validate security tokens.
- A Web service using WS-Trust can implement this system through the use of a Security Token Service (STS), a mechanism for attaching security tokens to messages and a set of mechanisms for key exchanges that are used to validate tokens and messages.
- Another approach to enforcing security in SOA is to use an XML gateway that intercepts XML messages transported by SOAP or REST, identifies the source of the message, and verifies that the message was securely received.
- Providing XML Gateway SOA security requires a Public Key Infrastructure (PKI) so that encryption is enforced by digital signatures.
- Progress Software's Actional 8.0 (http://web.progress.com/en/actional/index.html) now includes Mindreef's SOAPscope Server and has an XML middleware service that performs diagnostic testing and Web services governance, adding that component to Actional's ability to monitor and map XML appliances and application servers.

The Open Cloud Consortium

- The Open Cloud Consortium (OCC; see http://opencloudconsortium.org/) is an organization comprised of several universities and interested companies that supports the development of standards for cloud computing and for interoperating with the various frameworks.

OCC working groups perform these functions:

- They develop benchmarks for measuring cloud computing performance. Their benchmark and data generator for measuring large data clouds is called MalStone (http://code.google.com/p/malgen/).
- They provide testbeds that vendors can use to test their applications, including the Open Cloud Testbed and the Intercloud Testbed that are part of the work of the Open Cloud Testbed and Intercloud working groups.
- They support the development of open-source reference implementations for cloud computing.
- The Working Group on Standards and Interoperability For Large Data Clouds extends the architecture for data storage with a distributed file system, table services, and computing using MapReduce following the model that is part of Google's offering.

The Open Cloud Consortium

- MapReduce is Google's patented software framework that supports distributed large data sets organized by the Google File System (GFS) accessed by clusters of computers.
- The Apache Hadoop (http://hadoop.apache.org/) open-source system is based on MapReduce and GFS.
- They support the management of cloud computing infrastructure for scientific research as part of the Open Science Data Cloud (OSDCP) Working Group's initiative.

Relating SOA and Cloud Computing

- SOA components are often best-of-breed service providers that can provide a measured service level and can play a role in Business Process Management (BPM) systems.
- The separation of services from their design allows for much easier system upgrades and maintenance. Many Web 2.0 applications use SOA components, and SOA will become increasingly useful in larger applications that require many Web services.
- These applications often rely on REST and feature AJAX components in a user interface that supports Web syndication blogs, and wikis. Some people regard mashups as Web 2.0 applications as well.
- A mashup is the combination of data from two or more sources that creates a unique service. The layers added to Google maps are examples of mashups.
- AJAX stands for Asynchronous JavaScipt and XM. AJAX is a set of development tools that allow for client input into Web applications; it is not a standard.
- AJAX describes a group of technologies that leverage HTML and CSS for styling, Web objects in the Document Object Model (DOM) or data, XML and XSLT for data interchange, the XMLHttpRequest for asynchronous communication, and JavaScript commands to request data from data sources.

Relating SOA and Cloud Computing

- The challenge SOA faces in designing systems to support Web 2.0 is the lack of standardization in how components in Web 2.0 are used.
- People believe that SOA will play a role in creating what has been dubbed an “Internet of Services” where complex services will be available for use as a set of building blocks based on the convergence of SOA and Web 2.0.
- The Gartner Group refers to this trend as the development of “Advanced SOA,” but features of SOA that are event-driven have been part of many vendors’ middleware offerings for several years now.