FFV 2012

MOBILE SERVER PLATFORM

Architectures and Protocols for Future M2M Ecosystems

Fahad Aijaz

RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

KOMMUNIKATIONSNETZE COMMUNICATION NETWORKS ComNets

ComNets Research Group
RWTH Aachen University

March 16, 2012

Outline of Talk

- Research Area in a Nutshell
- Motivation and Research Gap
- The Mobile Server Platform
  - The Mobile Web Server Layer
    - Architecture, Payload and Performance Aspects
    - Multimedia Extensions and Topologies
  - The SLA Framework Layer
    - Components and Life Cycles
- Performance Analysis
- The Software Application Layer
  - MEDICARE – A Context-Aware Health Care Application
  - Social Network in Pocket (SNiP)
- Conclusion

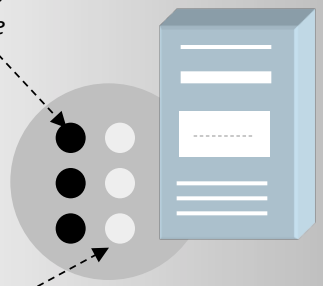
Research Area in a Nutshell

Today's Internet and the WWW

- → High-tech Web Servers.
- → Hosts Web Service and Resources.
- → Transparent Access to the Clients.
- → Neutral towards diverse clients.

TRANSPARENT ACCESS

WEB SERVICES
- *Specialized functions*
- *Internal process*
- *Access interface*

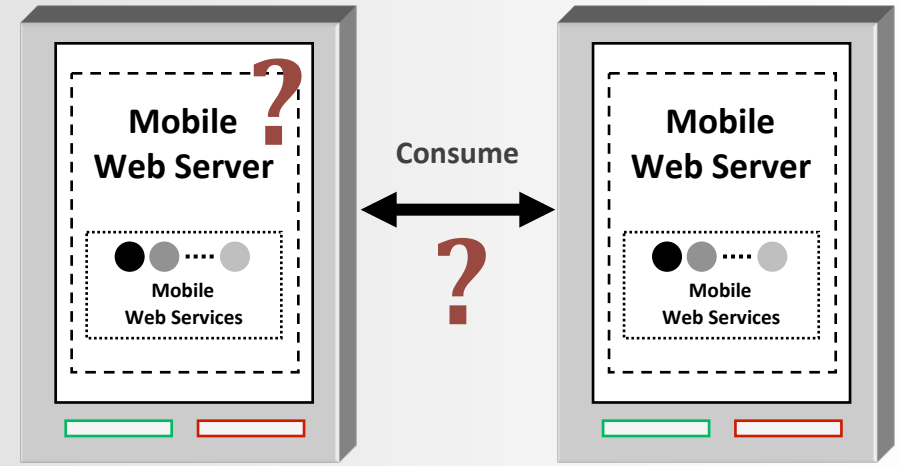


- *Private Data*
- *Multimedia*
- *Websites*

RESOURCES

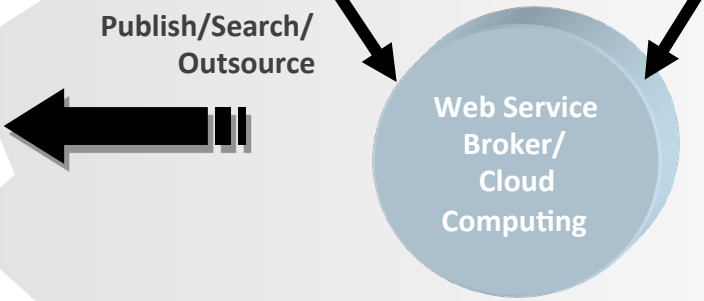
Web Server

Internet of Things = M2M



Publish/Search/ Outsource

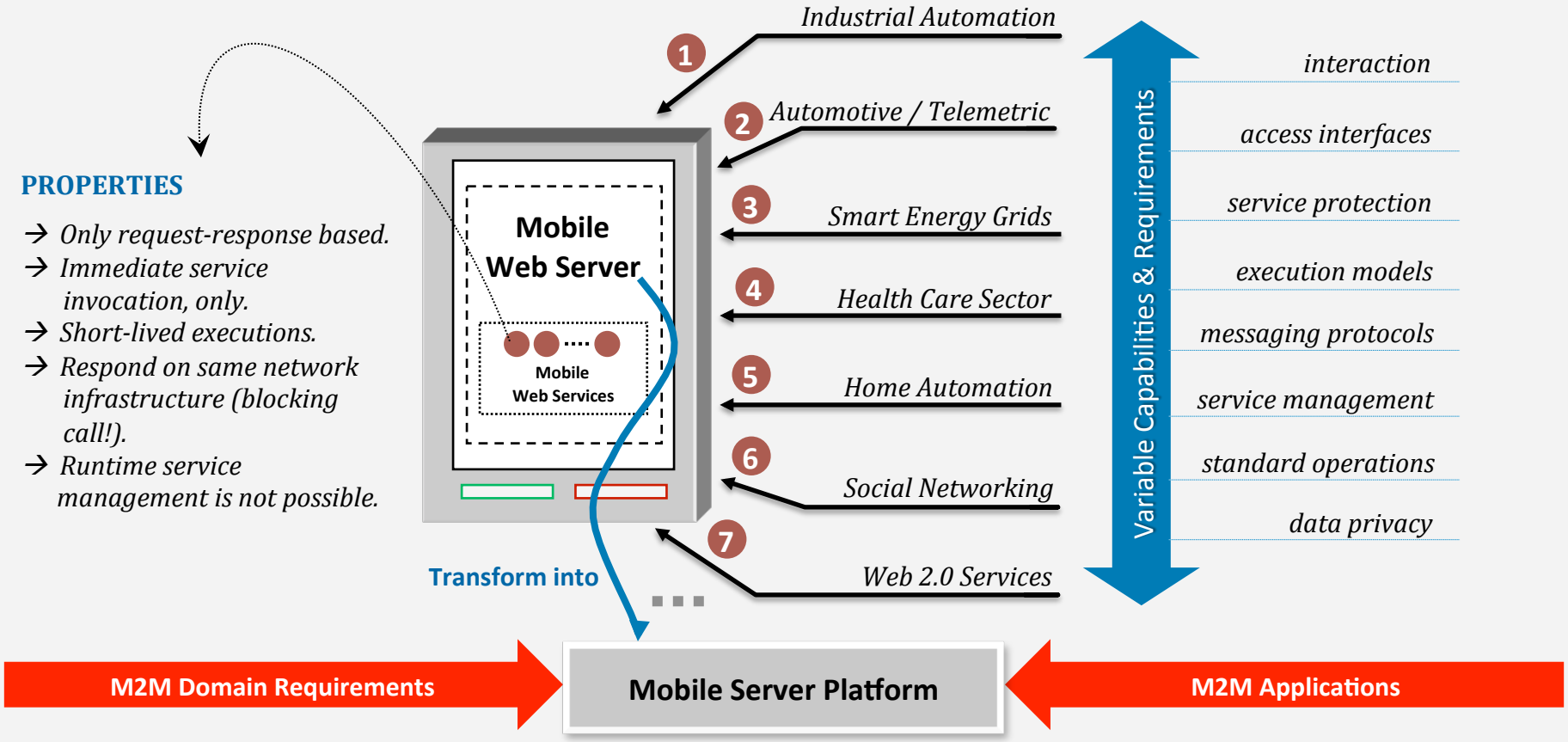
Publish/Search/ Outsource



M2M Services

Motivation and Research Gap

Mobile devices are not only mobile phones!



Provides a general middleware architecture and protocol stack for rapid development of M2M applications

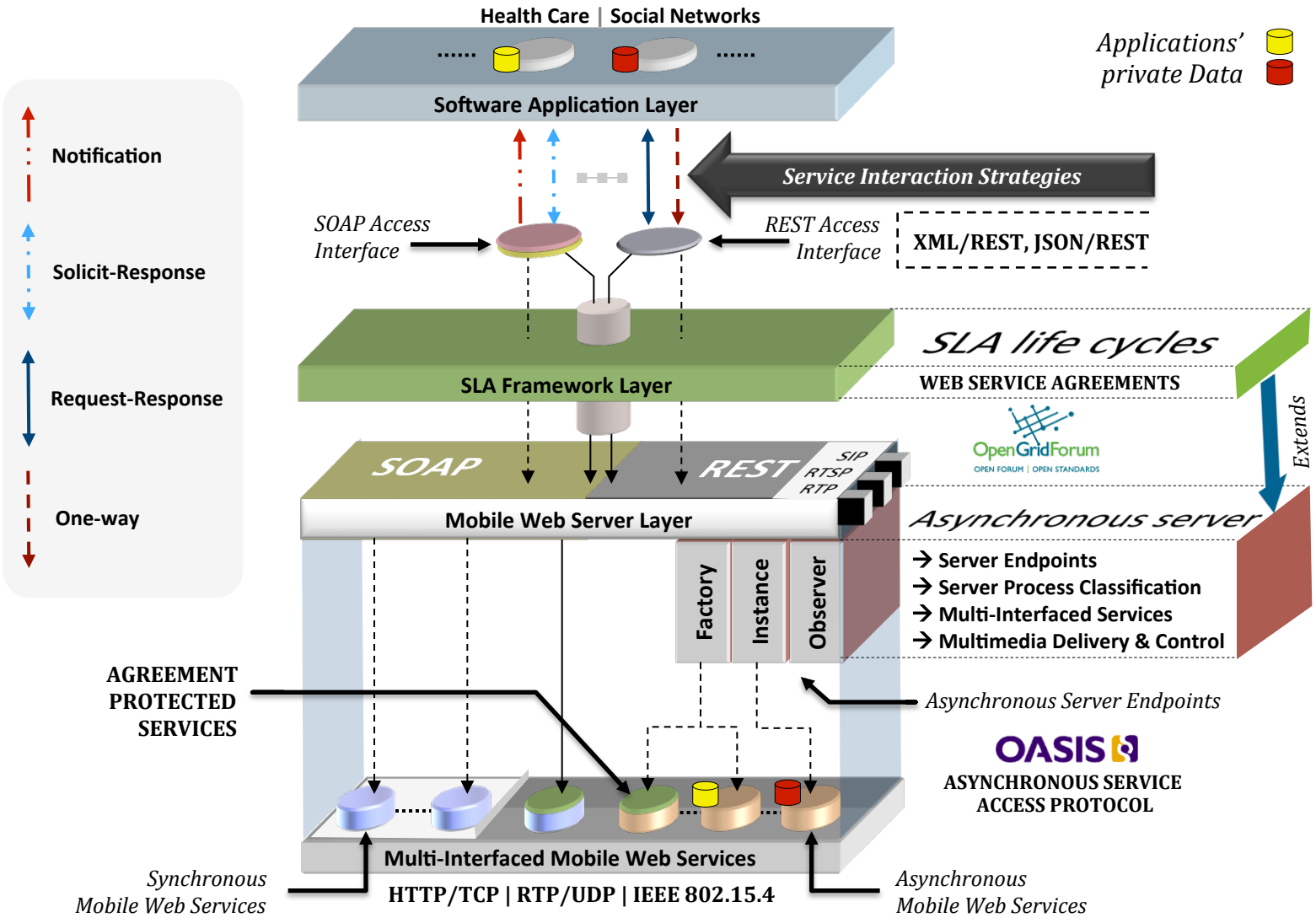
This research gap has been confirmed through literature!

