SAKARYA ÜNİVERSİTESİ
Bilgisayar ve Bilişim Bilimleri Fakültesi
Bilgisayar Mühendisliği Bölümü

BSM 451

Internet of Things (IoT) and Applications

IoT Protocols-(REST, SOAP, XMPP)

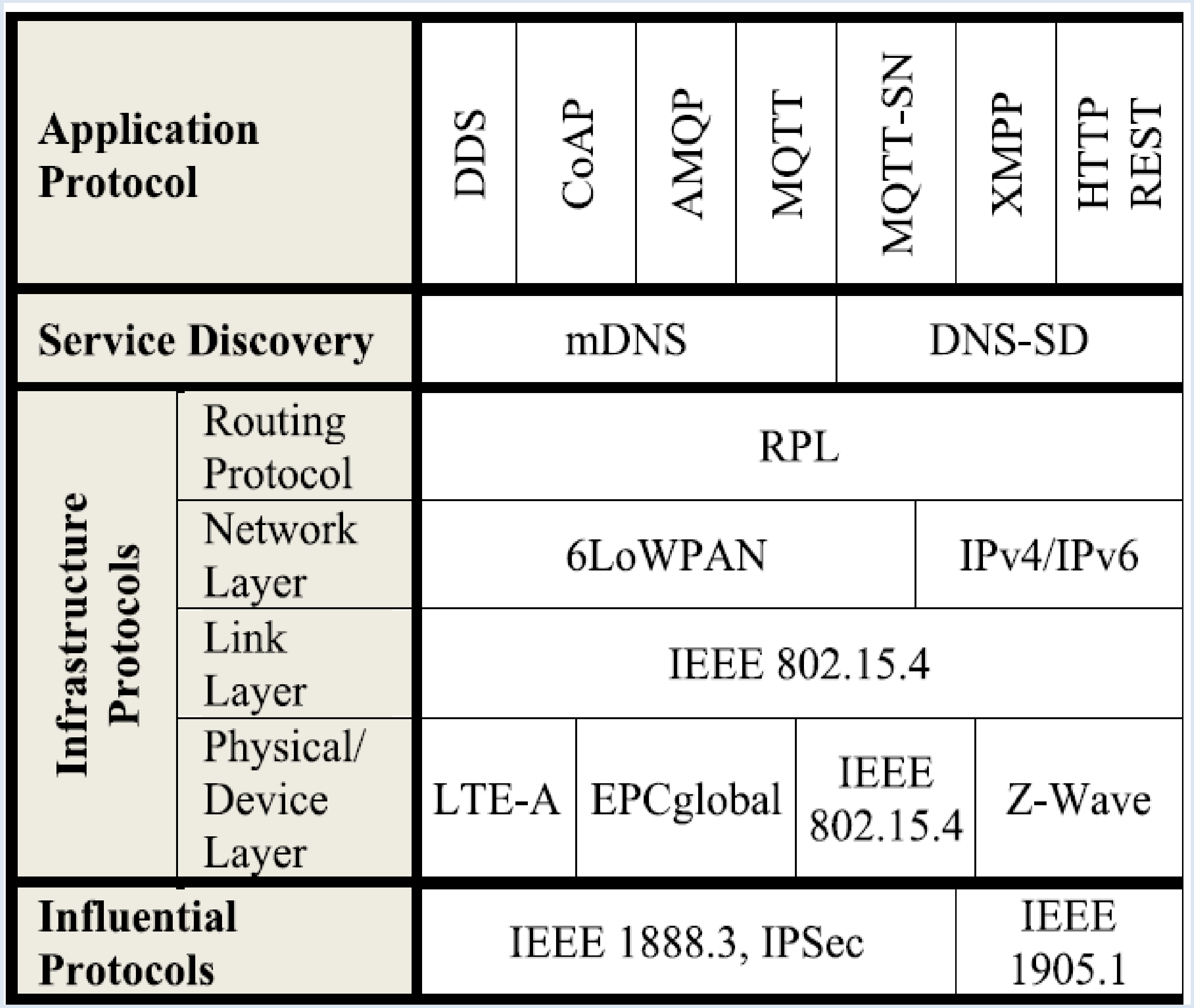
Assoc. Prof. Cüneyt BAYILMIŞ
Researcher Dr. Ünal ÇAVUŞOĞLU

Application Protocols of IoT

- REpresentational State Transfer, (REST)
- Simple Object Access Protocol, (SOAP)
- Extensible Messaging and Presence Protocol, (XMPP)
- Constrained Application Protocol, (CoAP)
- Messega Queue Telemetry Transport, (MQTT)
- Advanced Message Queuing Protocol, (AMQP)
- Data Distribution Service, (DDS)
- OMA LightwieghtM2M (LWM2M)
- oneM2M

IoT used Protocols

- The most commonly defined protocols of organizations such as IEEE, ETSI.



