Release Management

Isaac Griffith

CS 3321
Department of Computer Science
Idaho State University

Topics Covered

- Everything as a service

Everything as a service

- The idea of a service that is rented rather than owned is fundamental to cloud computing.
- Infrastructure as a service (IaaS)
  - Cloud providers offer different kinds of infrastructure service such as a compute service, a network service and a storage service that you can use to implement virtual servers.
- Platform as a service (PaaS)
  - This is an intermediate level where you use libraries and frameworks provided by the cloud provider to implement your software. These provide access to a range of functions, including SQL and NoSQL databases.
- Software as a service (SaaS)
  - Your software product runs on the cloud and is accessed by users through a web browser or mobile app.

Idaho State University
Computer Science

Everything as a service

ROAR

Management of IaaS and Paas

Software as a service

- Increasingly, software products are being delivered as a service, rather than installed on the buyer's computers.
- If you deliver your software product as a service, you run the software on your servers, which you may rent from a cloud provider.
- Customers don't have to install software and they access the remote system through a web browser or dedicated mobile app.
- The payment model for software as a service is usually a subscription model.
  - Users pay a monthly fee to use the software rather than buy it outright.

Software as a service

Benefits of SaaS

- Cash flow
  - Customers either pay a regular subscription or pay as they use the software. This means you have a regular cash flow, with payments throughout the year. You don't have a situation where you have a large cash injection when products are purchased but very little income between product releases.
- Update management
  - You are in control of updates to your product and all customers receive the update at the same time. You avoid the issue of several versions being simultaneously used and maintained. This reduces your costs and makes it easier to maintain a consistent software code base.
- Continuous deployment
  - You can deploy new versions of your software as soon as changes have been made and tested. This means you can fix bugs quickly so that your software reliability can continuously improve.

Idaho State University
Computer Science

Benefits of SaaS for software product providers

- Payment flexibility
  - You can have several different payment options so that you can attract a wider range of customers. Small companies or individuals need not be discouraged by having to pay large upfront software costs.
- Try before you buy
  - You can make early free or low-cost versions of the software available quickly with the aim of getting customer feedback on bugs and how the product could be approved.
- Data collection
  - You can easily collect data on how the product is used and so identify areas for improvement. You may also be able to collect customer data that allows you to market other products to these customers.

ROAR

Advantages/disadvantages of SaaS for customers

Idaho State University
Computer Science

Data storage and management issues for SaaS

- Regulation
  - Some countries, such as EU countries, have strict laws on the storage of personal information. These may be incompatible with the laws and regulations of the country where the SaaS provider is based. If a SaaS provider cannot guarantee that their storage locations conform to the laws of the customer's country, businesses may be reluctant to use their product.
- Data transfer
  - If software use involves a lot of data transfer, the software response time may be limited by the network speed. This is a problem for individuals and smaller companies who can't afford to pay for very high speed network connections.
- Data security
  - Companies dealing with sensitive information may be unwilling to hand over the control of their data to an external software provider. As we have seen from a number of high profile cases, even large cloud providers have had security breaches. You can't assume that they always provide better security than the customer's own servers

ROAR

Design issues for SaaS

SaaS design issues

- Local/remote processing
  - A software product may be designed so that some features are executed locally in the user's browser or mobile app and some on a remote server.
  - Local execution reduces network traffic and so increases user response speed. This is useful when users have a slow network connection.
  - Local processing increases the electrical power needed to run the system.
- Authentication
  - If you set up your own authentication system, users have to remember another set of authentication credentials.
  - Many systems allow authentication using the user's Google, Facebook or LinkedIn credentials.
  - For business products, you may need to set up a federated authentication system, which delegates authentication to the business where the user works.

SaaS design issues

- Information leakage
  - If you have multiple users from multiple organizations, a security risk is that information leaks from one organization to another.
  - There are a number of different ways that this can happen, so you need to be very careful in designing your security system to avoid this.
- Multi-tenant and multi-instance systems
  - In a multi-tenant system, all customers are served by a single instance of the system and a multitenant database.
- In a multi-instance system, a separate copy of the system and database is made available for each user.

Multi-tenant systems

- A multi-tenant database is partitioned so that customer companies have their own space and can store and access their own data.
  - There is a single database schema, defined by the SaaS provider, that is shared by all of the system's users.
  - Items in the database are tagged with a tenant identifier, representing a company that has stored data in the system. The database access software uses this tenant identifier to provide 'logical isolation', which means that users seem to be working with their own database.

Example multi-tenant database

Advantages of multi-tenant databases

- Resource utilization
  - The SaaS provider has control of all the resources used by the software and can optimize the software to make effective use of these resources.
- Security
  - Multitenant databases have to be designed for security because the data for all customers is held in the same database. They are, therefore, likely to have fewer security vulnerabilities than standard database products. Security management is simplified as there is only a single copy of the database software to be patched if a security vulnerability is discovered.
- Update management
  - It is easier to update a single instance of software rather than multiple instances. Updates are delivered to all customers at the same time so all use the latest version of the software.

