UNIVERSITY OF
Southampton

UNIVERSITY OF Southampton

# RESTful Web Services

COMP3220 Web Infrastructure

Dr Nicholas Gibbins

# Service-Oriented Architecture

Term used to describe a style of software design

- Systems comprise a number of distributed components
- Components interact over a network
- Components provide discrete functionalities with well-defined interfaces: services
- Loose coupling of components

Popular from the early 90s

- DCOM, CORBA, etc

Widespread adoption of the Web gave SOA a new lease of life as Web Services

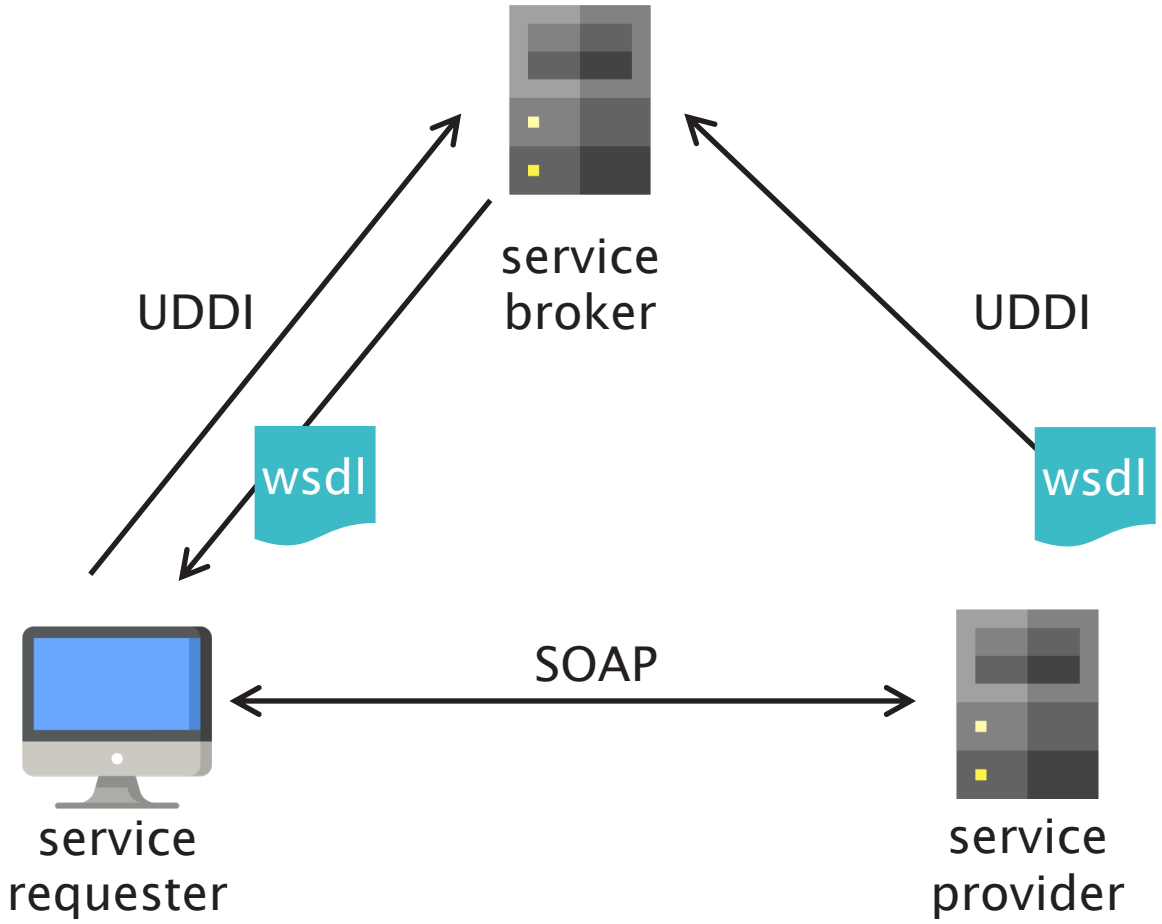
# Web Services

"Traditional" Web Services based on W3C Web Services Architecture (+WS-I, etc)

- SOAP for invoking services
- WSDL for describing services
- UDDI for publishing and locating services

Didn't fit with the Web Architecture

- HTTP used for message transport only (and everything sent as a POST)
- URIs identify endpoints, not resources