The Mobile Server Platform
(Conceptual Overview)

The MSP provides a set of strategically connected architectures

Offers a Web Services provisioning solution for M2M apps

The platforms architecture is categorized into logical layers



The Mobile Web Server Layer

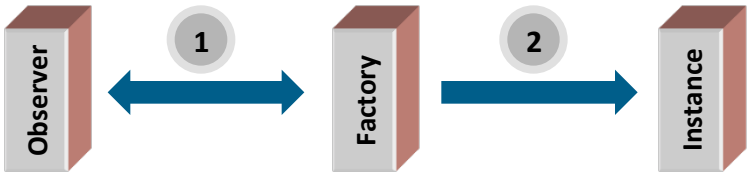
(Classification of Server Processes)

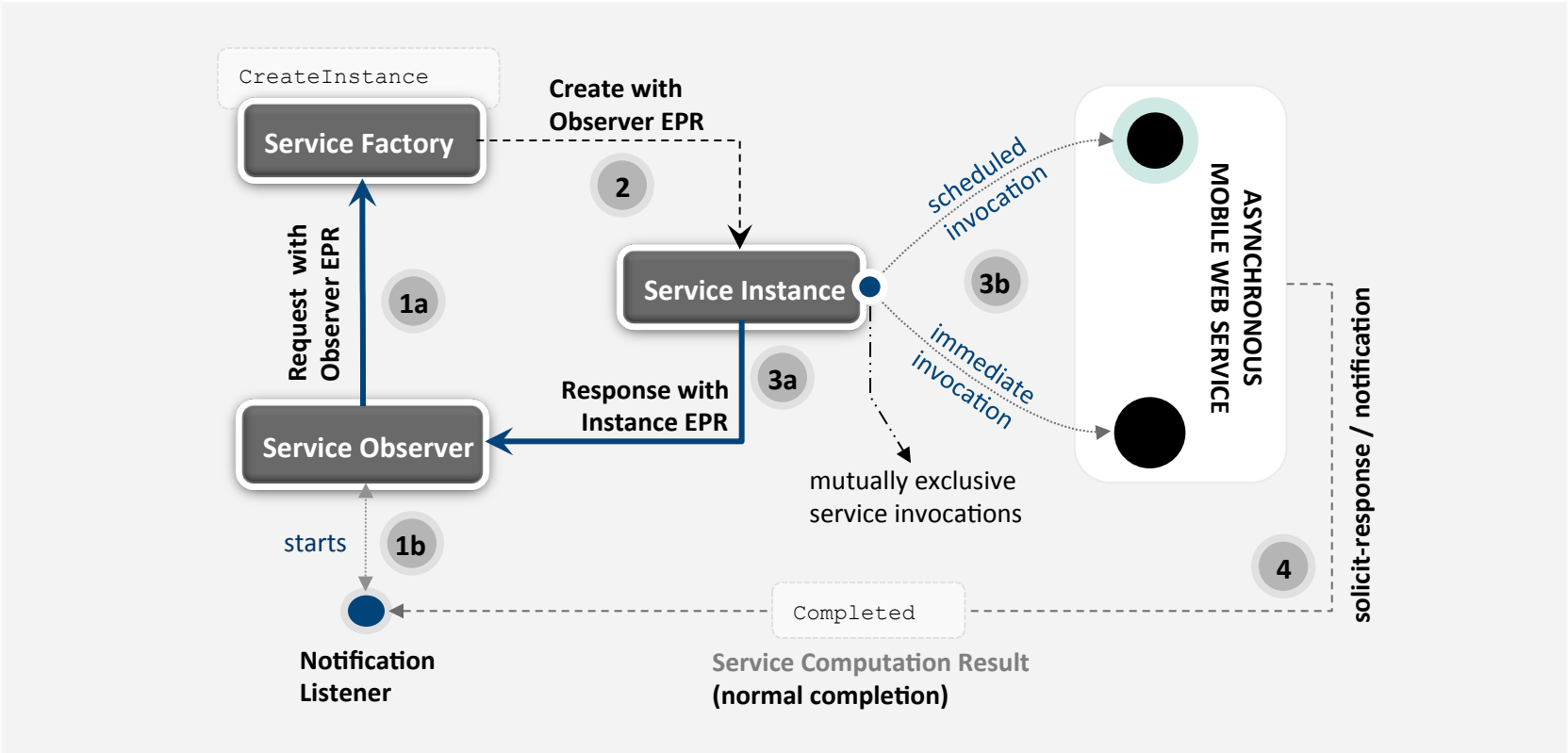
Service Management Process

Service Creation Process

Mandatory process to consume asynchronous Mobile Web services

Involved Server Endpoints





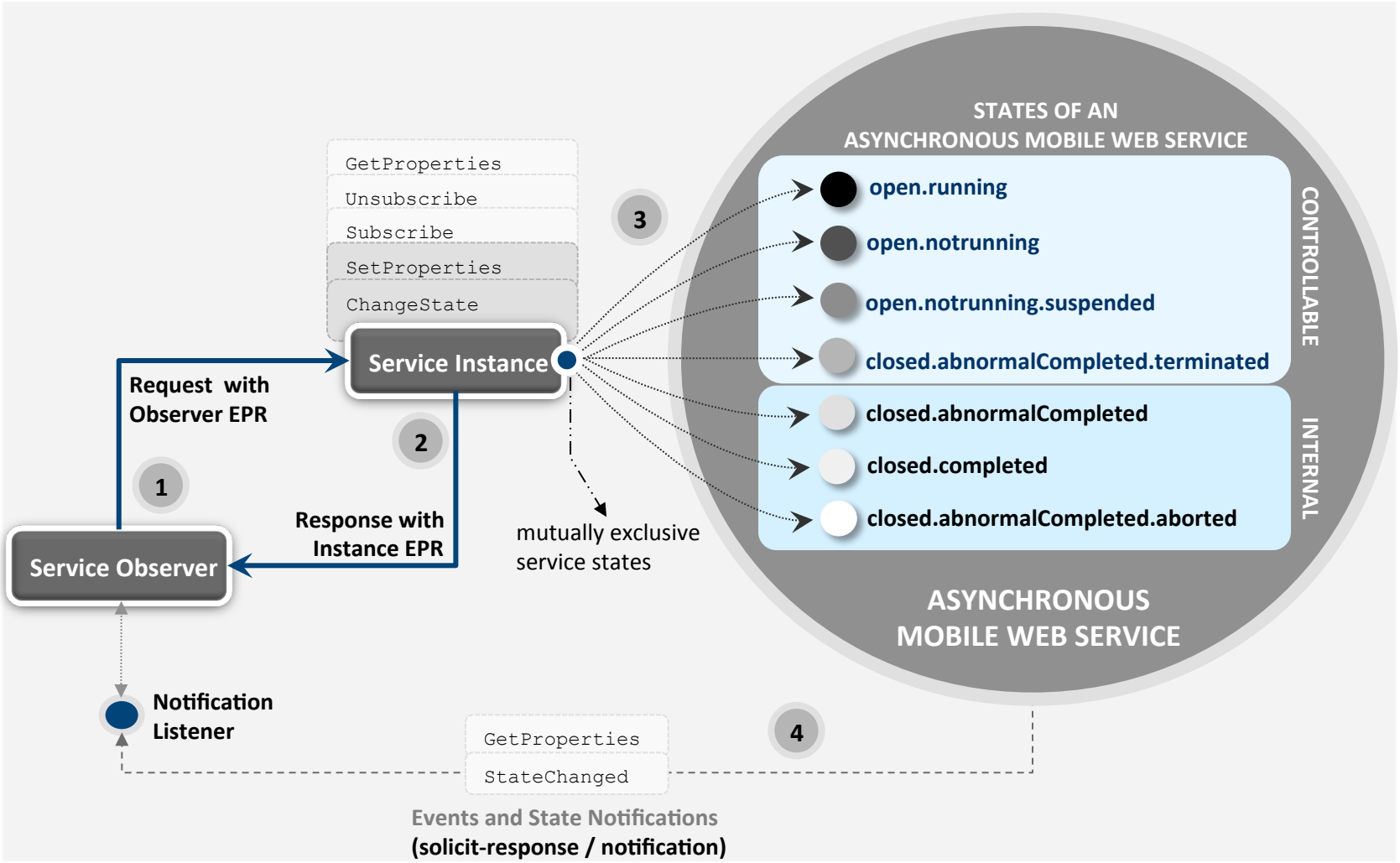
The Mobile Web Server Layer

(Classification of Server Processes)

Service Creation Process

Service Managemenr Process

Only possible subsequent to the Service Creation Process



The Mobile Web Server Layer

(SaaR Design Approach)

