Cloud Computing, and REST-based Architectures

Group 10

Mexiang Jin (m38jin)
Ziniu Wu (z82wu)

Cloud Computing pattern:

- motivation
  - If a company has a huge customer base, then the server is really likely to slow down, and company's product cannot keep up with the demand. Therefore, lots of big company buy or rent server to prevent this situation, buts this costs a lot of money and lots of time, since they have to pay for it to keep the server running all the time. That too expensive for some small companies, so they choose cloud computing.
- Intended use case
  - end user to cloud: Applications running on the cloud and accessed by end users
  - Enterprise to Cloud to End User: Applications running in the public cloud and accessed by employees and customers
  - Enterprise to Cloud : Cloud applications integrated with internal IT capabilities
  - Enterprise to Cloud to Enterprise: Cloud applications running in the public cloud and interoperating with partner applications (supply chain)
  - Private Cloud: A cloud hosted by an organization inside that organization's firewall.
  - Changing Cloud Vendors: An organization using cloud services decides to switch cloud providers or work with additional providers.
  - Hybrid Cloud: Multiple clouds work together, coordinated by a cloud broker that federates data, applications, user identity, security and other details.
- Vocabulary
  - On-demand self-service
    - Customer purchases cloud services as needed
    - Customer could change their levels of services without actually cotacting a service provider.
  - Broad network access
    - Resources hosted in cloud can be accessed from a wide range of devices, such as tablets, PCs, Macs and smartphones.
  - Resource pooling
    - A situation that providers serve multiple clients, customers with provisional and scalable services.
  - Rapid elasticity
    - The ability to provide scalable services
    - Allows users to automatically request additional space in the cloud.
    - provider needs to allocate and de-allocate resources, that's irrelevant to user's side.

    - Measured service
        - This is a reference to services where the cloud provider measures the provision of services for various reasons, including billing, effective use of resources.
- Specific structure or runtime behaviour
    - SaaS: Software as a service
        - uses web to deliver applications
        - can run directly through a wen browser
    - PaaS: Platform as a Service
        - PaaS are used for applications, and other development, while providing cloud components to software.
        - offers a framework that developers can build upon to develope or customize applications.
        - developer take care of application, and the third party provider handle all others
    - IaaS: Infrastructure as a Service
        - IaaS are self service models for accessing, monitoring, and managing remote data center infrastructures.
        - Instead of purchase hardware outside, users can purchase IaaS based on consumption just like unities.
- Advantage
    - Improved disaster recovery: if a company's business data got destroyed, then they can easily retrieve data from cloud. It's easier and less expensive if their data in the cloud is the most up-to-date information.
    - Cost efficient: unlike traditional hosting, company doesn't need to spend lots of money on maintaining server and licensing fee, since it's pay-as-you-go.
    - Less management effort: since all resources are maintained by the service provider.
- Disadvantage
    - need very good connection to access cloud server all the times.
    - cloud service can become overwhelmed.
- NFP
    - Improve
        - Scalability
            - Information and data can be easily added and peak load.
        - Reliability
            - If server fails, all hosted application and services can easily be transfer to other available servers.
        - Maintainability
            - Minimal management effort required to change software or hardware components
        - Agility
            - Cloud environments can usually provide new compute instances or storage in minutes.
        - Security