# Overcomplication is not a virtue

| Layer | Technologies and Standards |
|---|---|
| **Transport** | HTTP, IIOP, SMTP, JMS |
| **Messaging** | XML, SOAP, WS-Addressing |
| **Description** | XML Schema, WSDL, WS-Policy, SSDL |
| **Discovery** | UDDI, WS-MetadataExchange |
| **Choreography** | WS-CL, WS-CI, WS-Coordination |
| **Business Processes** | WS-BPEL, BPML, WS-CDL |
| **Stateful resources** | WS-ResourceFramework |
| **Transactions** | WS-CAF, WS-AtomicTransactions, WS-BusinessActivities |
| **Reliable Messaging** | WS-Reliability, WS-ReliableMessaging |
| **Security** | WS-Security, SAML, XACML, WS-Trust, WS-Privacy, WS-SecureConversation |
| **Event Notification** | WS-Notification, WS-Eventing |
| **Management** | WS-DM, WS-Management |
| **Data Access** | OGSA-DAI, SDO |

# Resource-Oriented Architecture

Software architecture building on REST and the Web architecture

Components = resources

Key REST constraints

- Client-server
- Stateless
- Uniform interface

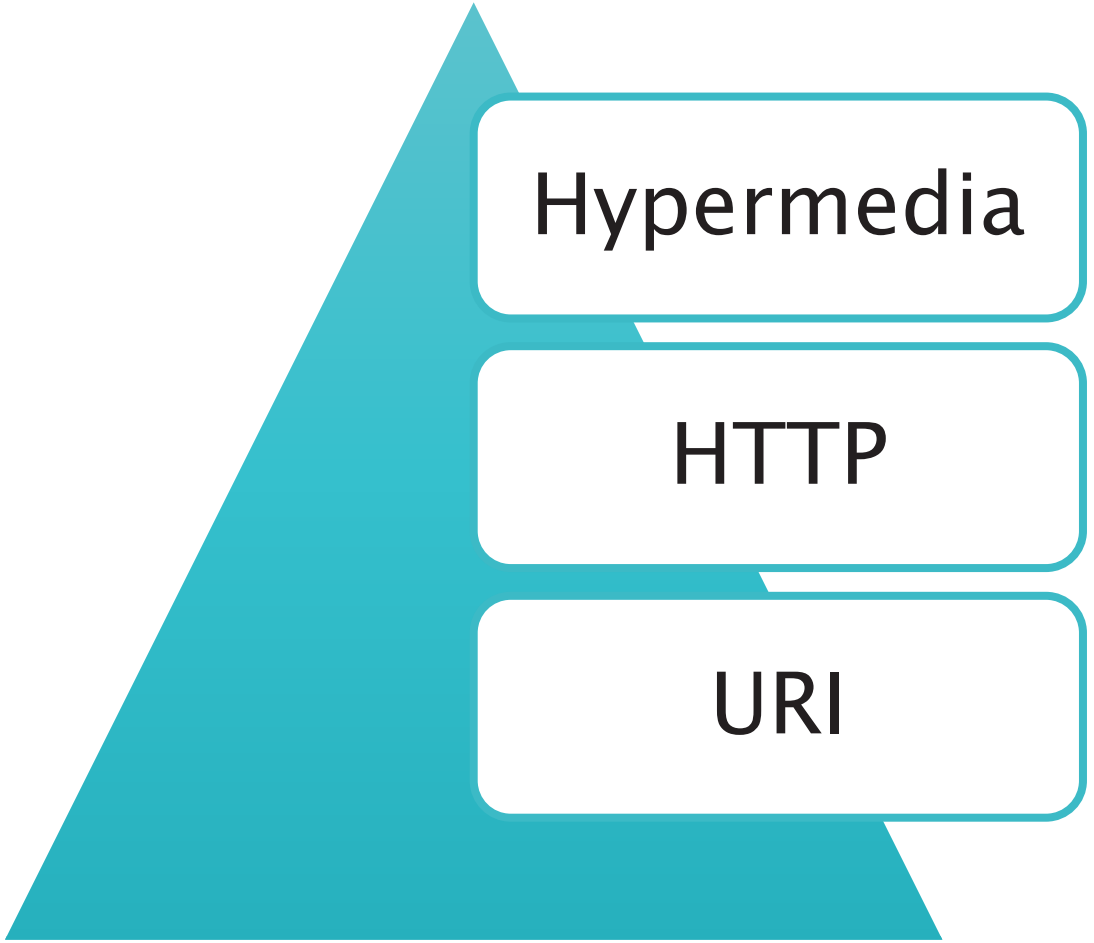
Richardson, L. and Ruby, S. (2007) *RESTful Web Services*. Sebastopol, CA: O'Reilly.