Service Provider specifies the mapping semanctics for each method

REpresentational

Network resources
Example: HTML, XML, JPG, GIF…

State

Every resource changes the client's state

Transfer

Resources are transferred using HTTP

Access Interface

Synchronous Access URL

SYNCHRONOUS URL ELEMENTS

HTTP://IP:PORT/SERVICE/METHOD

mandatory — optional

Service Logic

CREATE ST
READ T
UPDATE T
DELETE LETE

Asynchronous Access URL

ASYNCHRONOUS URL ELEMENTS

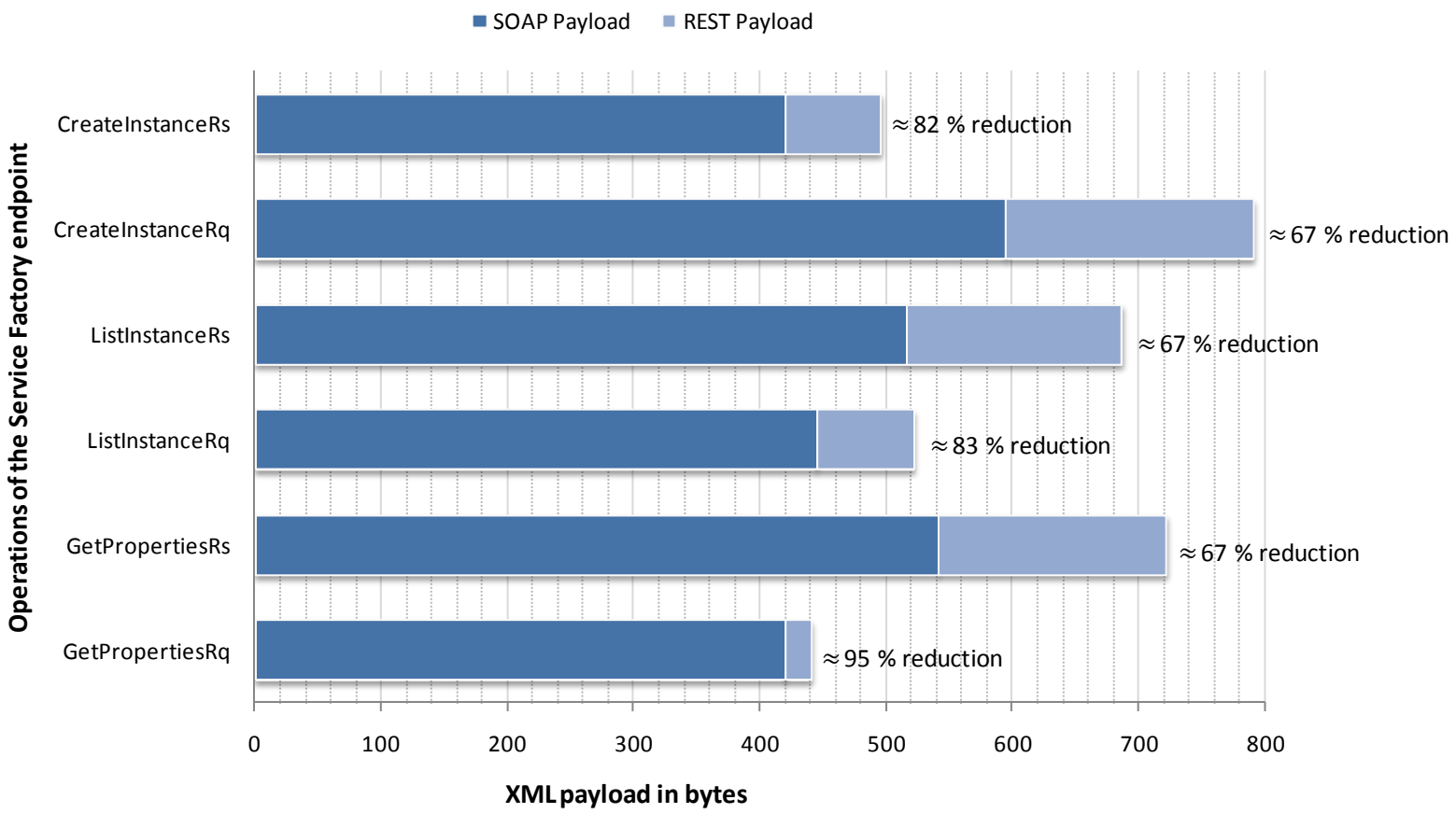
HTTP://IP:PORT/SERVICE/METHOD/STRATEGY/ENDPOINT/OPERATION

mandatory — optional — mandatory

The Mobile Web Server Layer

(Payload Optimization)

XML payload comparison of the operations offered by Service Factory Endpoint



JSON/REST results in much optimized payload than the XML/REST

The Mobile Web Server Layer

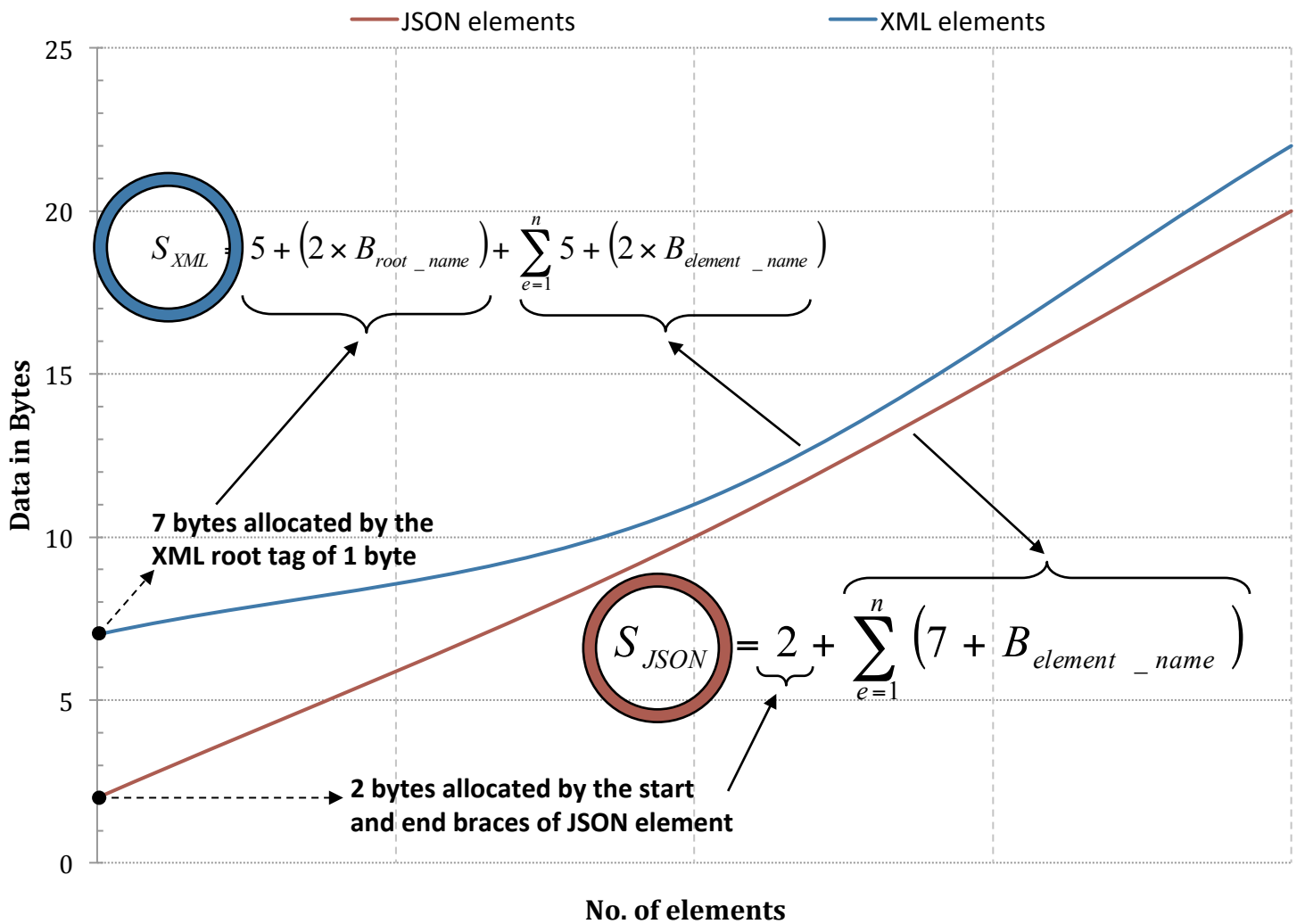
(Basic Analysis of XML and JSON Payloads)

XML Encoding

```
<R>
<123> </123>
<123> </123>
<123> </123>
...
</R>
```

JSON Encoding

```
{
"123" : null
"123" : null
"123" : null
...
}
```

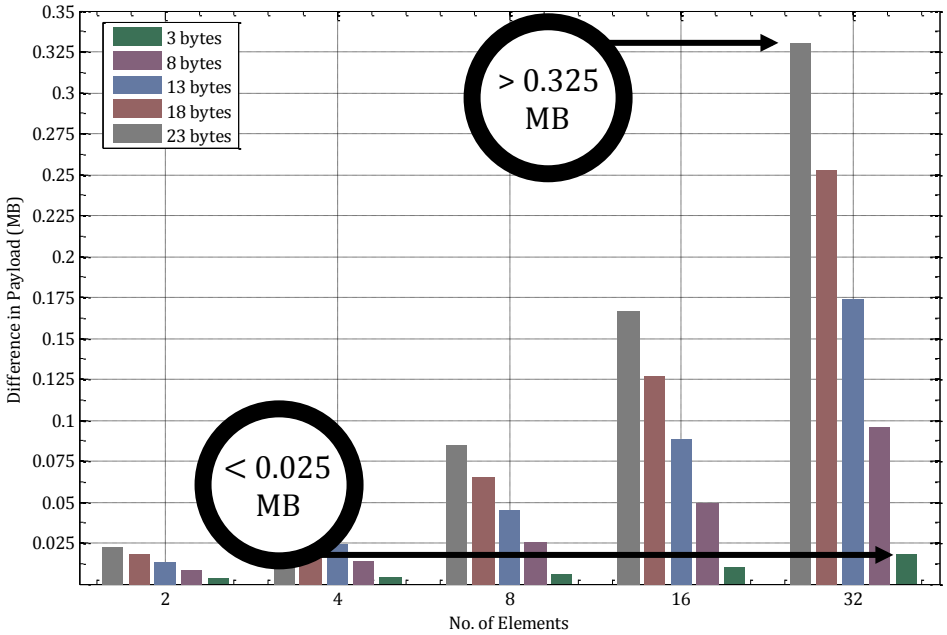


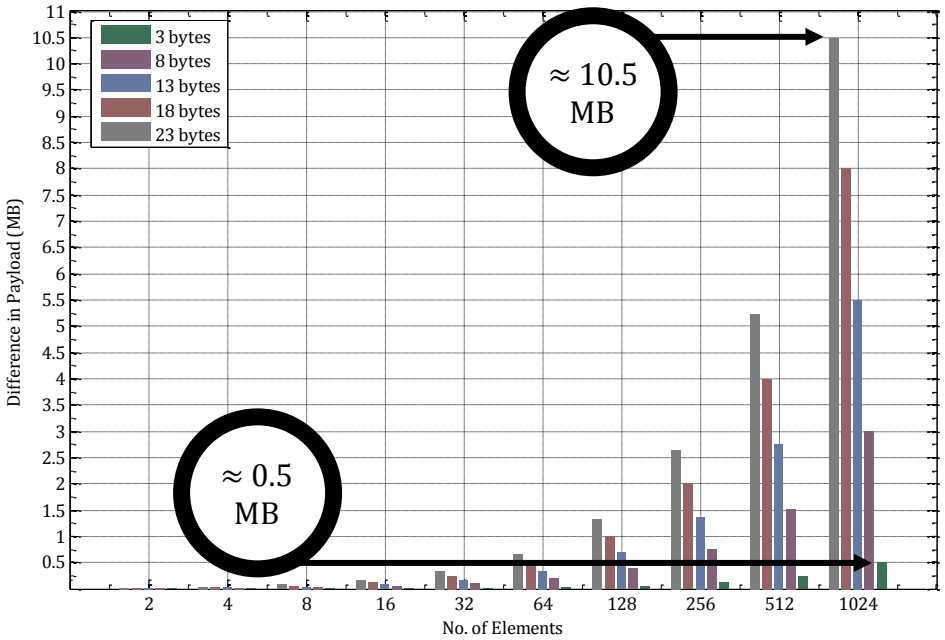
The Mobile Web Server Layer

(Basic Analysis of XML and JSON Payloads)

