



oneM2M Protocol

SeungMyeong JEONG (sm.jeong@keti.re.kr)

Korea Electronics Technology Institute

2020.11.12

The project "International Digital Cooperation - ICT Standardisation" is funded by the European Union



Outline



- Core Protocol
- Protocol Binding



Core Protocol

- XML (Extensible Markup Language) Serialization
 - oneM2M provides XSD (XML Schema Definition) for primitive validation

```
<?xml version="1.0" encoding="UTF-8"?>
<m2m:rqp xmlns:m2m="http://www.oneM2M.org/xml/protocols">
  <op>1</op>
  <to>/mn-cse/home_gateway/light_ae1/light</to>
  <fr>/mn-cse/Clight_ae1</fr>
  <rqi>mncse/24345</rqi>
  <ty>4</ty>
  <rcn>0</rcn>
  <pc>
    <m2m:cin>
      <con>OFF</con>
      <cnf>text/plain:0</cnf>
    </m2m:cin>
  </pc>
</m2m:rqp>
```

- JSON (Javascript Simple Object Notation) Serialization
 - less verbose and lighter than XML

```
{
  "op": "1",
  "to": "/mn-cse/home_gateway/light_ae1/light",
  "fr": "/mn-cse/Clight_ae1",
  "rqi": "mncse/24345",
  "ty": 4,
  "rcn": 0,
  "m2m:cin":
  {
    "con": "OFF",
    "cnf": "text/plains:0"
  }
}
```

- CBOR (Concise Binary Object Representation) Serialization
 - JSON binary compression for lightweight payload

```
{"op":1,"to":"//example.net/mncse1234","rqi":"A1000",  
"rcn":7,"pc":{"m2m:ae":{"rn":"SmartHomeApplication", "api":"Na56", "apn":"app1234"}}, "ty":2}
```

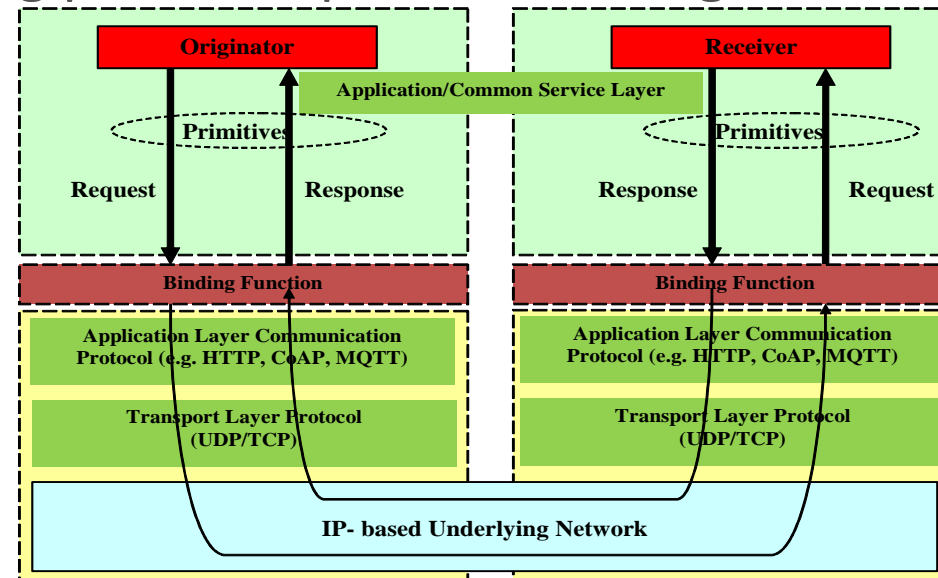


JSON to CBOR

```
a6427063a1466d326d3a6165a342726e54536d617274486f6d654170706c69636174696f6e436170694  
44e6135364361706e47617070313233344274790242746f572f2f6578616d706c652e6e65742f6d6e637  
365313233344372636e07426f700143727169454131303030
```

Core Protocol

- Core Protocol and Bindings
 - oneM2M primitives are carried over HTTP/CoAP/MQTT/WebSocket protocols
 - primitive parameters are mapped to binding message existing/new headers and body, vice versa
 - there are some binding protocol specific handlings



Communication model using Request and Response primitives



Protocol Binding

Protocol Binding

- Binding Protocols
 - HTTP, CoAP and MQTT from Rel-1
 - WebSocket from Rel-2

	Server/Client	Publish/Subscribe
TCP	HTTP, WebSocket	MQTT
UDP	CoAP	-

Protocol Binding

- HTTP Binding
 - The Content parameter is carried in the body part
 - The other parameters are carried in the header part