Application Protocol		DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	HTTP REST
Service Discovery		mDNS				DNS-SD		
Infrastructure Protocols	Routing Protocol	RPL						
	Network Layer	6LoWPAN				IPv4/IPv6		
	Link Layer	IEEE 802.15.4						
	Physical/ Device Layer	LTE-A	EPCglobal	IEEE 802.15.4	Z-Wave			
Influential Protocols		IEEE 1888.3, IPSec					IEEE 1905.1	

Overview of IOT Application / Communication Protocols

- Software systems designed to support machine interaction / communication on a network-capable machine are called 'Web Services'.

- The concept of Web Services is covered in the application / communication protocols of IOT.

- Resource usage of IOT devices with limited hardware in the selection of Application / Communication Protocol CPU, memory, etc. The need for effective use of resources must be considered. For this reason, IOT applications require communication with lightweight protocols.

Overview of IOT Application / Communication Protocols

- It is possible to classify IOT application / communication protocols according to two basic architectures.

- Bus-based
  Clients publish messages that directly reach subscriptions for a specific topic.
  There is no centralized server or server based service.
  DDS, REST, XMPP
- Server-based (Broker-based)
  The server checks the delivery of the information / message.
  It stores, transmits, filters, prioritizes, and publishes server information / messages, publishing requests from publishers (publishers) to subscribers.
  Clients are switches between publisher and subscriber roles, depending on the purpose of the application.
  AMPQ, CoAP, MQTT, Java Message Service API (JMS)

- IoT application / communication protocols can also be classified in terms of messages.

- Message-centric

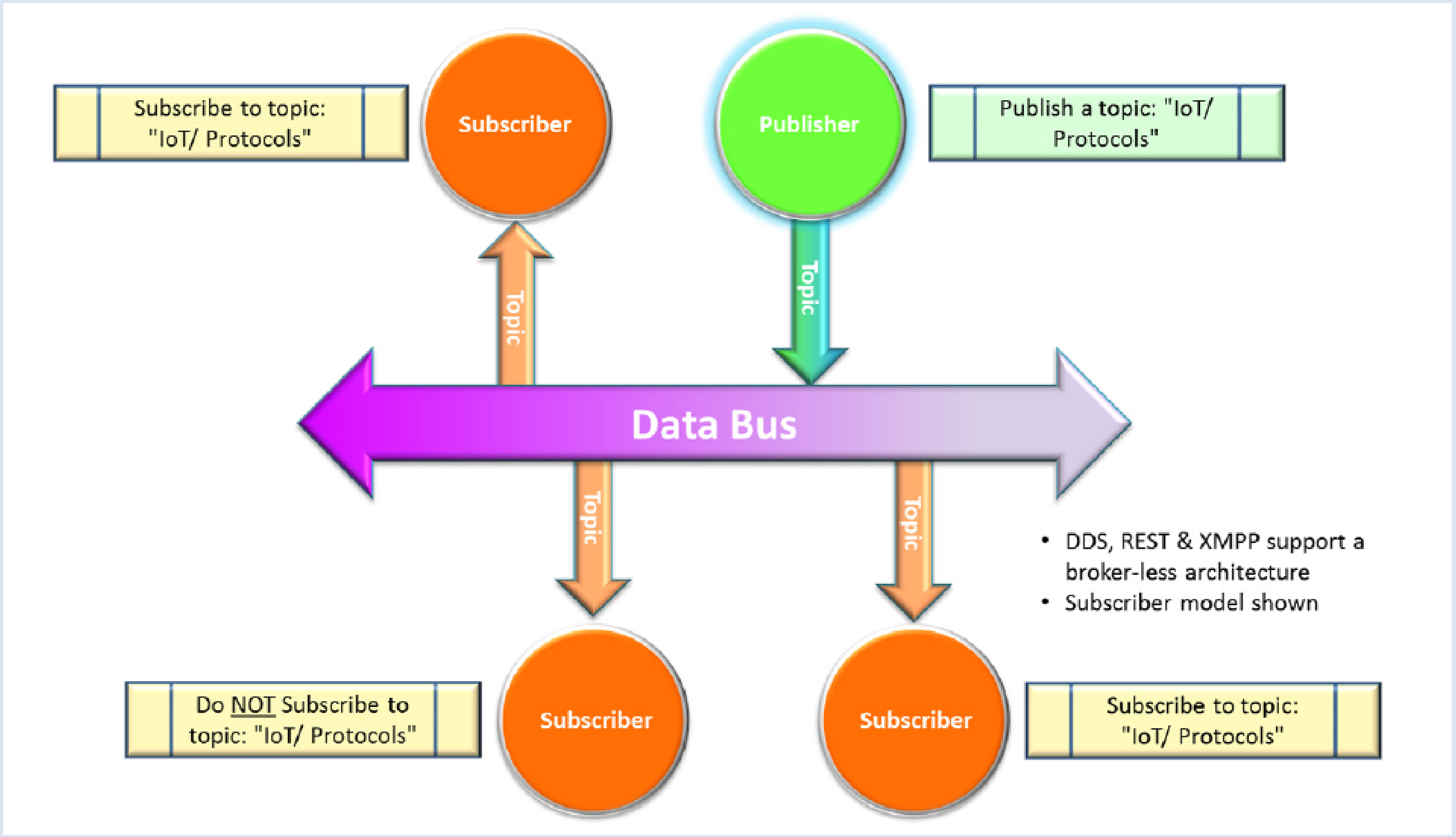
It focuses on delivering the message / payload to its payloads regardless of its content.
AMQP, MQTT, JMS, REST

