Name : Mahima Bisht
EmailId : mahima.bisht@capgemini.com
EmpId : 46010069

9/09/2020

REST FUNDAMENTALS.

△ Challenges for modern distributed Application

→ there are few key drivers that introduces challenges for modern distributed Apps.

1) Heterogeneous Operability.

- this means integration of devices that are built with different frameworks and run on different plateforms.

2) Rise of devices

- y applications built for several devices. to build - need to evolve seperately (device & service, to reduce prevent unstability.

3) Cloud: application architecture should be scalable as service infrastructure.

☆ REST is Network based API

△ Properties of REST

1) Heterogeny - It (REST) helps to solve problem related to heterogenous operability.

2) Scalibility - REST reduces the Comblexities that can occur/exist b/w components in distributed system
- helps in efficiently manage requests
- and scale out horizontally when needed.

Scanned by TapScanner

09/09/2020

3) Evoluability - helps in evolving (client & server) independently of of another.

4) Visibility - REST enables components that monitors applications and intelligent gateways

5) Efficiency - enables components such as proxy servers & caches to participate in handling requests.

6) Performance - caches can improve efficiency and speed of response to user agent.

△ REST - architectural style for distributed system.

- Stands for "Representational state transfer".
- constraints must be satisfied if an interface needs to be reffered as RESTful.

★ every http service is not restFul.

→ constraint driven approach
- in this approach forces are recognised and constraints are applied so that system work with that forces

FORCES

→ Network reliability
→ Latency.
→ Bandwidth
→ Security
→ Network topology
→ administrator
→ transport cost.
→ Heterogenous network
→ Complexity.

Scanned by TapScanner

09/09/2020

Constraint

1) Client - Server : defines interaction b/w nodes in a distributed architecture
   goal : different types of nodes/clients can interact with same servers.
   : clients can evolve independently.
   → Supports heterogenous architecture
   → clients knows about the server only not about other nodes.
   → Server don't knows about any client.

2) Stateless : applies to communication b/w client/server
   - means server should be able to get all info that it needs to process client request from REQUEST only.
   - not from other info.
   - suited for env where client/servers are added & removed.

3) Cache constraint : it says that responses from server must be explicitely labled as cachable/not cacheable.
   - responses can be cached at ~~browses~~
   → browsers
   → company proxy server
   → anywhere along the path to orgin server.

Scanned by TapScanner

8/09/2020

4) Uniform Interface: to help web to scale quickly and reliably.
purpose: is to apply more general software principle to the communication b/w distributed components.
- components uses standard mechanism.

5) layered system constraint:

- says the component in a system can only know about the component in the layer with which it is interacting.

- limit the amount of complexity in a layer

△ Components

→ Software elements that interact with each other.

→ Types: Categorized by ROLF
- Origin Server
- User agent
- Gateway
- Proxy

User Agent → responsible for initiating a request.

* Gateway & Proxy sits b/w Origin Servers and user agent.
* Proxy → represents multiple user agents to the network.
* Gateway → represents multiple origin servers to the network.

Scanned by TapScanner