Difference trend with 2^5 elements	Difference trend with 2^{10} elements

$$S_{XML} - S_{JSON}$$





JSON results in less data transmission as number of elements increase

Lengthy element names result in much rapid increase is payload

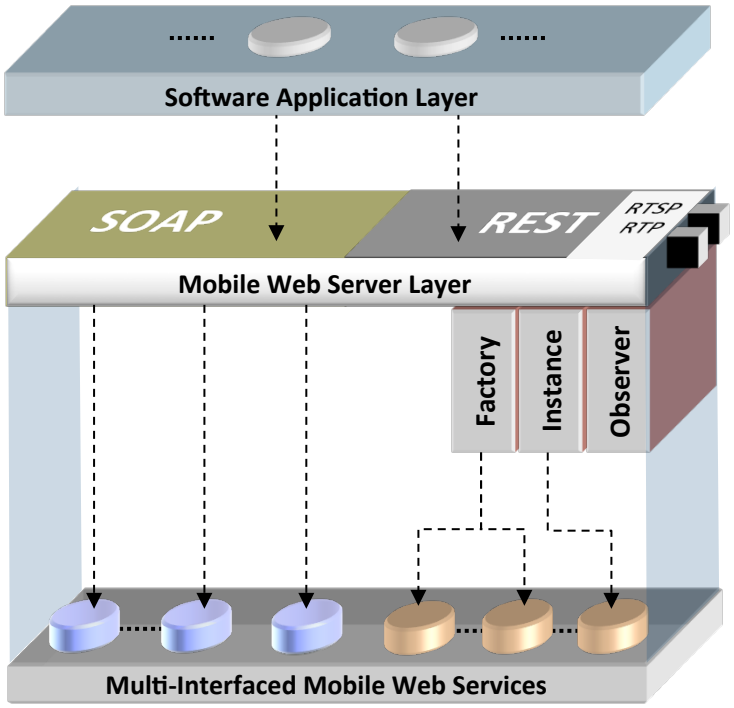
JSON/REST for M2M terminals → Less computation requirements

The Mobile Web Server Layer

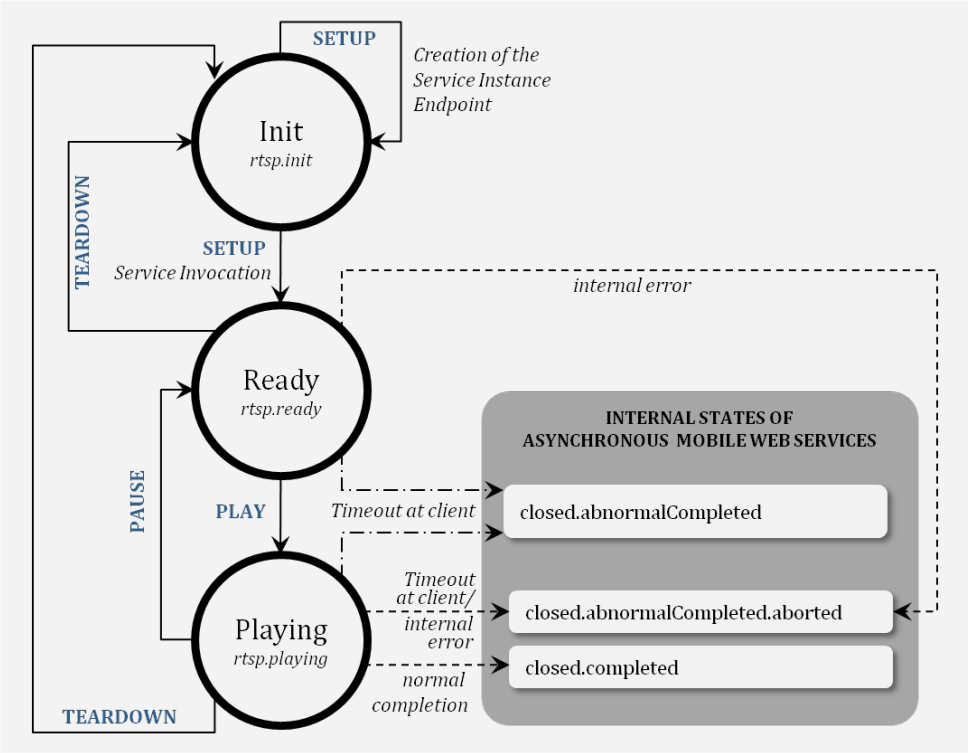
(Multimedia Extensions and Topologies)

Offers a Web Service provisioning solution through M2M terminals

Integration of RTSP and RTP



Extended RTSP States in Asynchronous Server



Delivery Resource → Asynchronous Services

Control Resource → Synchronous and Asynchronous Services

Multimedia Delivery Strategy → RTP/UDP

Multimedia Control Strategy → RTSP/TCP

The RTSP SETUP, PLAY, PAUSE and TEARDOWN are exposed as states of asynchronous services

The RTSP OPTIONS and DESCRIBE methods are exposed as synchronous mobile Web Services

The Mobile Web Server Layer

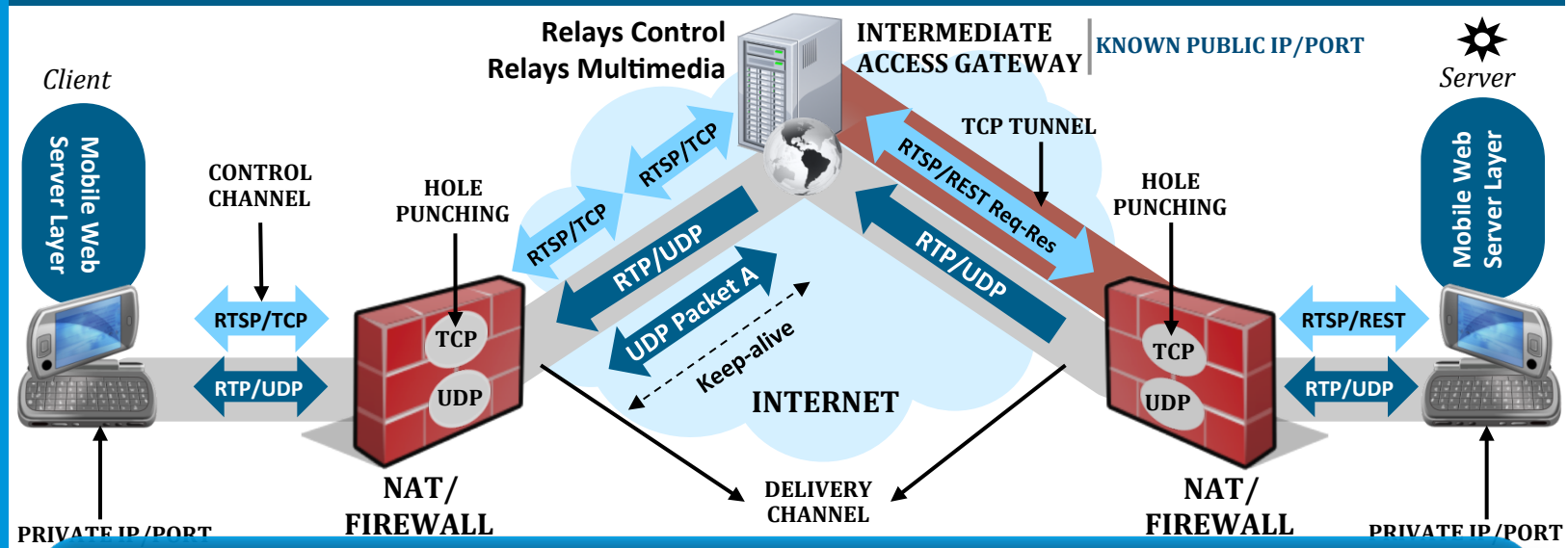
(Multimedia Extensions and Topologies)

Multimedia Delivery Strategy → RTP/UDP

Multimedia Control Strategy → RTSP/TCP

Synchronous and Asynchronous Servers work together

Relayed Delivery and Relayed Control



Works with every network where TCP/UDP are enabled!

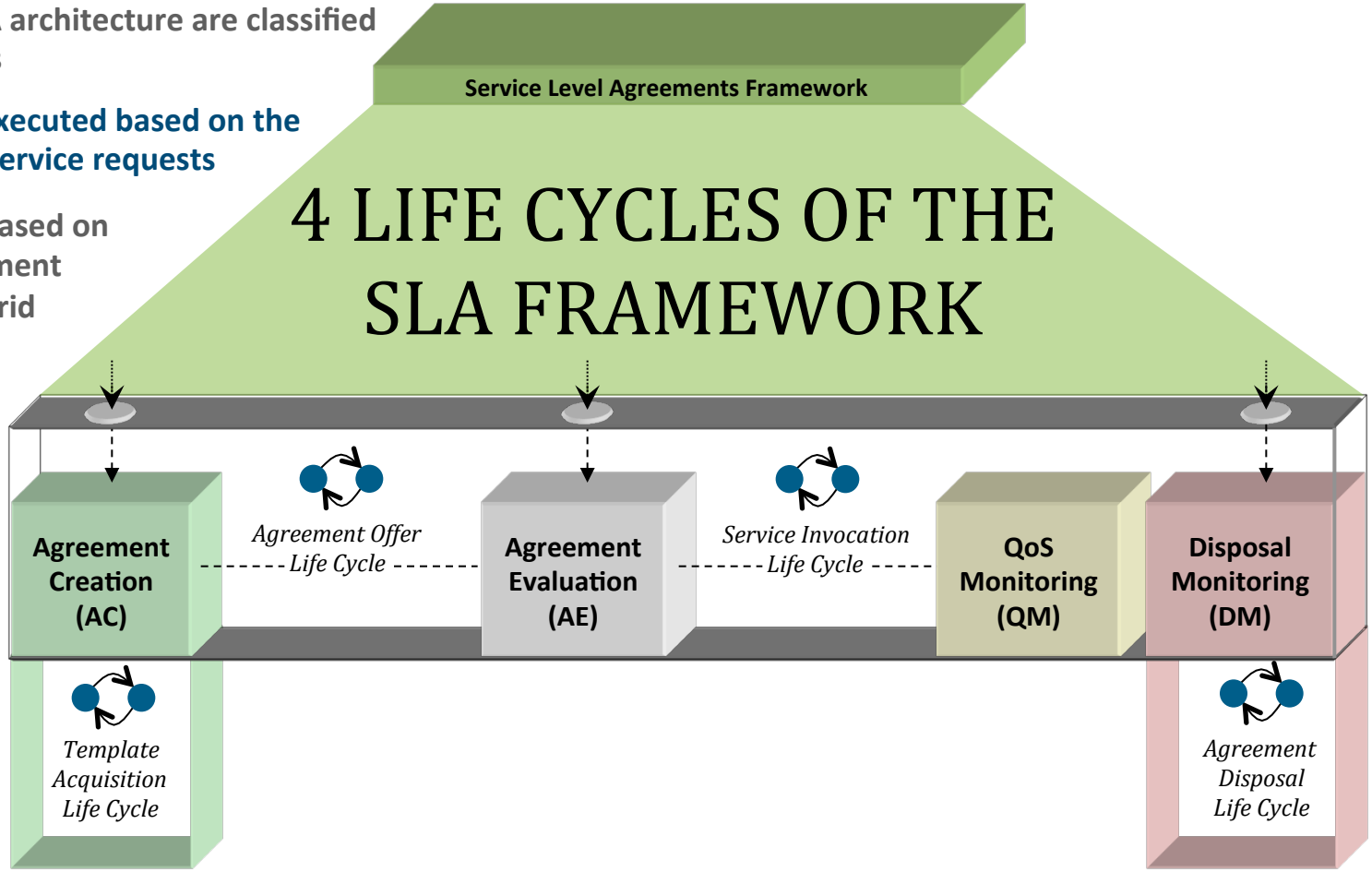
1. Server establishes a permanent TCP tunnel with the IAG → TCP hole punching
2. The client sends RTSP SETUP to the IAG → IAG relays it to server → SCP response
3. Only client sends UDP to IAG → IAG is not introducer gateway → sends keep-alive to client
4. Client sends RTSP PLAY to IAG → IAG relays it to server → Service Management Process
5. Server indirectly delivers media over the RTP/UDP channel that is relayed by the IAG
6. Client may send RTSP PAUSE, TEARDOWN via IAG → Service Management Process