HTTP Request:

```
POST /~/mn-cse/home_gateway/light_ae1/light?rcn=0 HTTP/1.1
Host: http://mn.provider.com:8080
X-M2M-Origin: /mn-cse/Clight_ae1
Content-Type: application/vnd.oneM2M-res+xml;ty=4
X-M2M-RI: mncse/24345
```

```
<m2m:cin xmlns:m2m="http://www.oneM2M.org/xml/protocols">
<con>OFF</con>
<cnf>text/plain:0</cnf>
</m2m:cin>
```

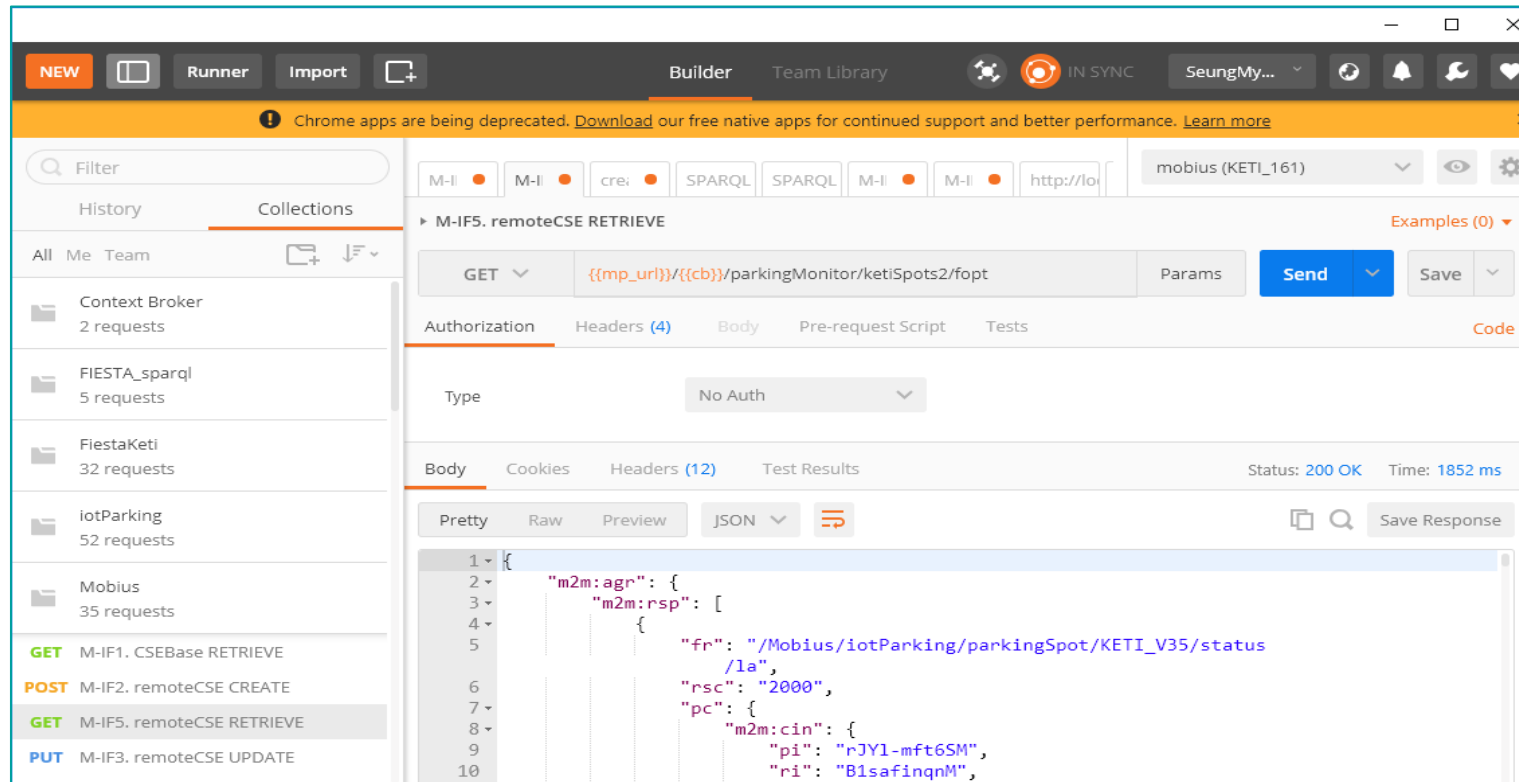
HTTP Response:

```
201 Created
X-M2M-RSC: 2001
X-M2M-RI: mncse/24345
Content-Location: /mn-cse/cin-394798749
Content-Type: application/vnd.oneM2M-res+xml
```