# The Richardson Maturity Model

Model of RESTful maturity

REST constraints made concrete, in context of Web architecture components

- Identification = URI
- Interaction = HTTP
- Representation = Hypermedia




Webber, J. et al (2010) *REST in Practice*. Sebastopol, CA: O'Reilly

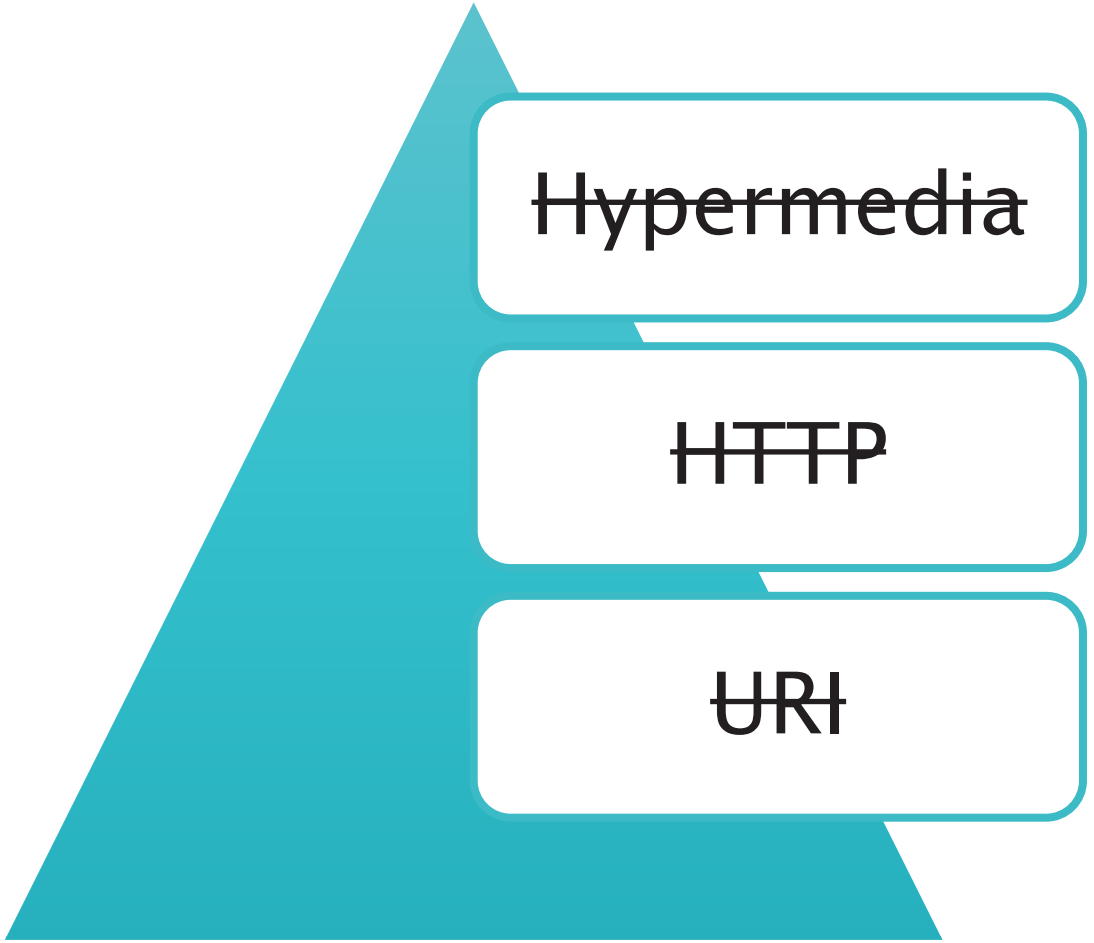
# Level Zero

Service has a single well-known endpoint

- URI identifies endpoint, not service
- Uses HTTP as a transport only
- No hypermedia to express state/protocol