The SLA Framework Layer

(Conceptual Overview)

Extends the Mobile Web Server Layer

- → The functions of the SLA architecture are classified into 4 distinct life cycles
- → **The SLA life cycles are executed based on the incoming mobile Web Service requests**
- → The SLA negotiation is based on the Web Service Agreement standard of the Open Grid Forum
- → **The standard is optimized to define the SLA messaging for mobile nodes**
- → The SLA framework is compatible with the REST and SOAP access interfaces
- → **Protects both, the synchronous and asynchronous services**

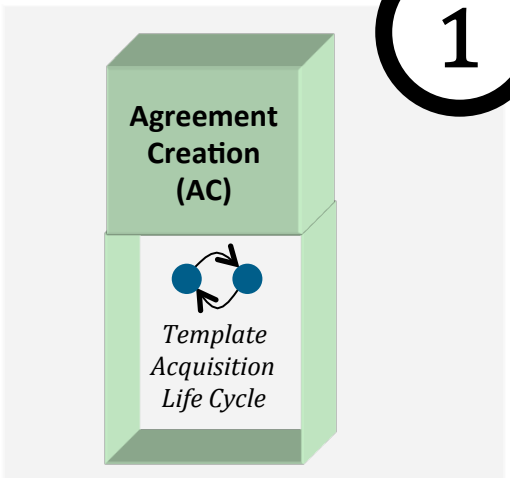


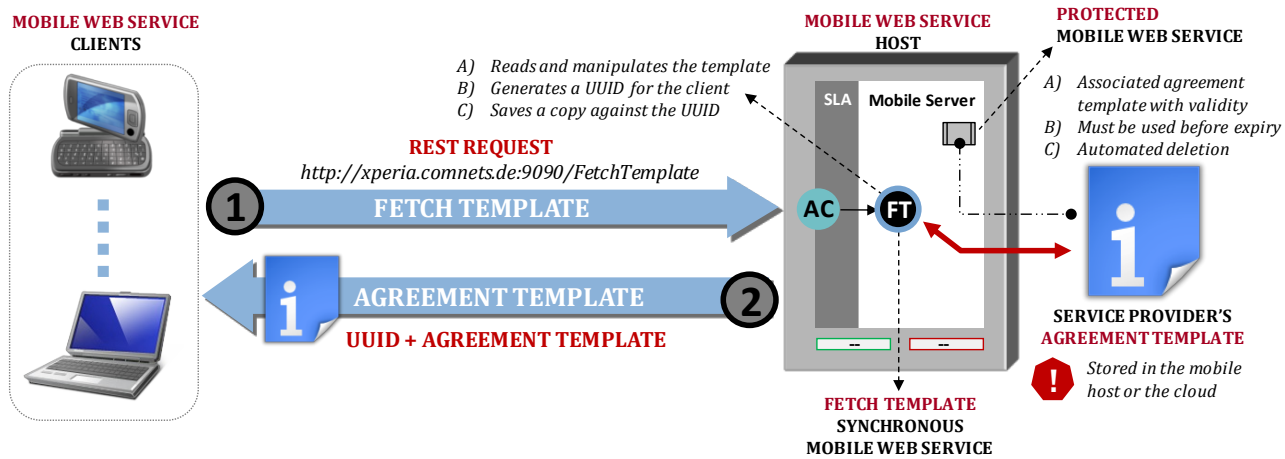
The life cycles utilize the properties of the Synchronous and Asynchronous Servers

The SLA Framework Layer

(Life Cycles)

Template Aquisition Life Cycle

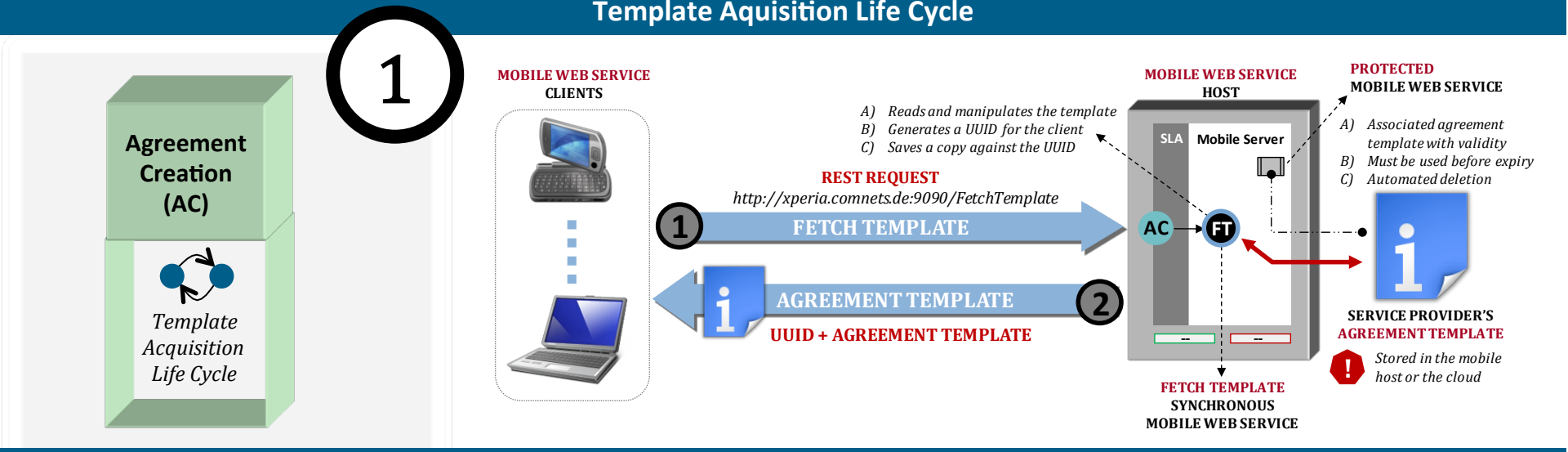




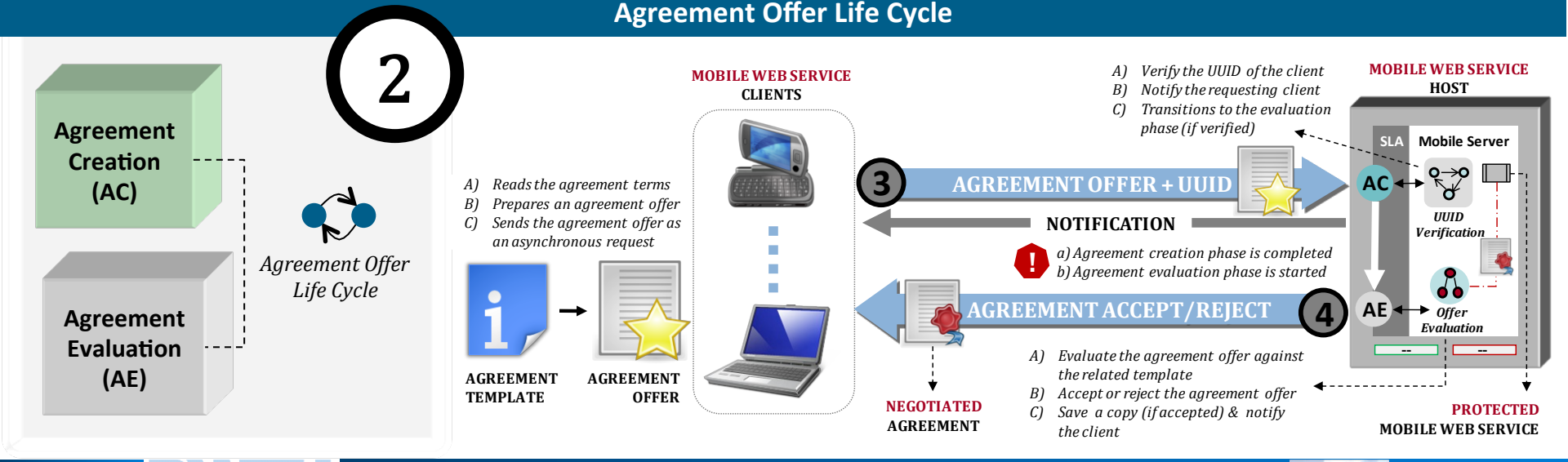
The SLA Framework Layer

(Life Cycles)

