Software-as-a-Service

Idaho State University | Computer Science

Isaac Griffith

CS 3321
Department of Computer Science
Idaho State University

ROAR

Topics Covered

- Software-as-a-Service

Use of p2p architecture

- When a system is computationally-intensive and it is possible to separate the processing required into a large number of independent computations.
- When a system primarily involves the exchange of information between individual computers on a network and there is no need for this information to be centrally-stored or managed.

Security issues in p2p system

- Security concerns are the principal reason why p2p architectures are not widely used.
- The lack of central management means that malicious nodes can be set up to deliver spam and malware to other nodes in the network.
- P2P communications require careful setup to protect local information and if not done correctly, then this is exposed to the peers.

Software as a service

- Software as a service (SaaS) involves hosting the software remotely and providing access to it over the Internet.
  - Software is deployed on a server (or more commonly a number of servers) and is accessed through a web browser. It is not deployed on a local PC.
  - The software is owned and managed by a software provider, rather than the organizations using the software.
  - Users may pay for the software according to the amount of use they make of it or through an annual or monthly subscription.

Key elements of SaaS

- Software is deployed on a server (or more commonly a number of servers) and is accessed through a web browser. It is not deployed on a local PC.
- The software is owned and managed by a software provider, rather than the organizations using the software.
- Users may pay for the software according to the amount of use they make of it or through an annual or monthly subscription. Sometimes, the software is free for anyone to use but users must then agree to accept advertisements, which fund the software service.

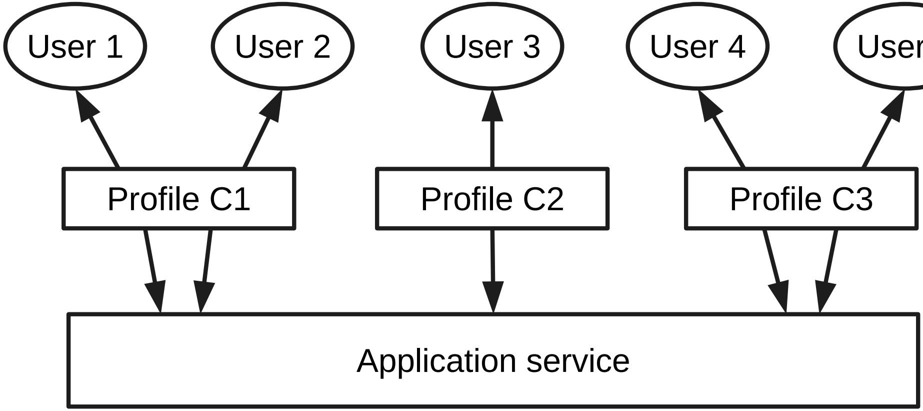
SaaS and SOA

- Software as a service is a way of providing functionality on a remote server with client access through a web browser. The server maintains the user's data and state during an interaction session. Transactions are usually long transactions e.g. editing a document.
- Service-oriented architecture is an approach to structuring a software system as a set of separate, stateless services. These may be provided by multiple providers and may be distributed. Typically, transactions are short transactions where a service is called, does something then returns a result.

Implementation factors for SaaS

- **Configurability** How do you configure the software for the specific requirements of each organization?
- **Multi-tenancy** How do you present each user of the software with the impression that they are working with their own copy of the system while, at the same time, making efficient use of system resources?
- **Scalability** How do you design the system so that it can be scaled to accommodate an unpredictably large number of users?

Configuration of a software system offered as a service



Service configuration

- **Branding**, where users from each organization, are presented with an interface that reflects their own organization.
- **Business rules and workflows**, where each organization defines its own rules that govern the use of the service and its data.
- **Database extensions**, where each organization defines how the generic service data model is extended to meet its specific needs.
- **Access control**, where service customers create individual accounts for their staff and define the resources and functions that are accessible to each of their users.

Multi-tenancy

- Multi-tenancy is a situation in which many different users access the same system and the system architecture is defined to allow the efficient sharing of system resources.
- It must appear to each user that they have the sole use of the system.
- Multi-tenancy involves designing the system so that there is an absolute separation between the system functionality and the system data.

A multi-tenant database

Tenant	Key	Name	Address
234	C100	XYZ Corp	43, Anystreet, Sometown
234	C110	BigCorp	2, Main St, Motown
435	X234	J. Bowie	56, Mill St, Starville
592	PP37	R. Burns	Alloway, Ayrshire

Scalability

- Develop applications where each component is implemented as a simple stateless service that may be run on any server.
- Design the system using asynchronous interaction so that the application does not have to wait for the result of an interaction (such as a read request).
- Manage resources, such as network and database connections, as a pool so that no single server is likely to run out of resources.
- Design your database to allow fine-grain locking. That is, do not lock out whole records in the database when only part of a record is in use.

Key points

- Peer-to-peer architectures are decentralized with no distinguished clients and servers. Computations can be distributed over many systems in different organizations.
- Software as a service is a way of deploying applications as thin client-server systems, where the client is a web browser.

Are there any questions?