Examples:

- XML-RPC
- Most SOAP services
- POX – Plain Old XML over HTTP



# POX: Plain Old XML over HTTP

Use HTTP POST to transfer XML documents between systems

- SOAP-like, without SOAP headers, etc
- Platform independent

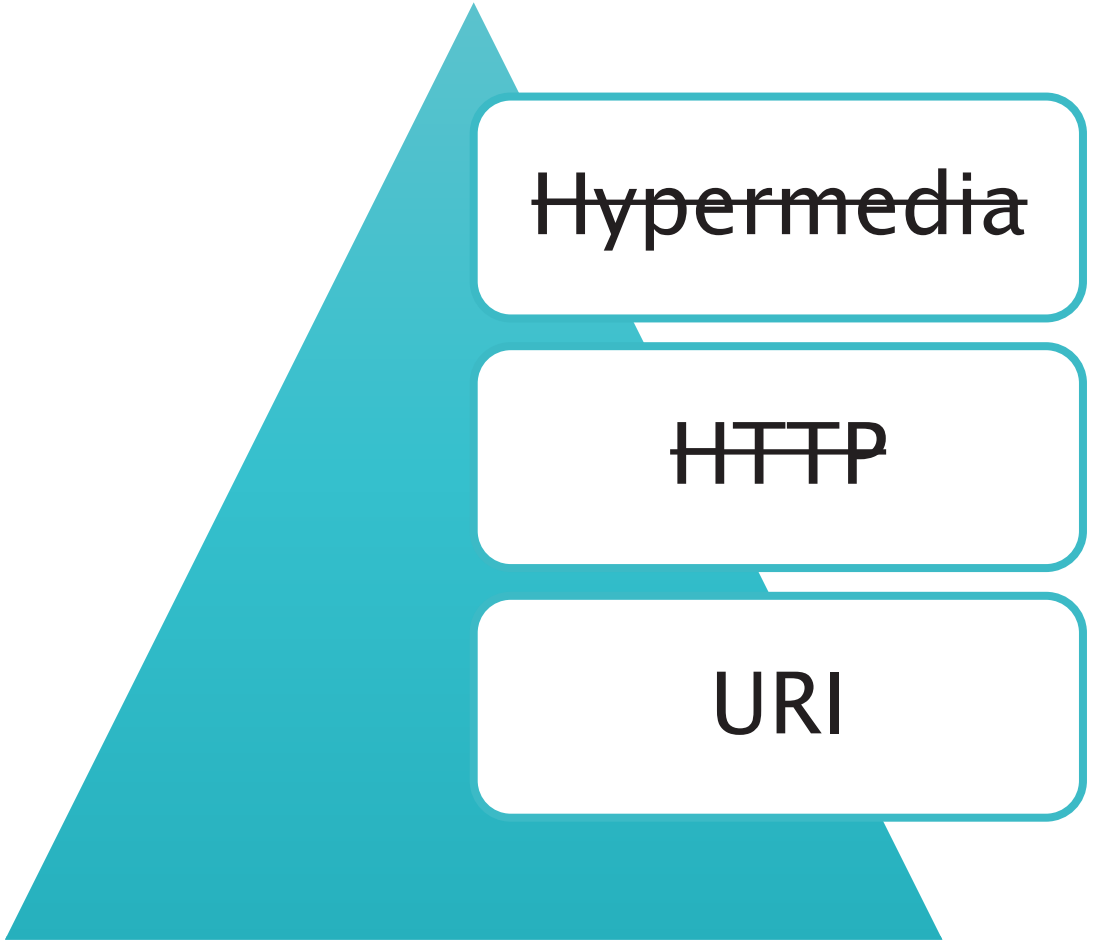
Uses HTTP as a *transport protocol*, not an *application protocol*

- HTTP metadata is ignored

# Level One

Multiple endpoints

- URIs identify different operations, possibly different resources
- Uses HTTP as a transport only (single verb, usually GET or POST)
- No hypermedia to express state/protocol



# Templates and Tunnelling

URI templates

- `http://restbucks.com/order/{order_id}`

URI tunneling

- Use GET for safe/idempotent operations
- Use POST otherwise
- Map existing method parameters to URI query parameters
  `http://restbucks.com/PlaceOrder?coffee=latte&size=large`