oneM2M Parameter	HTTP Header/component
<i>To</i>	path component
<i>From</i>	X-M2M-Origin
<i>Operation</i>	Method
<i>Filter Criteria</i>	query component
<i>Response Status Code</i>	Status-Code
<i>pointOfAddress</i> (attribute)	Host
<i>Resource Type</i>	Content-Type
<i>Request Identifier</i>	X-M2M-RI
notificationURI (element) of <i>Response Type</i>	X-M2M-RTU
<i>Response Status Code</i>	X-M2M-RSC
<i>Content</i>	Message-Body

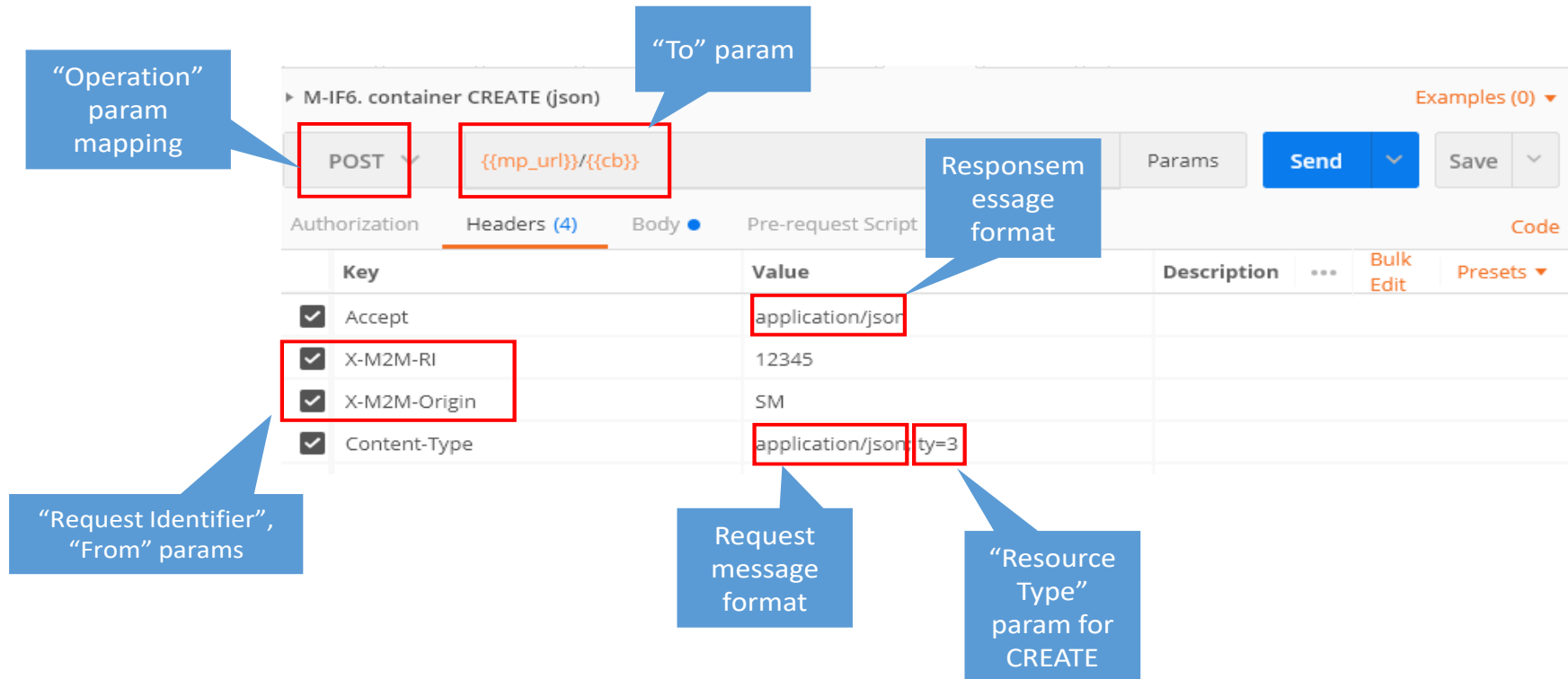
Protocol Binding

- Postman is the Chrome extension
 - Send a HTTP Request with Request-line, Headers, Body
 - And receives HTTP Response



Protocol Binding

- Postman Configurations for oneM2M APIs
 - oneM2M request parameters are mapped to HTTP request-line, headers, body
 - Postman provides GUI for HTTP message generation