- Data-centric

Assumes that the message is understood by the recipient and focuses on the delivery of the data
DDS, CoAP, XMPP

Overview of IOT Application / Communication Protocols

- Bus-based



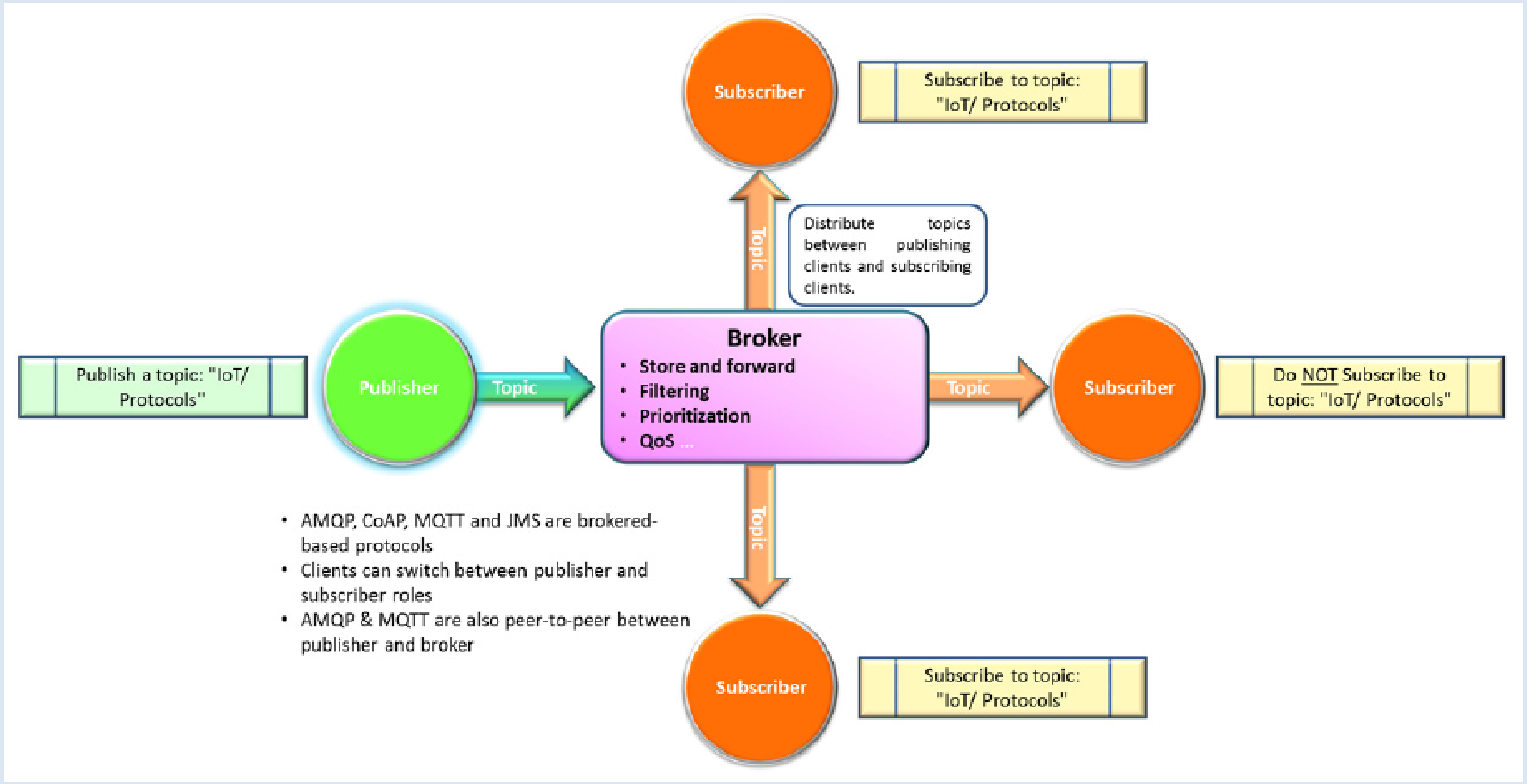
Subscribe to topic: "IoT/ Protocols"
Subscriber
Publisher
Publish a topic: "IoT/ Protocols"
Topic
Topic
Data Bus
Topic
Topic
Do NOT Subscribe to topic: "IoT/ Protocols"
Subscriber
Subscriber
Subscribe to topic: "IoT/ Protocols"

