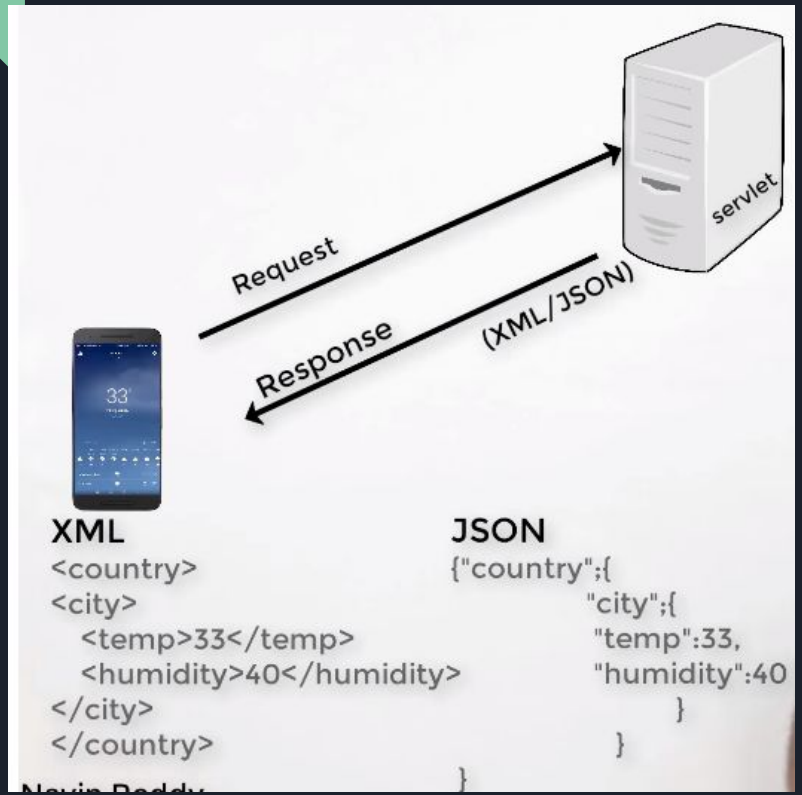
RESTful Service Pattern

Adarsh Venkatesh Bodineni
Yunsheng Guo
Jiacheng Zhang

Introduction



- Structured request and structured response
- What is Web service?
- What is REST?
- What is API?

Architectural Constraints

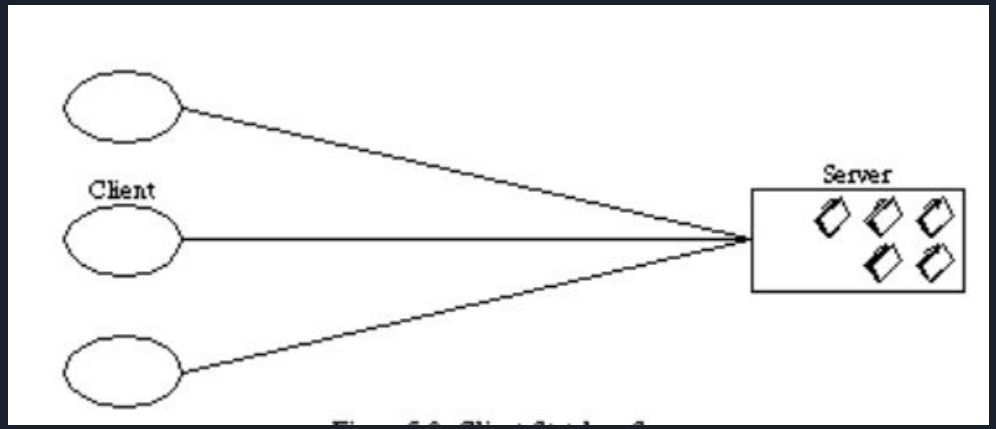
- These constraints restrict the ways the server can process and respond to client requests, so that by operating within these constraints, the system gains desirable non-functional properties
- Performance - how fast the requests can be processed
- Modifiability - of components to changing requirements
- Simplicity - of a uniform interface
- Portability - of components by moving program code with the data
- Reliability - in resistance to failures at system level