Template Aquisition Life Cycle



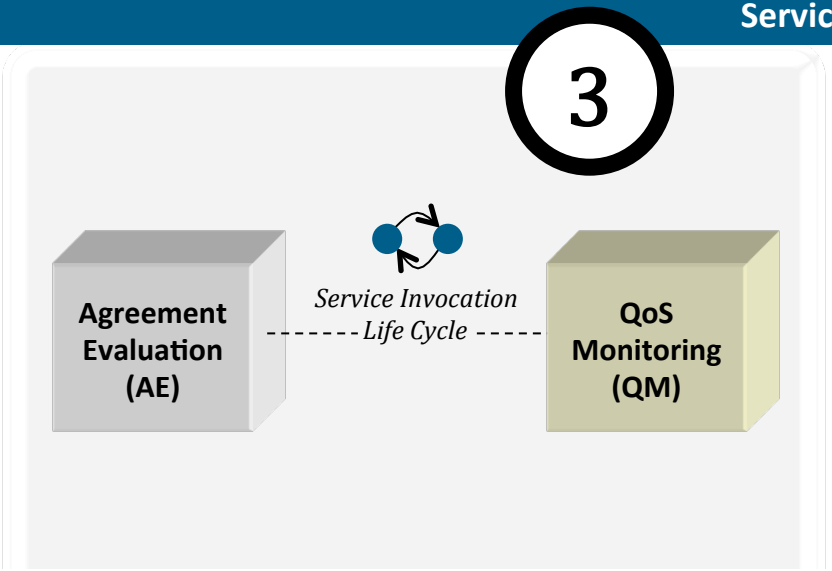
Agreement Offer Life Cycle

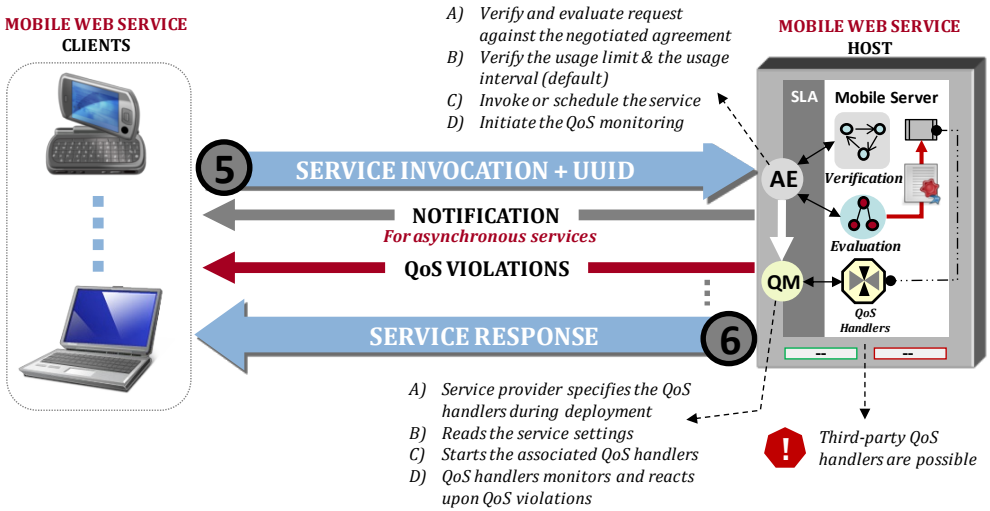


The SLA Framework Layer

(Life Cycles)

Service Invocation Life Cycle

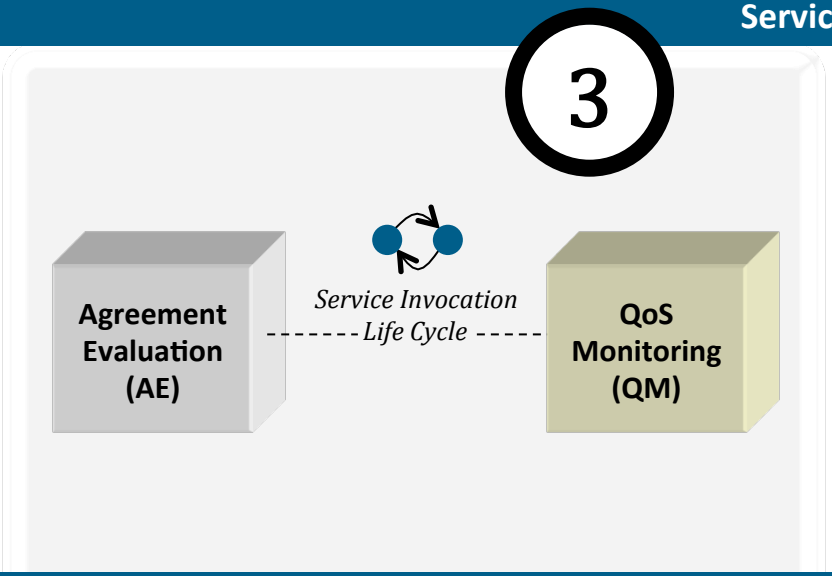


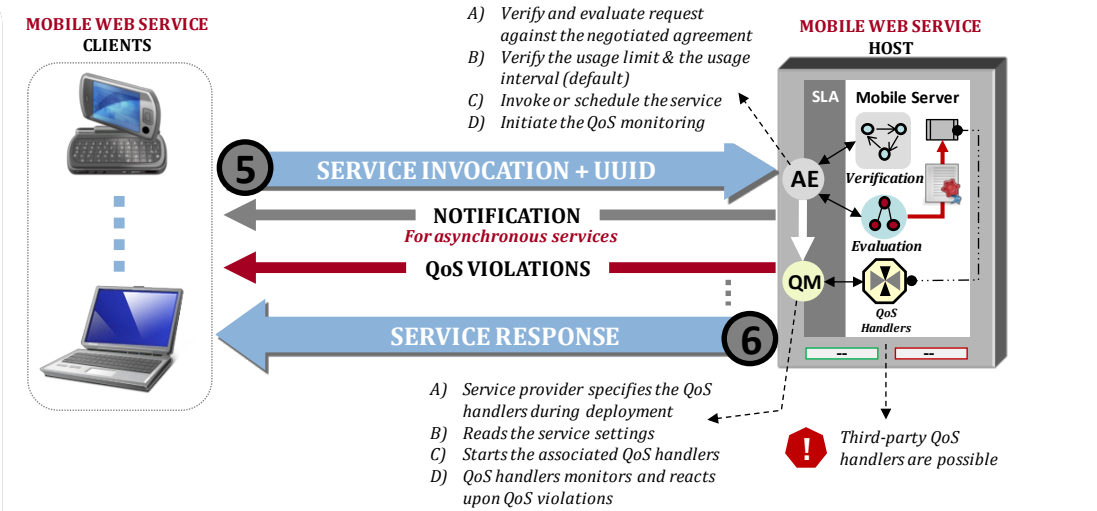


The SLA Framework Layer

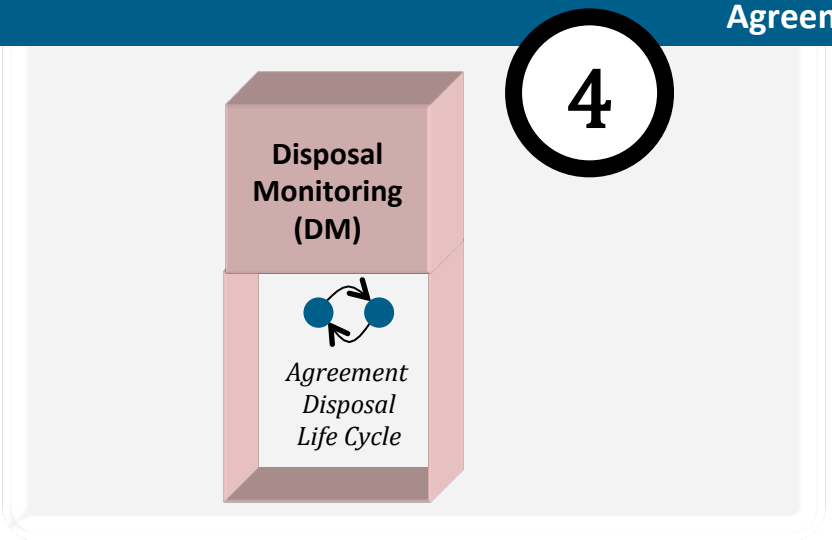
(Life Cycles)