- DDS, REST & XMPP support a broker-less architecture
- Subscriber model shown

Kaynak: O. Vermesan, P. Friess (Editors), ‘‘Internet of Things – From Research and Innovation to Market Deployment’’, River Publishers, 2014.

Overview of IOT Application / Communication Protocols

- Broker-based


Subscriber
Subscribe to topic: "IoT/ Protocols"
Topic
Distribute topics between publishing clients and subscribing clients.
Publish a topic: "IoT/ Protocols"
Publisher
Topic
Broker
• Store and forward
• Filtering
• Prioritization
• QoS ...
Topic
Subscriber
Do NOT Subscribe to topic: "IoT/ Protocols"
Topic
Subscriber
Subscribe to topic: "IoT/ Protocols"

- AMQP, CoAP, MQTT and JMS are brokered-based protocols
- Clients can switch between publisher and subscriber roles
- AMQP & MQTT are also peer-to-peer between publisher and broker

Kaynak: O. Vermesan, P. Friess (Editors), ''Internet of Things – From Research and Innovation to Market Deployment'', River Publishers, 2014.

Source information(XML ve JSON)

- In the XML structure, header tags are used in data definitions.
- In the JSON structure, data definitions are made with punctuation marks.
- JSON data size is less than XML.

XML

```
 <?xml version="1.0" Encoding=="UTF-8">
<sensor>
    <_class>cihaz.Model </_class>
   <cihazID> 4e22c934e7</cihazID>
   <cihazAdi>Termometre </cihazAdi>
   <sicaklik> 23.00</sicaklik>
</sensor>
```

JSON

```
{
       "_class" : "cihaz.Model"
         "cihazID" : " 4e22c934e7 "
         "cihazAdi : " Termometre "
         "sicaklik" : " 23.00 "
}
```

REpresentational State Transfer, REST

(REpresentational State Transfer, REST)