- If a system violates any of these constraints then it is not considered Restful

Client-server Architecture

- Separates user interface concerns from data storage concerns
- This improves the portability of user interface across multiple platforms
- This separation helps them to evolve independently

Statelessness

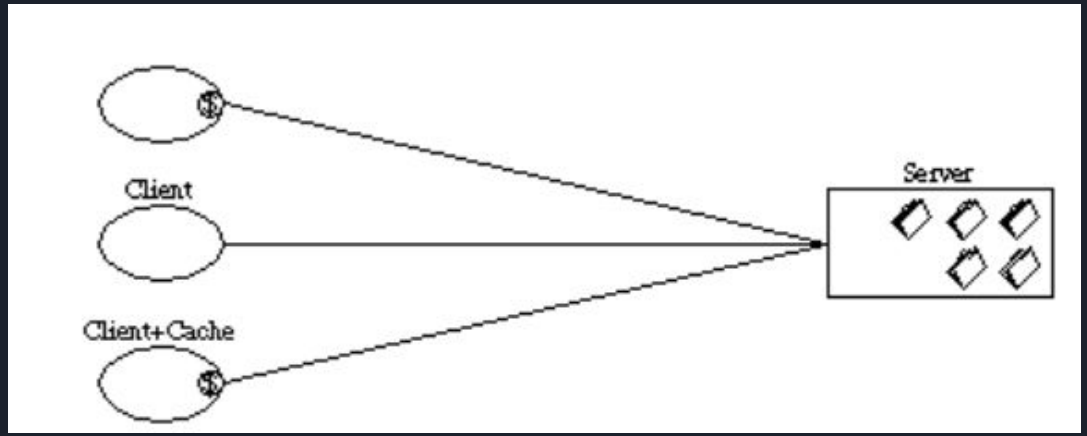


- What is stateless communication between client and server?
- Statelessness is helpful

Overhead on client side:

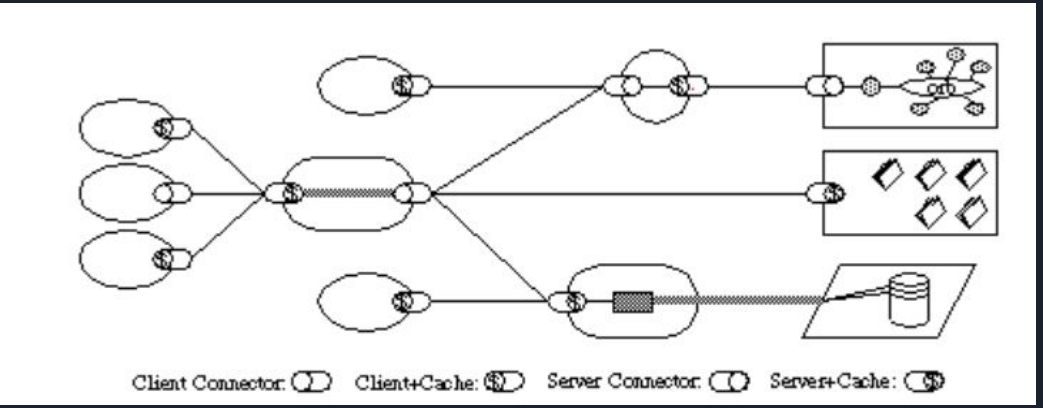
- It makes client to add additional information everytime it makes a request

Cacheability



Responses must implicitly or explicitly define themselves as cacheable
This can improve performance
Decreases reliability - if cache data differs significantly from the data that would have been obtained had the request been directly sent to the server