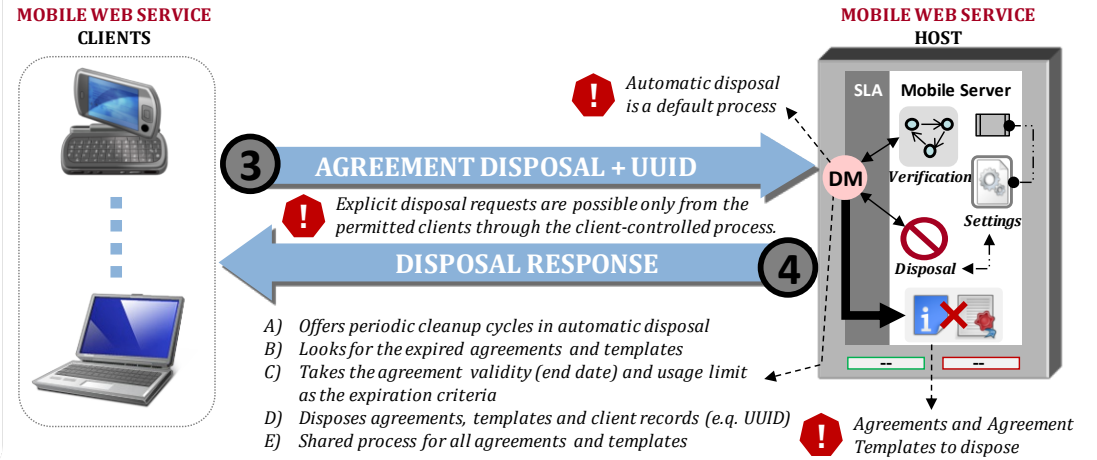
Service Invocation Life Cycle





Agreement Disposal Life Cycle

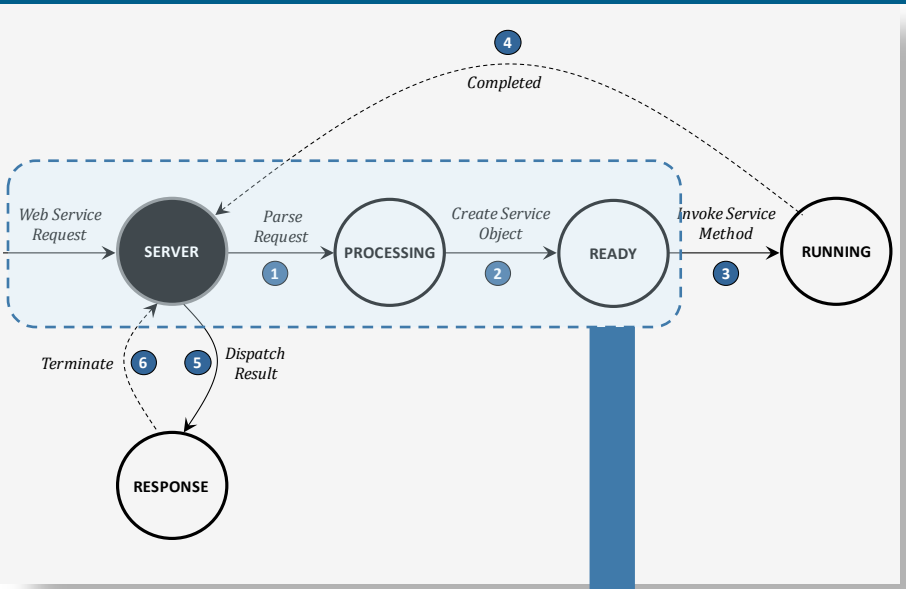




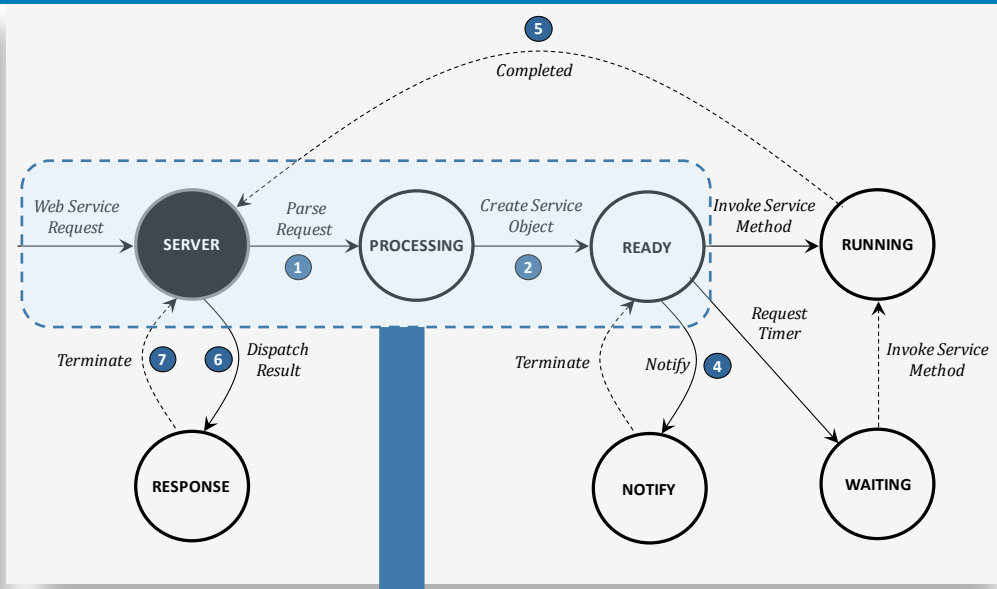
Performance Analysis

(Service Creation Process)

Synchronous Server

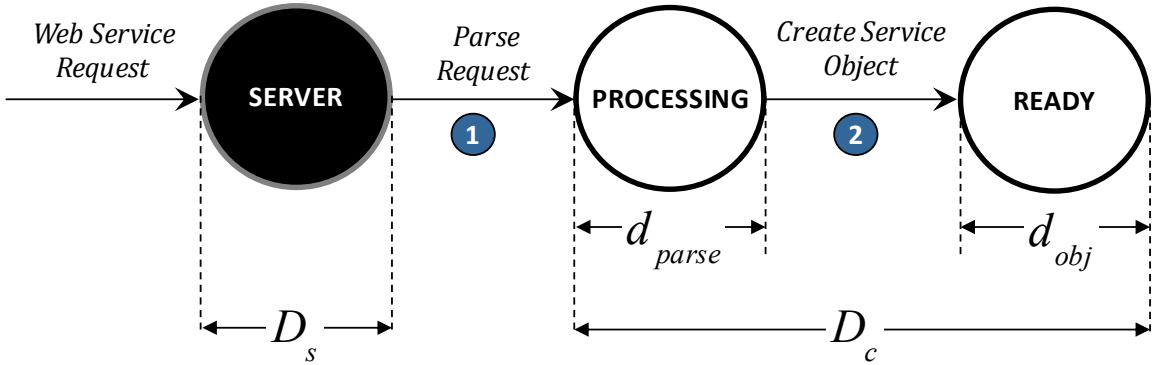


Asynchronous Server



! *Until the service object is created*

! *Until the Service Instance endpoint is created*



Performance Analysis

(Service Creation Process)

Asynchronous SCP with SOAP

```
<SOAP-ENV:Envelope
  ...
  xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <SOAP-ENV:Header>
    <wsa:To> http://xperia.comnets.de </wsa:To>
    <wsa:Action> soaprpc</wsa:Action>
    <wsa:MessageId>ffb471ec-194-1cb37664-125c53c2</wsa:MessageId>
    <wsa:ReplyTo>
      <wsa:Address> http://schemas.xmlsoap.org/ws/.../role/anonymous </wsa:Address>
    </wsa:ReplyTo>
   <as:FactoryEPR>http://xperia.comnets.de/factory</as:FactoryEPR>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body SOAP-ENV:encodingStyle=http://schemas.xmlsoap.org/soap/encoding/>
```

```
    <GPSLocation>
      <ServiceType>Asynchronous</ServiceType>
      <createInstanceRq xmlns="" xsi:type=":createInstanceRq">
            <StartImmediately xsi:type="xsd:boolean">true</StartImmediately>
            <ObserverEPR xsi:type="xsd:string">
                http://tattoo.comnets.de/3466b6a-0df-5fe9ad
            </ObserverEPR>
            <Name xsi:type="xsd:string">GPS Location</Name>
            <Subject xsi:type="xsd:string"> Create Service Instance </Subject>
            <Description xsi:type="xsd:string">
                Provides GPS coordinates of the host mobile
           </Description>
```

```
            <ContextData xsi:type="xsd:ContextData">
                <!– user-defined input XML elements -->
            </ContextData>
```

```
      </createInstanceRq>
    </GPSLocation>
```

```
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

e_{in} e_{st} $e_{k=asyn}$

$$d_{soap}^{asyn}(X) = D_S + \underbrace{\underbrace{(e_{in} + e_{st} + e_{asyn})}_{=d_{parse}^{asyn}} + d_{obj}}_{=D_c^{asyn}}$$

Synchronous SCP with SOAP

```
<SOAP-ENV:Envelope
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xmlns:xsd=http://www.w3.org/2001/XMLSchema
  xmlns:SOAP-ENC=http://schemas.xmlsoap.org/soap/encoding/
  xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/>
  <SOAP-ENV:Body SOAP-ENV:encodingStyle=http://schemas.xmlsoap.org/soap/encoding/>
```

```
    <echoService xmlns="urn:Services" id="o0" SOAP-ENC:root="1">
      <ServiceType xmlns="" xsi:type="xsd:string">Synchronous</ServiceType>
```

```
      <ContextData xmlns="" xsi:type="xsd:ContextData">
          <!-- user-defined input XML elements -->
      </ContextData>
```

```
    </echoService>
```

```
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

e_{in} e_{st} $e_{k=syn}$

$$d_{soap}^{syn}(X) = D_S + \underbrace{\underbrace{(e_{in} + e_{st} + e_{syn})}_{=d_{parse}^{syn}} + d_{obj}}_{=D_c^{syn}}$$

Performance Analysis

(Service Creation Process)

Asynchronous SCP with XML/REST

Asynchronous Access URL

GPSLocation | Coordinates | Req-Res | Factory | CreateInstance

HTTP : // IP : PORT / SERVICE / METHOD / STRATEGY / ENDPOINT / OPERATION

```
<createInstanceRq xmlns="" xsi:type=":createInstanceRq">
  <StartImmediately xsi:type="xsd:boolean">true</StartImmediately>
  <Name xsi:type="xsd:string">GPS Location</Name>
  <Subject xsi:type="xsd:string"> Create Service Instance </Subject>
  <Description xsi:type="xsd:string"> Provides GPS coordinates of the host mobile</Description>
  <ContextData xsi:type="xsd:ContextData">
        <!– user-defined input XML elements -->
  </ContextData>
</createInstanceRq>
```

e_{in} $e_{k=asyn}$

$$d_{rest}^{asyn}(X) = D_S + \underbrace{\underbrace{(e_{in} + e_{asyn})}_{=d_{parse}^{asyn}} + d_{obj}}_{=D_c^{asyn}}$$

Synchronous SCP with XML/REST

Synchronous Access URL

echoString | (optional)

HTTP : // IP : PORT / SERVICE / METHOD

```
<!-- user-defined input data -->
```

e_{in}

$$d_{rest}^{syn}(X) = D_S + \underbrace{d_{obj}}_{=D_c^{syn}}$$

Mean

$$D_m^k(X) = \frac{\sum_{i=1}^{N} d_m^k(X)_i}{N}$$

Hypothesis

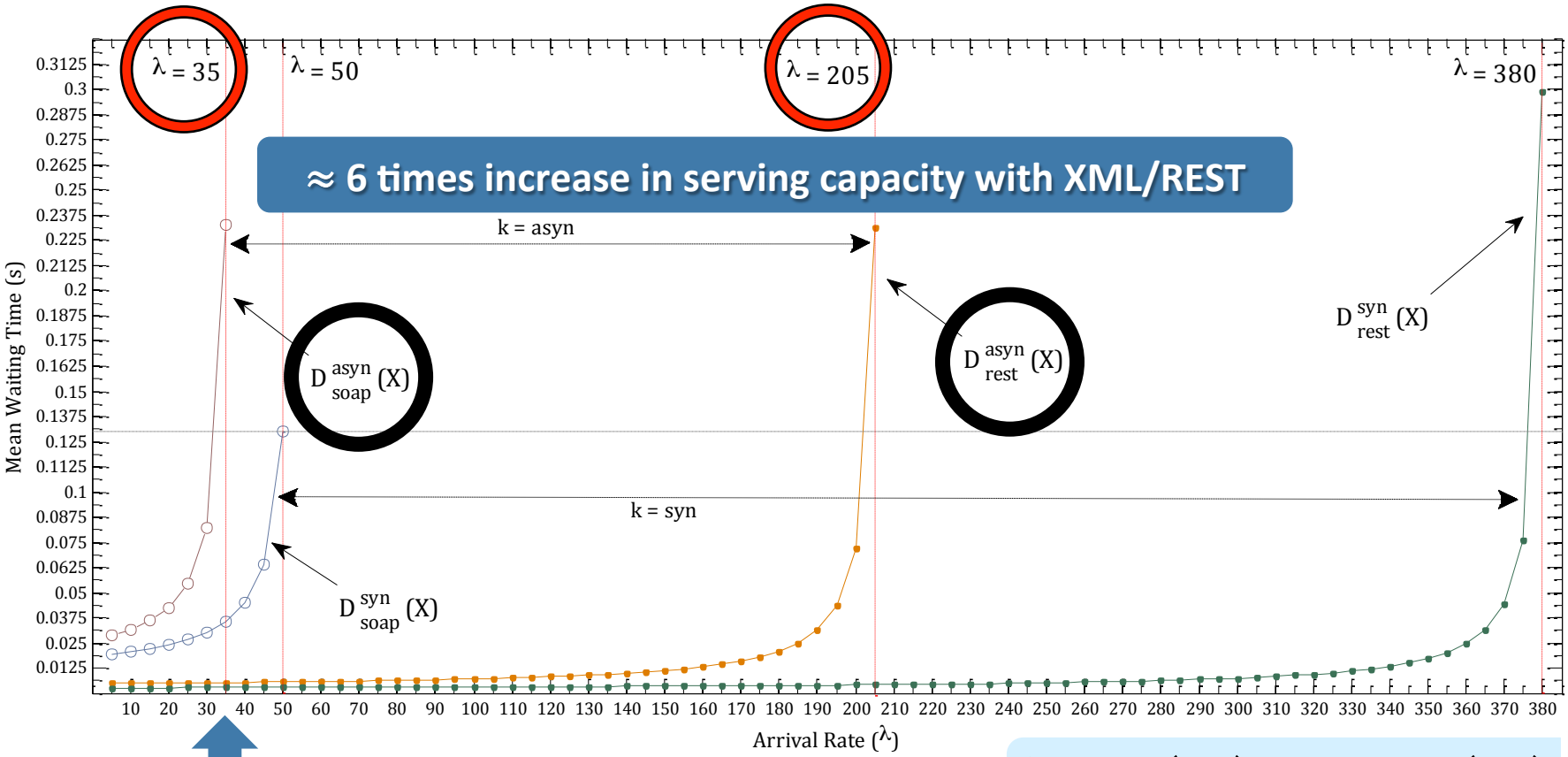
$$D_{rest}^{syn}(X) < D_{soap}^{syn}(X)$$