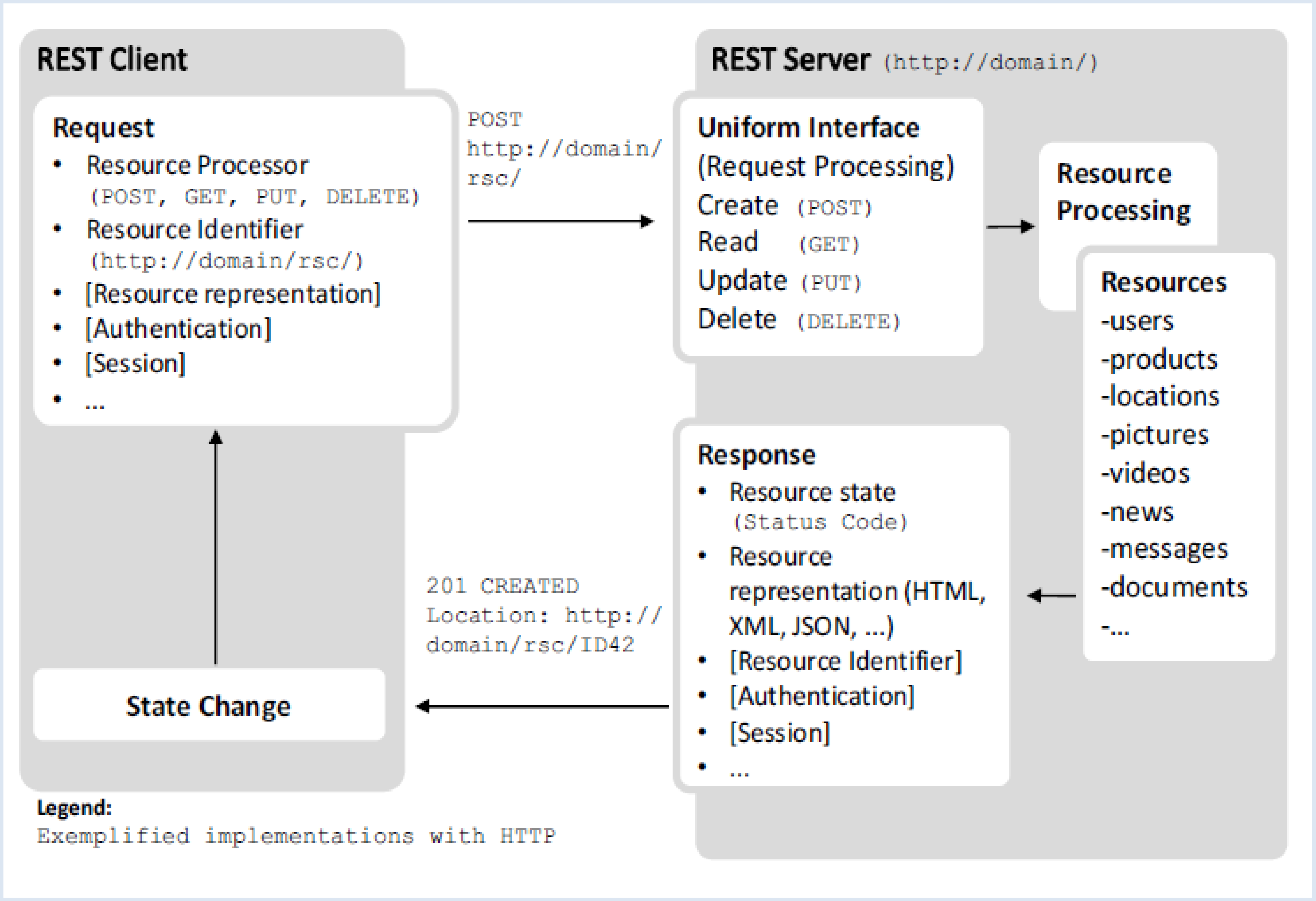
- REST is an operating system-independent architecture for designing network applications that use simple HTTP to connect machines.
- It is a phrase used to describe an architectural style for networking systems in Fielding's work with the doctor.
- REST is not a standard, including HTTP, URL, XML, etc. is an architectural approach that incorporates standards.
- It has peer-to-peer (P2P) communication structure based on client / server and request / response model.
- Services that use the REST architecture are commonly referred to as RESTful services.
- It is widely used in web-based applications because it is simple, flexible and easily extensible.
- Anything that can be done with other web services can be done with RESTful services.

(REpresentational State Transfer, REST)

- It communicates via POST, GET, and DELETE methods over HTTP.
- Platform independent (Windows, Linux).
- The programming language is independent (C #, JAVA, etc.).
- It uses TCP as the transport layer.
- Service quality is not supported.
- Security https
- In the message format, the header is in ASCII format, while the data / payload is in XML, JSON, and HTML.
- It uses more power, resources and data than other protocols.

(REpresentational State Transfer, REST)

- REST Web Service structure


REST Client
Request
• Resource Processor (POST, GET, PUT, DELETE)
• Resource Identifier (http://domain/rsc/)
• [Resource representation]
• [Authentication]
• [Session]
• ...
State Change
POST http://domain/rsc/
201 CREATED Location: http://domain/rsc/ID42
REST Server (http://domain/)
Uniform Interface (Request Processing)
Create (POST)
Read (GET)
Update (PUT)
Delete (DELETE)
Resource Processing
Resources
-users
-products
-locations
-pictures
-videos
-news
-messages
-documents
-...
Response
• Resource state (Status Code)
• Resource representation (HTML, XML, JSON, ...)
• [Resource Identifier]
• [Authentication]
• [Session]
• ...
Legend:
Exemplified implementations with HTTP

Kaynak: P. L. Gorski, L. L. Iacono, H. V. Nguyen, D. B. Torkian, ''Service Security Revisited'', IEEE International Conference on Services Computing, 464-471, 2014

(REpresentational State Transfer, REST)

Features and Restrictions of REST Web Service (Design Principles)

- (Client – Server)
  - The basic design (the first restriction) is the client - server architecture.
  - It means that the structure of the client (user interfaces) and the server (data storage units) are separate.
  - The separation of the client and the server allows for increased platform-independent operation (portability of the user interface) and scalability by simplifying server elements.
  - The client does not know anything about the data source and responds as long as the server receives the correct requests.
- (Stateless)
  - Statefulness is that the application is not in any state when the client-server system is communicating.
  - Since the system is not in any state, requests from clients at any time will return with appropriate response content by the server.
  - No information or session about the client is kept on the server. Each request made by the client carries the necessary information for the server.

(REpresentational State Transfer, REST)

Features and Restrictions of REST Web Service (Design Principles)

❑ Cache

- Increases network performance. It allows the user to respond faster.
- If the data between the request and the reply is marked as cacheable by the server, this data is cached by the client (user) and can then be used again for the same requests.