Idaho State University Computer Science

Disadvantages of multi-tenant databases

- Inflexibility
  - Customers must all use the same database schema with limited scope for adapting this schema to individual needs. I explain possible database adaptations later in this section.
- Security
  - As data for all customers is maintained in the same database, then there is a theoretical possibility that data will leak from one customer to another. In fact, there are very few instances of this happening. More seriously, perhaps, if there is a database security breach then it affects all customers.
- Complexity
  - Multitenant systems are usually more complex than multi-instance systems because of the need to manage many users. There is, therefore, an increased likelihood of bugs in the database software.

ROAR

Possible customizations for SaaS

- Authentication
  - Businesses may want users to authenticate using their business credentials rather than the account credentials set up by the software provider. I explain, in Chapter 7, how federated authentication makes this possible.
- Branding
  - Businesses may want a user interface that is branded to reflect their own organisation.
- Business rules
  - Businesses may want to be able to define their own business rules and workflows that apply to their own data.
- Data schemas
  - Businesses may want to be able to extend the standard data model used in the system database to meet their own business needs.
- Access control
  - Businesses may want to be able to define their own access control model that sets out the data that specific users or user groups can access and the allowed

User profiles for SaaS access

Database extensibility using additional fields

Adding fields to extend the database

- You add some extra columns to each database table and define a customer profile that maps the column names that the customer wants to these extra columns. However:
  - It is difficult to know how many extra columns you should include. If you have too few, customers will find that there aren't enough for what they need to do.
  - If you cater for customers who need a lot of extra columns, however, you will find that most customers don't use them, so you will have a lot of wasted space in your database.
  - Different customers are likely to need different types of columns.
    - For example, some customers may wish to have columns whose items are string types, others may wish to have columns that are integers.
    - You can get around this by maintaining everything as strings. However, this means that either you or your customer have to provide conversion software to create items of the correct type.

Extending a database using tables

- An alternative approach to database extensibility is to allow customers to add any number of additional fields and to define the names, types and values of these fields.
- The names and types of these values are held in a separate table, accessed using the tenant identifier.
- Unfortunately, using tables in this way adds complexity to the database management software.
  - Extra tables must be managed and information from them integrated into the database.

Database extensibility using tables

Database security

- Information from all customers is stored in the same database in a multii-multi-tenant system so a software bug or an attack could lead to the data of some or all customers being exposed to others.
- Key security issues are multilevel access control and encryption.
  - Multilevel access control means that access to data must be controlled at both the organizational level and the individual level.
  - You need to have organizational level access control to ensure that any database operations only act on that organization's data. The individual user accessing the data should also have their own access permissions.
- Encryption of data in a multitenant database reassures corporate users that their data cannot be viewed by people from other companies if some kind of system failure occurs.

Multi-instance databases

- Multi-instance systems are SaaS systems where each customer has its own system that is adapted to its needs, including its own database and security controls.
- Multi-instance, cloud-based systems are conceptually simpler than multi-tenant systems and avoid security concerns such as data leakage from one organization to another.
- There are two types of multi-instance system:
  - VM-based multi-instance systems are multi-instance systems where the software instance and database for each customer runs in its own virtual machine. All users from the same customer may access the shared system database.
  - Container-based multi-instance systems* These are multi-instance systems where each user has an isolated version of the software and database running in a set of containers.
  - This approach is suited to products in which users mostly work independently, with relatively little data sharing. Therefore, it is best used for software that

Advantages of multi-instance databases

- Flexibility
  - Each instance of the software can be tailored and adapted to a customer’s needs. Customers may use completely different database schemas and it is straightforward to transfer data from a customer database to the product database.
- Security
  - Each customer has their own database so there is no possibility of data leakage from one customer to another.
- Scaleability
  - Instances of the system can be scaled according to the needs of individual customers. For example, some customers may require more powerful servers than others.
- Resilience
  - If a software failure occurs, this will probably only affect a single customers. Other customers can continue working as normal.

Disadvantages of multi-instance databases

- Cost
  - It is more expensive to use multi-instance systems because of the costs of renting many VMs in the cloud and the costs of managing multiple systems. Because of the slow startup time, VMs may have to be rented and kept running continuously, even if there is very little demand for the service.
- Update management
  - It is more complex to manage updates to the software because many instances have to be updated. This is particularly problematic where individual instances have been tailored to the needs of specific customers.

Idaho State University
Computer Science

Key points

- A fundamental feature of the cloud is that ‘everything’ can be delivered as a service and accessed over the internet. A service is rented rather than owned and is shared with other users.
- Infrastructure as a service (IaaS) means computing, storage and other services are available over the cloud. There is no need to run your own physical servers.
- Platform as a service (PaaS) means using services provided by a cloud platform vendor to make it possible to auto-scale your software in response to demand.
- Software as a service (SaaS) means that application software is delivered as a service to users. This has important benefits for users, such as lower capital costs, and software vendors, such as simpler deployment of new software releases.
- Multitenant systems are SaaS systems where all users share the same

ROAR

Are there any questions?