$$D_{rest}^{asyn}(X) < D_{soap}^{asyn}(X)$$

Performance Analysis

(Service Creation Process)

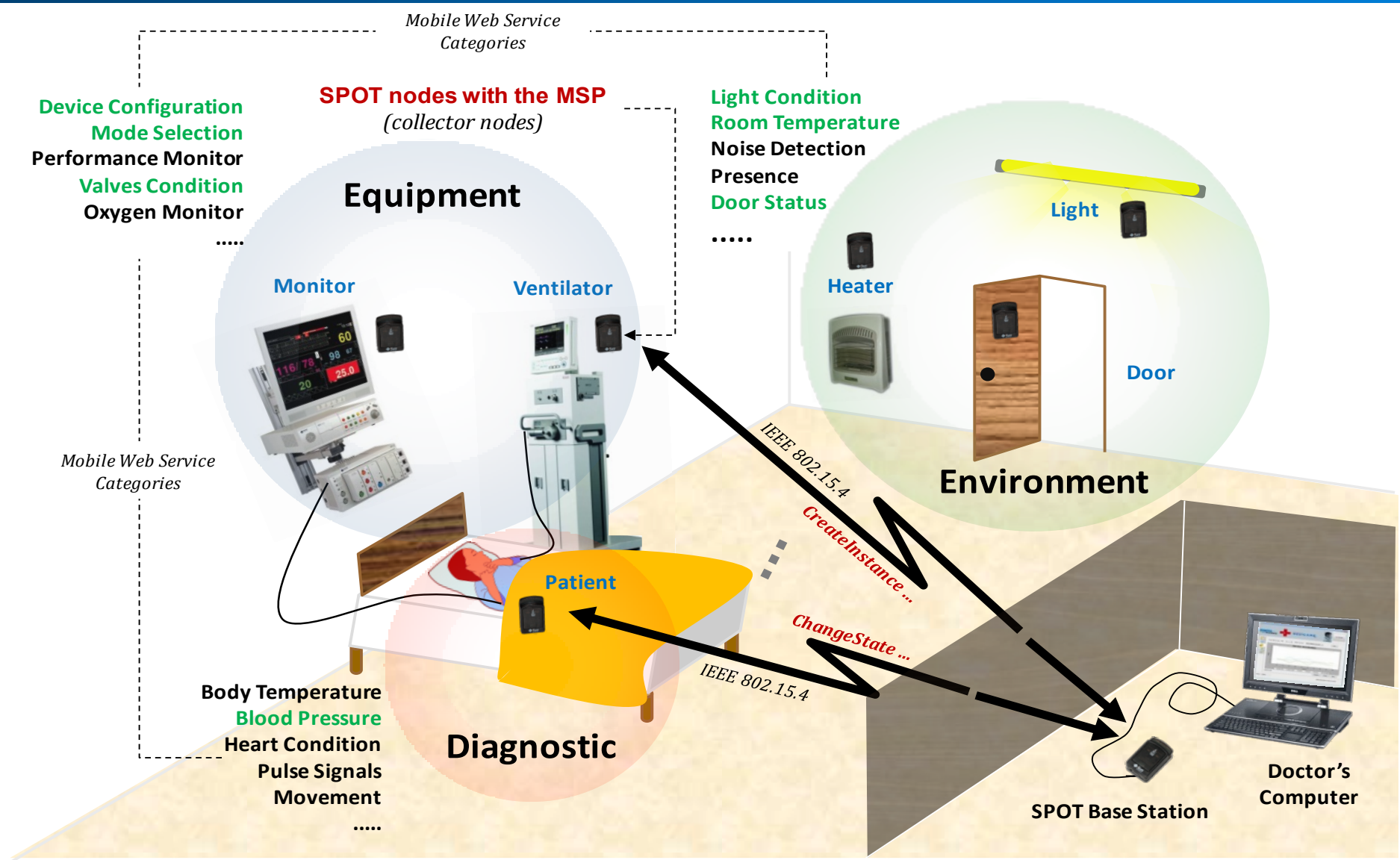
M/D/1 evaluation with increasing arrival rate



$$D_{rest}^{asyn}(X) < D_{soap}^{asyn}(X)$$

Mean waiting time with XML/REST at this arrival rate is about 2% of the one with SOAP

The Software Application Layer
(MEDICARE – A Context-Aware Health Care Application)



The Software Application Layer
(Social Network in Pocket [SNiP] – A privacy-focused social network concept)

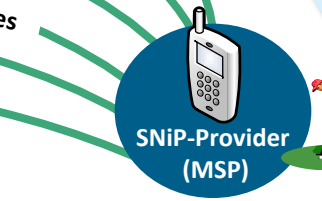
4 To socialize and share her experience, Alice allows her selected friends from her phones address book to access her SNiP services

Service Access Website
SNiP-Consumers

Locate and Track Alice
View Alice's picture gallery
Live Web-to-Mobile Chat
Live Web-to-Mobile Games
Live video streaming
Leave voice messages
... and more

7 She feels satisfied, as SNiP allows her to share her private mobile content without posting it to any third-party server – *PRIVACY!*

3 She has taken pictures, recorded videos and posted comments about many wonderful places during her Italian tours



SNiP-Provider (MSP)

5 Her authorized friends directly access her private data stored on her phone through the *Service Access website* to interact and view the mobile content that Alice has provided for shared use

1 ALICE, *a SNiP user* is on vacation in Italy

2 She has many SNiP services on her mobile phone, such as, *location, gallery, chat, games, video, voice* ...

Disallowed SNiP-Consumers

Access denied

6 Alice has complete control right on her phone to choose who can access her private data stored on the phone

8 Alice feels like her entire Social Network is in her pocket

Conclusion

Contributions of the Research

- **Novel software architecture and protocol stack, called Mobile Server Platform**
  - Offers multiple exceution models, protocols and interaction strategies
  - Solution for data privacy, remote service creation and management on mobile terminals
- **The MSP is developed and evaluated with multiple access interfaces**
  - JSON/REST
    - most suitable for M2M terminals, as the payload increases
  - XML/REST
    - increases 6.0 / 7.6 times the serving capacity of Asynchronous / Synchronous Server, respectively
  - The established hypotheses are verified by M/D/1 evaluations
- **Multiple service execution models leads to new architectural extensions**
  - Multimedia control and delivery through the Mobile Web Server layer
    - Solution to bypass network constraints
  - Contracts-based services through the SLA framework layer → Solution to service protection on terminals
- **The MSP is suitable for M2M terminals and consumer devices**
  - Social Network in Pocket (SNiP) → Solution to data privacy issues
  - The MEDICARE prototype → Solution to context-aware M2M environments

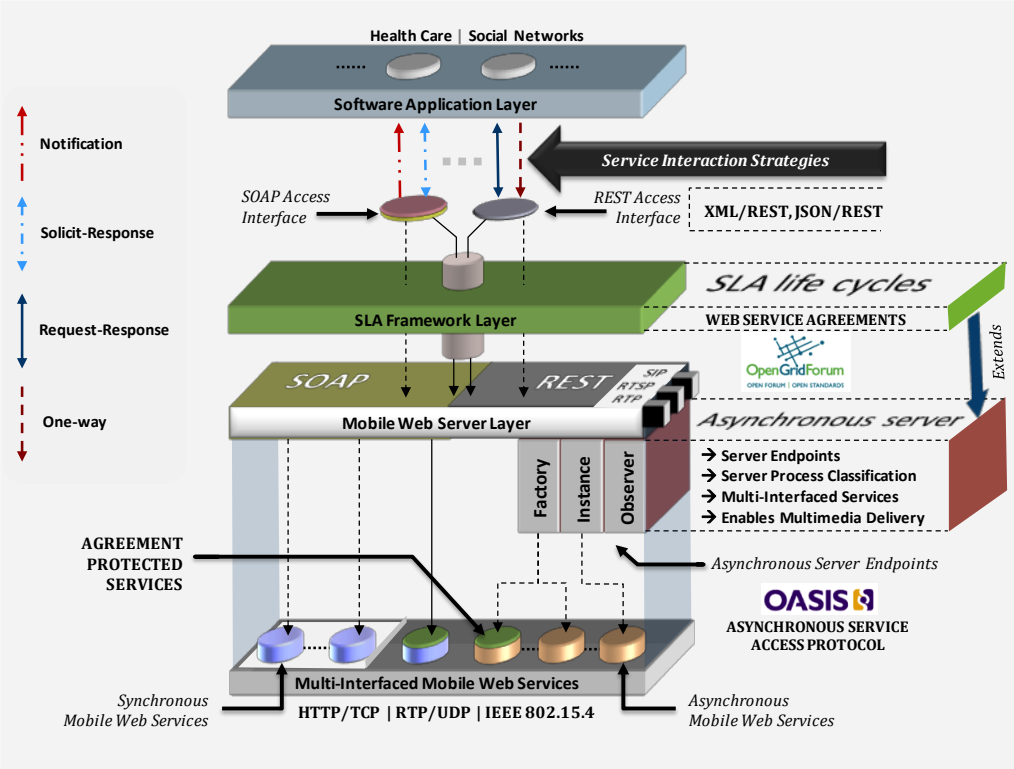
End of Talk

Thank you for your attention!

Fahad Aijaz

fah@comnets.rwth-aachen.de

www.fahadaijaz.com



The Mobile Server Platform

Questions?