❑ Uniform Interface

- The uniform (common) interface for the server and the client simplifies the communication method. This makes data communication easier to follow.
- It enables the development of applications independently of each other.
- Using a common interface can reduce productivity as it provides a certain standard instead of considering the specific requirements of applications, but encourages interoperable application development.
- The REST interface is designed for broadband communications.

(REpresentational State Transfer, REST)

Features and Restrictions of REST Web Service (Design Principles)

- Layered System
  - The client may not connect directly to the last server in any case, there may be other servers.
  - A layered system provides a hierarchical architecture, and each layer knows only the adjacent (lower or upper) layer.
  - The disadvantage of the layered system is that it causes additional delays.

- Code-On-Demand
  - An optional restriction.
  - If needed, the client can access some resources, but does not know how to use these resources.
  - In certain instances, the server may send client-side scripts and applications.

(REpresentational State Transfer, REST)

HTTP Methods Used in RESTful Web Services

- PUT
  - In the SQL database applications, the insert operation is performed in a uniform interface.
  - Used to create a new resource.
  - The Web Service gives you success or incorrect response.
- GET
  - Performs Read operation in select SQL database applications
  - Used to query a source or to retrieve data from a source.
  - The data returned from the source to the user is the representation of the requested resource.
- POST
  - Performs the update function in uniform and SQL database applications.
  - Enables updating of an existing resource or data.
  - It can also be used to create a new resource and delete an existing resource.
- DELETE
  - Tek biçimlilikte ve SQL veri tabanı uygulamalarında silme (delete) eylemini gerçekleştirir
  - Mevcut bir kaynağın ya da verinin silinmesini sağlar.

Simple Object Access Protocol, SOAP

Simple Object Access Protocol, SOAP

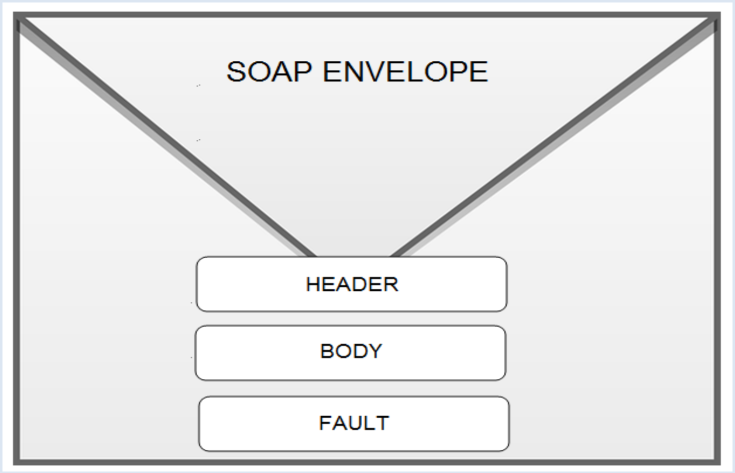
- It has similarities with the REST architecture.
- SOAP is designed for structural information exchange by non-centralized, distributed environments.
- The platform uses XML to send and define information to provide an extensible extensible format. Expand the HTTP protocol for XML messaging.
- It can be used for broadcast messages.
- It is a Client - Server architecture-based protocol that uses the Remote Procedure Call (RPC) model.
- The Internet Committee is the W3C (World Wide Web Consortium) standard.
- Provides data communication for Web Services.

Simple Object Access Protocol, SOAP

- SOAP consists of 4 basic parts.
- SOAP Envelope
  - An envelope that describes the message content and how to handle it (SOAP creates the entire framework of the message)
- SOAP Encoding Rules
  - Includes encoding rules for data types that can be defined by applications.
- SOAP RPC Representation
  - Define remote procedure call and return.
- Protocol Binding
  - Defines how to use SOAP in HTTP.
  - In other words, a connecting procedure for message exchange

Simple Object Access Protocol, SOAP

- The message structure called SOAP envelope consists of the Header, Body and Fault fields.
  - The envelope forms the entire SOAP message.
  - The header consists of data from which the optional features of the message providers are set.
  - The body generates the actual message and contains the XML data.
  - The error is optional and includes the error in the function.



```
 <?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Header>
   <m:Trans xmlns:m="http://www.example.com/transaction/">100 </m:Trans>
</soap:Header>

<soap:Body>
  <m:GetName xmlns:m="http://www.example.com/customers">    <m:Item>1234</m:Item>
     </m:GetName>
    </soap:Body>

 </soap:Envelope>
```