Layered System

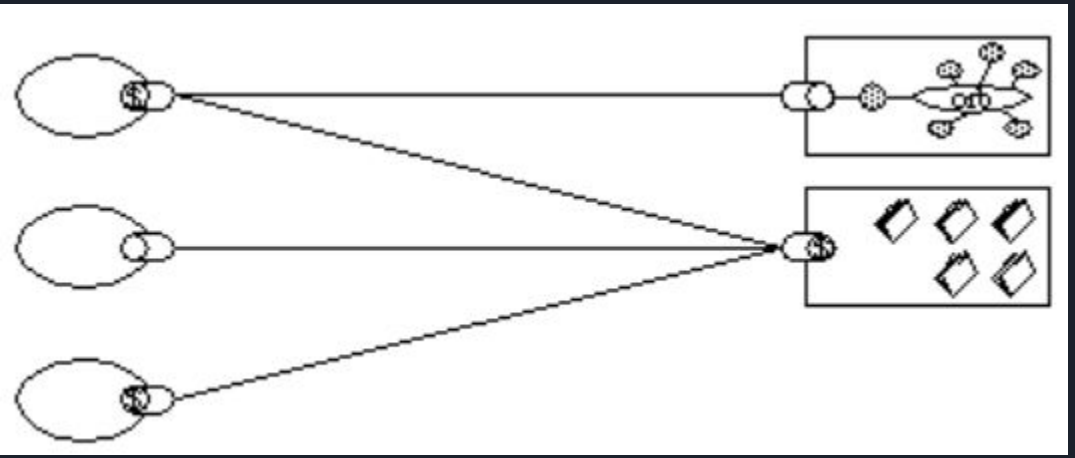


- This does not let a component to see beyond the intermediate layer with which they are interacting
- Improves system scalability by distributing load balance across multiple networks and processes
- Layers allow security policies ondata crossing the organizational boundary as is required by firewalls

Code on demand

- Servers can temporarily extend or customize the functionality of a client by transferring executable code
- Eg: Javascript
- Improves system extensibility

Uniform interface



- The uniform interface is a fundamental constraint to the RESTful system.
- Simplifies and decouples architecture and enables each part to evolve independently
- REST is defined by four interface constraints:
- Identification of resources
- Manipulation of resources through representations
- Self descriptive messages
- Hypermedia as the engine of application state

Architectural Properties

- Performance
  - Net Performance
  - User-Perceived performance
  - Network Efficiency
- Scalability
- Simplicity
- Modifiability
  - Evolvability
  - Extensibility
  - Customizability
  - Configurability
  - Reusability
- Visibility
- Portability
- Reliability