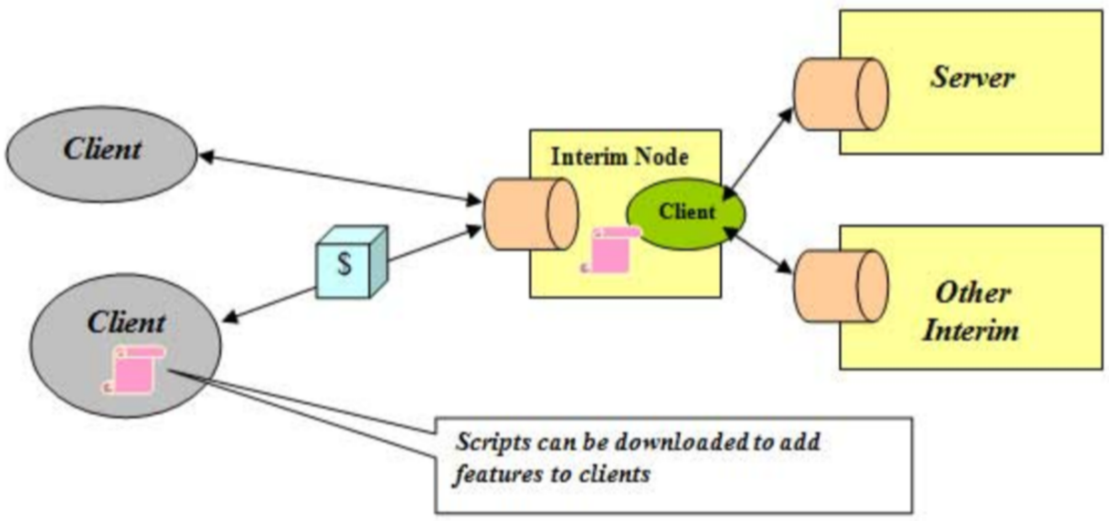
            - Cloud computing gives you greater security when user want to prevent the case of lost laptop or other physical devices that stores important data.
    - Degrade
        - Security
            - user may store information to untrusted third-party cloud provider.
- Senario
    - Blackboard = cloud

REST-based Architecture:

- What is REST-based Architecture
    - Representational State Transfer is the method used to create and in order to communicate with the web services.
- Vocabulary
    - Components
        - Origin Server
            - Use server connector to receive requests, it is the definitive source to representing its resources.
        - User
            - Use client connector to initiate requests.
            - The ultimate recipient of responses.
        - Gateway
            - Act as both server and client, in order to forward data.
    - Connectors
        - Protocols
            - Like HTTP to send, receive and listen requests and responses
        - Cache
            - Located at client, server connector or between them to save cacheable responses
        - Resolver
            - Transforms resource identifiers into network addresses.
        - Proxy
            - intermediary selected by client, to provide interfaces to services.
- Constraints
    - Client-Server
        - Requires the existence of client components to send requests, and server components to receive requests
    - Stateless
        - Server should not remenber the state of application
        - Client should send all information that necessary for executing all request.
    - Cacheable
        - data should be cached in somewhere to reduce total network traffic

- Uniform Interface
- Layered System
    - Intermediaries that can be added between request-response path, to do things on messages like translation, and pass them.
- Code On Demand
    - Optional constraint to allow clients to download and execute code from a server.

- Diagram



- What kinds of problems can be tackled?
    - Problem of high coupling between client and server components can be solved
    - If the application has many different clients, use REST
    - If server is updated frequently, use REST
- Advantage
    - Minimize coupling since it supports stateless communication
    - Light bandwidth needed since the message is JSON format, also it provide cacheability.
    - Can be consumed by any client
- Disadvantage
    - Not suitable for large amount of data.
    - Server has less control over application consistency
- NFP:
    - Support
        - Portability
            - interfaces are portable across multiple platforms
        - Scalability
            - Client-Server structure simplifying server components
        - Visibility
            - A monitoring system doesn't need to look beyond a single request.
            - visibility of interaction is imporved since it's uniform interface.

        - Efficiency is provided
    - Inhibited
        - Performance
            - reduced network performance since client needs to send state information every time it wants to send requests.
- Example
    - Client-Server
        - person A is Client, person B is server. A order an apple, B gives A an apple
        - this case is to show the constraint of client-server in REST
    - Stateless & cacheable
        - A gives B a piece of paper that wrote:” name: A; Like salade, don’t like cheese”, then B gives A a salad without cheese. A comes again, B ask:” Hi, How are you?”, A sends the paper again.
        - This shows server should not remember the state of application, and client should send all information.
        - Also shows that A somehow saved personal information, and doesn’t need to construct it again.
    - Uniform Interface
        - Server B gives same menu to both client A and C.
    - Layered System
        - Client A say:”I want an apple” in chinese. C as intermediary translate it to english and tell Server B.
        - this show the layered system constrain
        - Intermediaries that can be added between request-response path, to do things on messages like translation, and pass them.
    - Code On Demand
        - Client A said to Server B:” please give me recipe of how you made salad.” B give it to A.
        - This shows the optional constraint to allow clients to download and execute code from a server.