# URI Tunnelling

*Sometimes* Web-friendly

- Multiple URIs used to name resources
- (Sometimes) correct use of GET

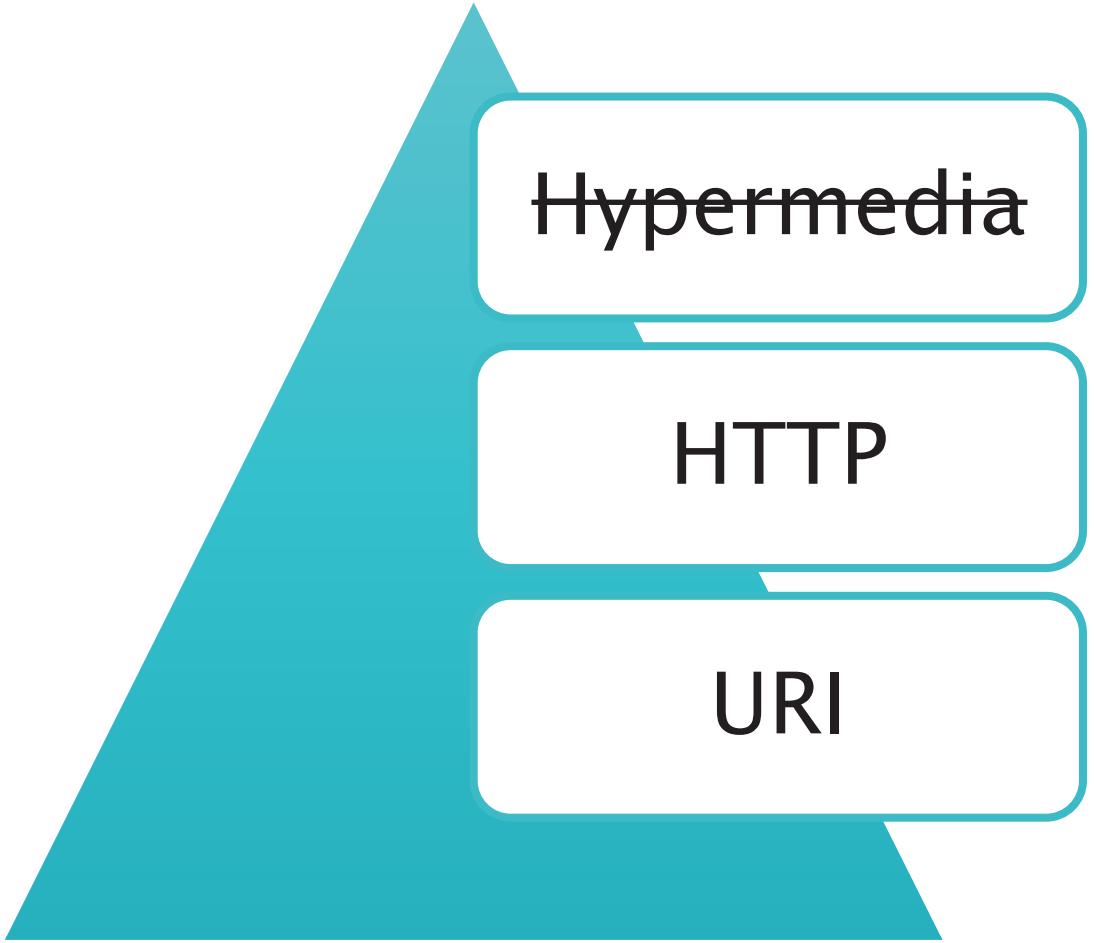
but

- URIs often used to encode operations, rather than identify resources
- Use of GET for non-safe operations breaches Web Architecture principle of Safe Retrieval