Deriving REST*

- Starting with the Null Style
- Adding constraints

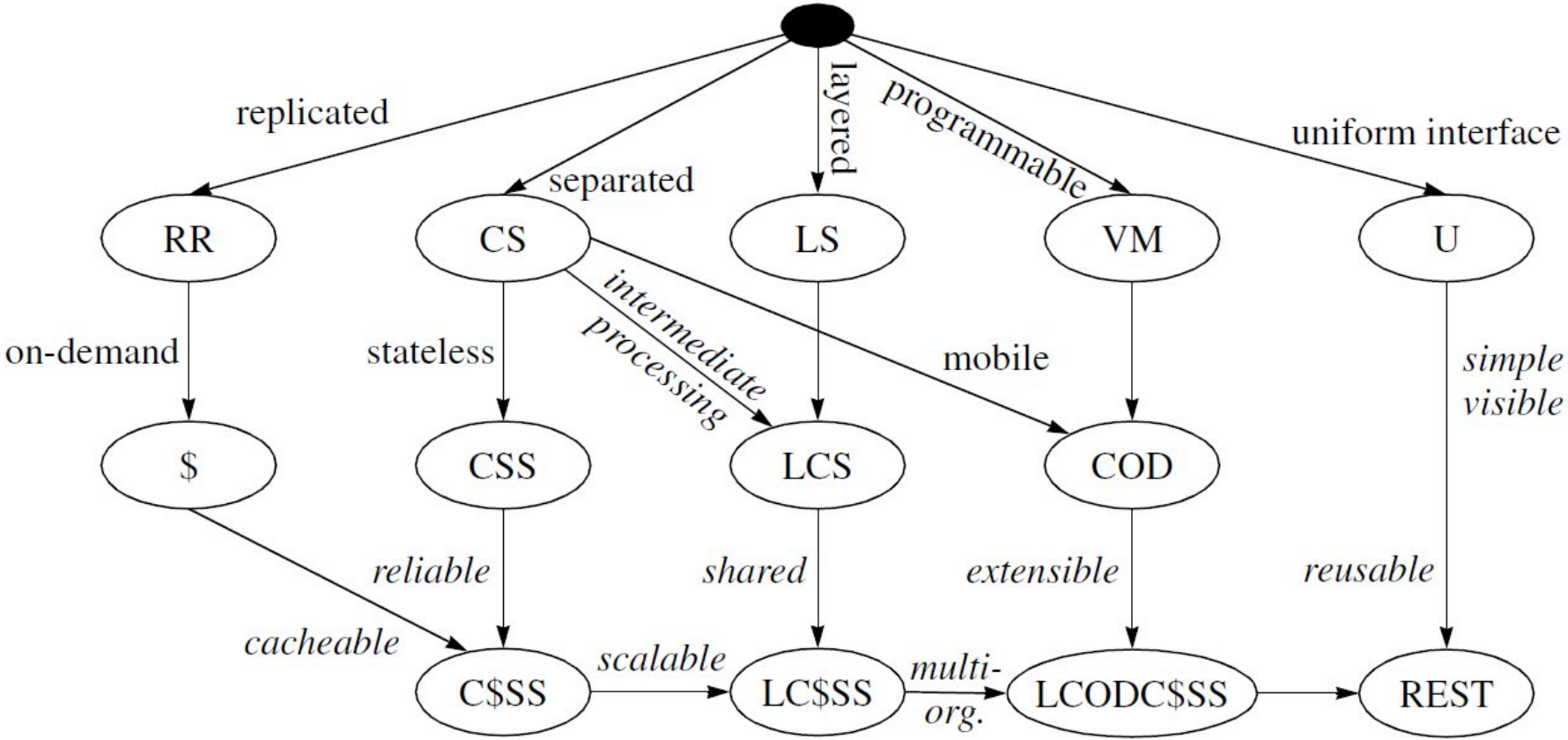
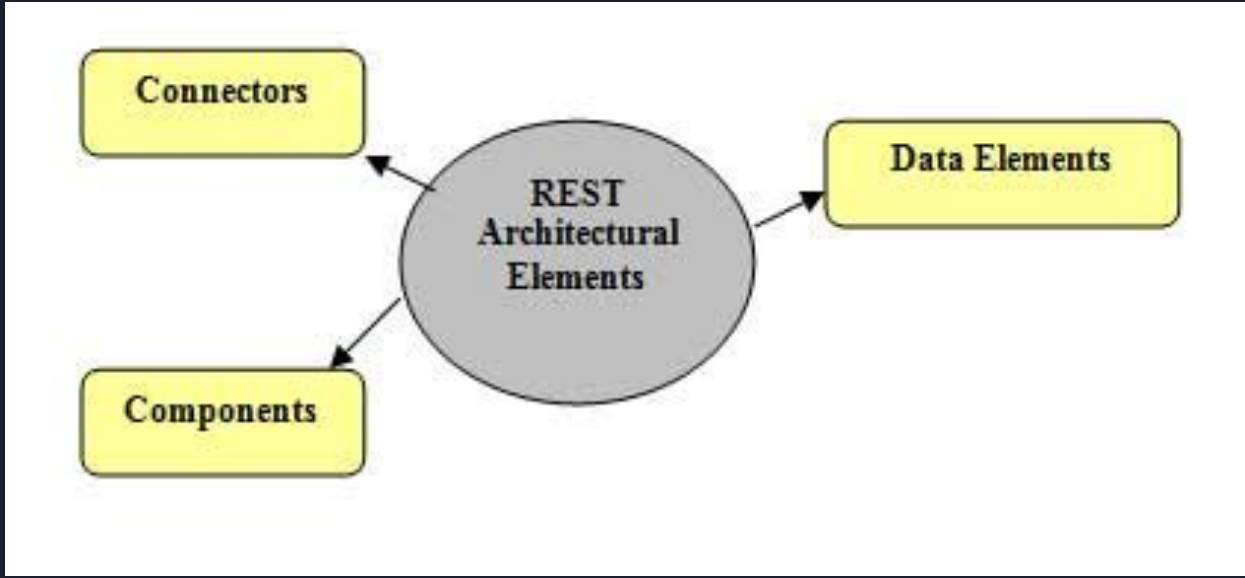