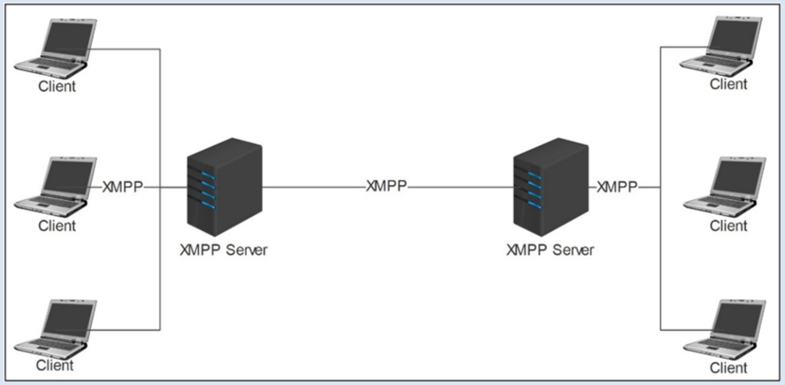
RESTful and SOAP Comparison

- RESTful reduces the latency and increases the speed with pre-storage. RESTful also uses lower data sizes because it does not use the SOAP envelope structure.
- RESTful systems are more advanced in server scaling than SOAP. This is because RESTful needs less access to session states.
- In RESTful, applications and resources can be accessed through browser programs, allowing less software to be run on the client side as compared to other services.
- SOAP is a service that is older than the enterprise as a software support. The use of RESTful is increasing day by day.
- While RESTful uses current internet standards, SOAP-based applications / approaches include their own standards.
- SOAP services are preferred for inter-system compatibility applications.
- The design and development of SOAP services is more complex because it offers security, recording, and some communication models such as synchronous, asynchronous, request / recall, warning.

Extensible Messaging and Presence Protocol, XMPP

Extensible Messaging and Presence Protocol, XMPP

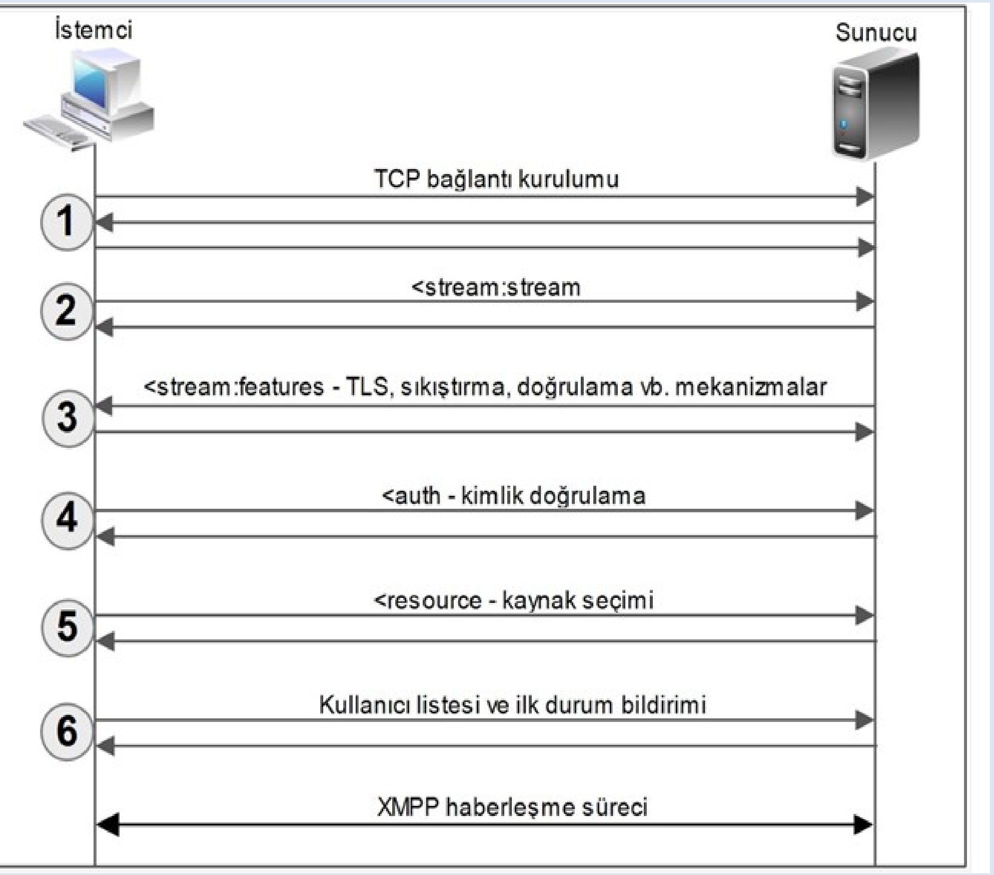
- XMPP is an application protocol developed for messaging, status management and request-response services that can be considered real-time in the extensible markup language (XML) base.
- XMPP is defined by the RFC 3920 and RFC 3921 published by the Internet Engineering Task Force (IETF).
- It was put forward by Jeremie Miller in 1999 for instant messaging. Originally known as Jabber.
- XMPP can create autonomous networks between two servers and services between two clients, which can provide P2P (Peer-to-Peer) communication and can connect to each other.
- XMPP is a real-time communication protocol that uses the XML packet format for data transmission between clients.
- XMPP allows users to communicate with each other via instant messaging over the internet.