# Level Two

Multiple endpoints, identifying resources

- Understands HTTP (multiple verbs)
- No hypermedia to express state/protocol



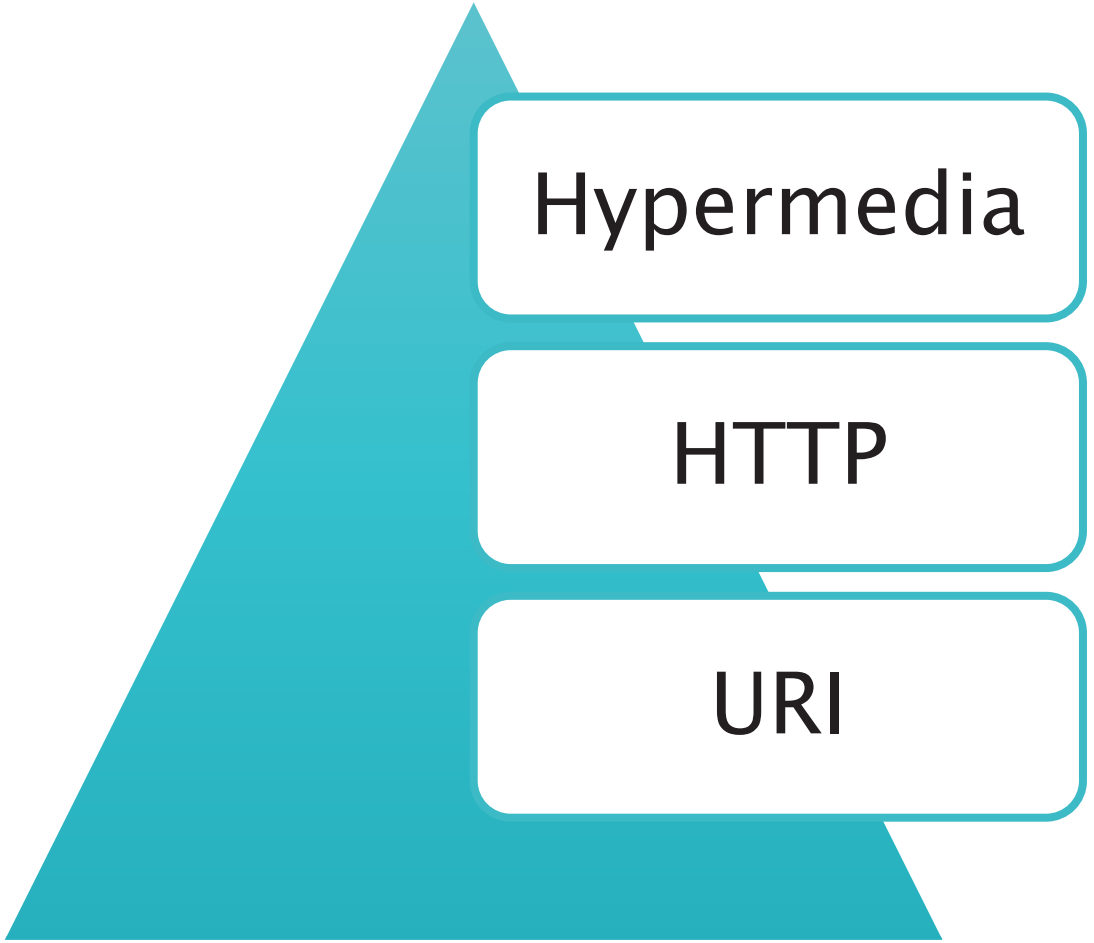
# Mapping CRUD to HTTP

| Operation | HTTP Verb |
|---|---|
| Create | POST to a URI, creates new subordinate resource |
| Read | GET, returns representation |
| Update | PUT, providing new representation |
| Delete | DELETE, removes resource |

# Level Three

Multiple endpoints, identifying resources

- Understands HTTP (multiple verbs)
- Uses hypermedia to communicate protocols (shared state)



# HATEOAS

Hypermedia As The Engine Of Application State

Resource representations include links to related resources

- Links that represent a transition to a possible future state of the current resource
- *Link types* used to indicate operations (HTTP verbs?) that may be used on the indicated resources
- State transitions constitute application protocol

# Further Reading

Richardson, L. and Ruby, S. (2007) *RESTful Web Services*. Sebastopol, CA: O'Reilly.

Webber, J. et al (2010) *REST in Practice*. Sebastopol, CA: O'Reilly

Webber, J. et al (2009) *REST in Practice: A Tutorial on Web-based services*
https://www.slideshare.net/guilhermecaelum/rest-in-practice

UNIVERSITY OF Southampton

# Next Lecture: REST in Practice