Figure 5-9. REST Derivation by Style Constraints

Style	Derivation	Net Perform.	UP Perform.	Efficiency	Scalability	Simplicity	Evolvability	Extensibility	Customiz.	Configur.	Reusability	Visibility	Portability	Reliability
PF			±			+	+	+		+	+			
UPF	PF	−	±			++	+	+		++	++	+		
RR			++		+									+
$	RR		+	+	+	+								
CS					+	+	+							
LS			−		+		+				+		+	
LCS	CS+LS		−		++	+	++				+		+	
CSS	CS	−			++	+	+					+		+
C$SS	CSS+$	−	+	+	++	+	+					+		+
LC$SS	LCS+C$SS	−	±	+	+++	++	++				+	+	+	+
RS	CS			+	−	+	+					−		
RDA	CS			+	−	−						+		−
VM						±		+				−	+	
REV	CS+VM			+	−	±		+	+			−	+	−
COD	CS+VM		+	+	+	±		+		+		−		
LCODC$SS	LC$SS+COD	−	++	++	+4+	+±+	++	+		+	+	±	+	+
MA	REV+COD		+	++		±		++	+	+		−	+	
EBI				+	− −	±	+	+		+	+	−		−
C2	EBI+LCS		−	+		+	++	+		+	++	±	+	±
DO	CS+CS	−		+			+	+		+	+	−		−
BDO	DO+LCS	−	−				++	+		+	++	−	+	

Table 3-6. Evaluation Summary

Architectural Elements



Connector

Connectors represent the activities involved in accessing resources and transferring representations.

Connector Type	Description	Example
Client	Sending requests, receiving responses.	HTTP library
Server	Listening for requests, sending responses.	Web Server API
Cache	Can be located at the client or server connector to save cacheable responses, can also be shared between several clients	Browser cache
Resolver	Transforms resource identifiers into network addresses.	bind (DNS lookup library)
Tunnel	Relays requests, any component can switch from active behavior to a tunnel behavior.	SOCKS, SSL after HTTP CONNECT

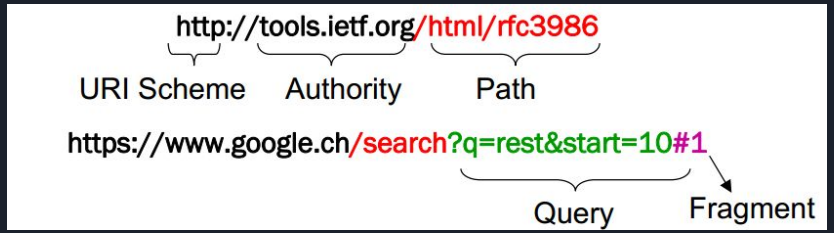
Components

In REST, the various software that interacts with one another are called components.