XMPP Adressing

- Each entity on the XMPP network has a single address called JID (jabber id).
- A JID ID has local name, domain name, and resource declarations in the user @ domain / resource (user@domain.com/ resource) structure.
- The domain name field that should be included in each XMPP ID is used to identify the subnet within that XMPP network. The jid structure obtained with the combination of the user name and the field name field (user@domain.com) is named as bare jid.
- The resource notification is used to define which client uses which resource a user can use when it is able to establish connections over different clients.
- The address structure of these three components that make up a jabber identity is called full jid. Basically bare jid refers to the user in the XMPP network, while the full jid is the instant session.

XMPP Singnalling


İstemci
Sunucu
TCP bağlantı kurulumu
1
<stream:stream
2
<stream:features - TLS, sıkıştırma, doğrulama vb. mekanizmalar
3
<auth - kimlik doğrulama
4
<resource - kaynak seçimi
5
Kullanıcı listesi ve ilk durum bildirimi
6
XMPP haberleşme süreci

❑ In general, an XMPP signaling process is structurally as follows:

- Establishing a long-term TCP connection with the server,
- Launching XML streams,
- Sharing of various XML schemas,
- Verifying the client ID on the server,
- Provision of resource selection,
- It takes the form of user lists before the instant messaging process and the notification of the first state to the server.

This slide was taken from the phd thesis of Halil Arslan.

XMPP Packet Structure

- After XMPP signaling is performed, the communication flow is maintained over the connection via the XML packet structures defined in XMPP.
- XML packet structures are called XML Stanza.
- In XMPP, the XML packet format has 3 basic definition fields.

  message (<message>),

  It can be used to transfer any information between users (client).

  presence (<presence>), (STATUS)

  checks and reports the presence of a user on the XMPP network. Briefly, it is the package structure used to manage a user's online, busy, offline status on the XMPP network. This allows users to control each other's accessibility.

  Information/query (<iq>)

  It is designed to be used as a information / query package when a data that is intended to be transmitted between the client and server does not match the message or status packet structures.

  Information / query packages are generally positioned as request / reply pack. This package structure is customized to ensure the extensibility of the protocol.

  This package structure is used especially in HTTP architecture such as GET, POST, PUT etc. It provides a structure for workflows to provide the request-response interaction as well as its methods.

  Each of these packet structures is processed differently by clients as well as being routed differently by the server.

XMPP Packet Structure

- XMPP package structures consist of attribute: value definitions that determine how the package will be handled by recipients.
- XMPP package structures have 5 different attributes.
  - to   attribute is JID, which identifies the recipient of its respective package.
  - from attribute is JID which identifies who sent the corresponding packet. If a client receives a packet that does not have a attribute from it, it accepts that the package came from the server to which the client is connected. However, if the domain from the incoming XMPP packet is invalid or unspecified, then the server will return the sa invalid from inde (<invalid-from />) response to the client.
  - id attribute is used selectively, except for the info / query (iq) package, to assist in the mapping / identification of response packets that occur in XMPP packets.
  - type attribute refers to information about the content or purpose of the message (<message>), status (<presence>), and information / query (<iq>).
  - (xml:lang) attribute refers to the default language for XML characters. The XML language definition can be changed by child tags. When this attribute is never defined, the default language definition of the server is used.

References

- A. Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, "*Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications*", IEEE Communication Survey&Tutorials, vol. 17 (4), 2347-2376 ,2015.
- L. Atzori, A. Iera, G. Morabito, "The Internet of Things: A Survey", Computer Networks, vol. 54, 2787-2805, 2010.
- O. Vermesan, P. Friess (Editors), "Internet of Things – From Research and Innovation to Market Deployment", River Publishers, 2014.
- R.T. Fielding, "Architectural Styles and the Design of Network-Based Software Architectures". PhD Thesis, University of California; 2000. https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm
- P. L. Gorski, L. L. Iacono, H. V. Nguyen, D. B. Torkian, "Service Security Revisited", IEEE International Conference on Services Computing, 464-471, 2014.
- A. N. Çiçek, "RESTful Web Servisleri ile E-Sağlık Sistemleri Gerçekleştirimi", Yüksek Lisans Tezi, TOBB Ekonomi ve Teknoloji Üniversitesi, Fen Bilimleri Enst. 2009.
- Halil Arslan, Doktora Tezi, SAÜ Fen Bilimleri Enstitüsü, 2016