Origin Server	Uses a server connector to receive the request, and is the source for representations of its resources.	Apache httpd, Microsoft IIS
User Agent	Uses a client connector to initiate a request and becomes the ultimate recipient of the response.	Browser
Intermediary Components	act as both a client and a server in order to forward	Gateway,Proxy

Data Elements

The key aspect of REST is the state of the data elements. Components communicate by transferring representations of the current or desired state of data elements.



· Resource : dynamic or static

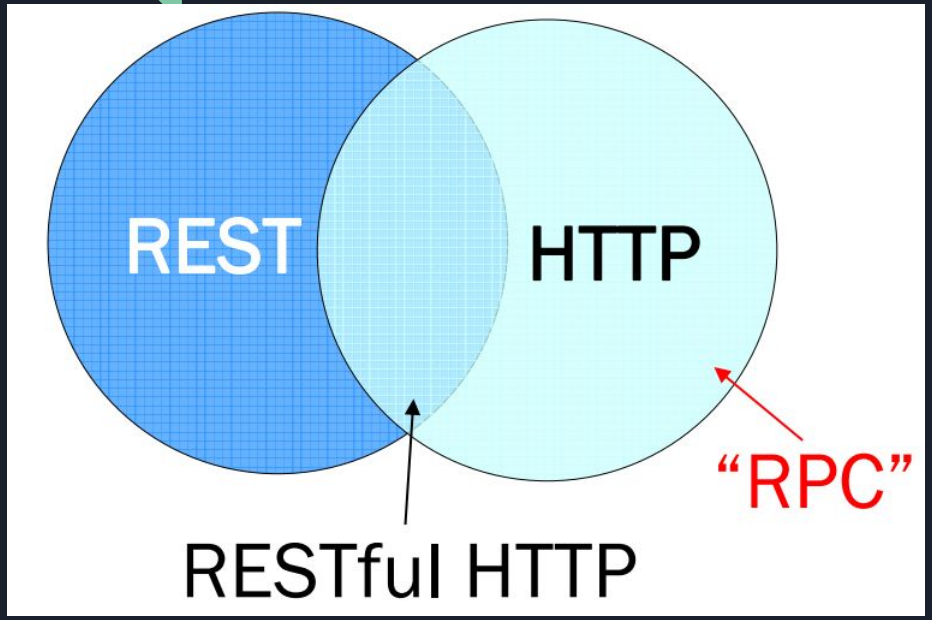
· Resource Identifier: URI

· Resource Metadata: source link, alternates, vary

· Representation: XML, HTML or text

· Representation Metadata: media type, last-modified time

Applied to Web Service



Web service APIs that adhere to the REST architectural constraints are called RESTful APIs. HTTP-based RESTful APIs are defined with the following aspects:

· a base URL, such as http://api.example.com/resources;

· a media type that defines state transition data elements

· standard HTTP methods (e.g., OPTIONS, GET, PUT, POST, and DELETE).

HTTP Methods for RESTful Services

HTTP	CRUD	idempotent
GET	READ	YES
POST	CREATE	NO
PUT	UPDATE/REPLACE	YES
DELETE	DELETE	YES

Request

Request Line: GET /utilities/weatherfull/city/Syracuse HTTP/1.1
Request Method: GET
Request Time: 2018-11-25 15:33:20
Accept-Encoding: gzip, deflate
Accept-Language: en,zh-CN;q=0.9
Host address: restapi.demoqa.com
Client Port: 49652
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36
Request body:

Response

Status Line: HTTP/1.1 200 OK
Response status code -> 200 OK
Server: openresty
Content-Type: application/json; charset=utf-8
Content-Length: 433
X-Cache-Key:
/data/2.5/weather?APPID=199c0c704dcd69f
f1a88fc99385dae08&q=Syracuse
Response Body: {

"City": "Syracuse",
"Temperature": "44 °F",
"Humidity": "70 Percent",
"Description": "cloudy",
"WindSpeed": "9 mph",
}

Questions?

Thank you!