The image shows a Postman configuration for the "M-IF6. container CREATE (json)" endpoint. The configuration is annotated with blue callouts explaining the mapping of oneM2M parameters to HTTP elements:

- "Operation" param mapping:** Points to the **POST** method dropdown.
- "To" param:** Points to the URL template `{{mp_url}}/{{cb}}`.
- Response message format:** Points to the **Headers** tab.
- "Request Identifier", "From" params:** Points to the **X-M2M-RI** and **X-M2M-Origin** headers.
- Request message format:** Points to the **Content-Type** header value `application/json`.
- "Resource Type" param for CREATE:** Points to the **ty=3** part of the **Content-Type** header value `application/json;ty=3`.

Key	Value	Description
<input checked="" type="checkbox"/> Accept	application/json	
<input checked="" type="checkbox"/> X-M2M-RI	12345	
<input checked="" type="checkbox"/> X-M2M-Origin	SM	
<input checked="" type="checkbox"/> Content-Type	application/json;ty=3	

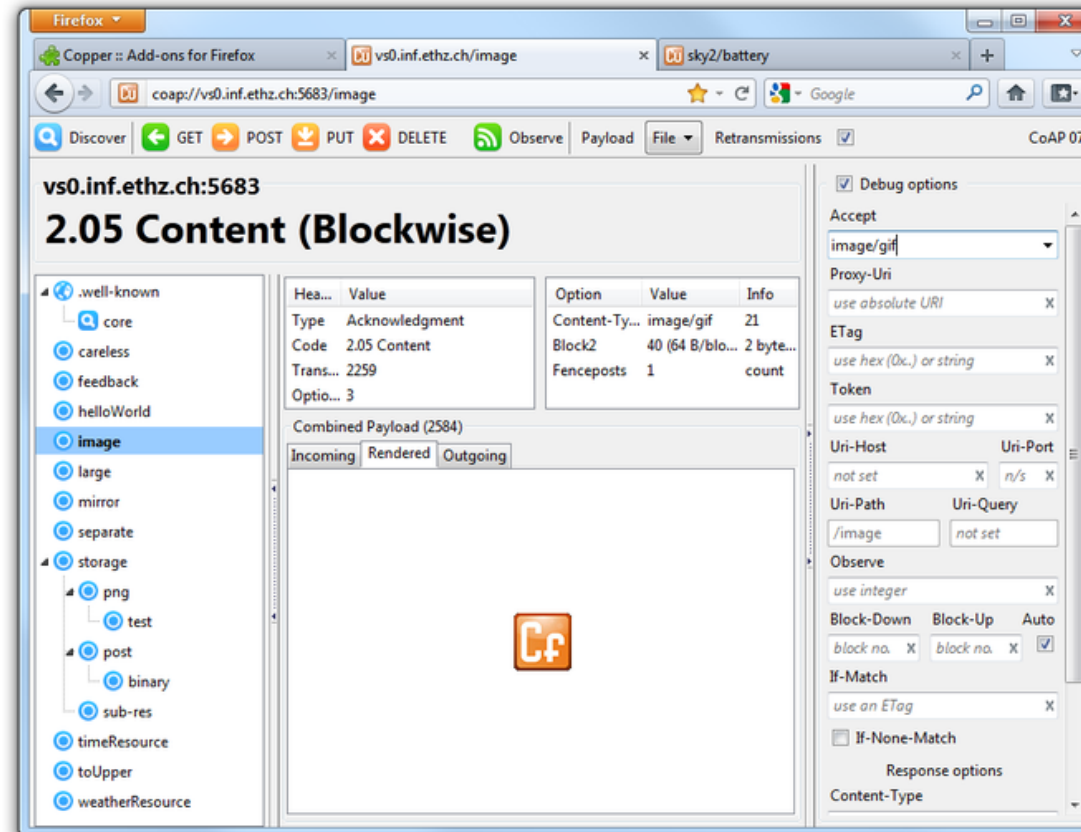
Protocol Binding

- CoAP binding
 - Similar to HTTP binding except the URI length constraint

oneM2M Parameter	CoAP Options
<i>To</i>	URI-Path
<i>From</i>	oneM2M-FR
<i>Operation</i>	Method
<i>Filter Criteria</i>	Uri-Query
<i>Response Status Code</i>	Response Code
<i>pointOfAddress</i> (attribute)	Uri-Host, Uri-Port
<i>Resource Type</i>	oneM2M-TY
<i>Request Identifier</i>	oneM2M-RQI
<i>Response Type</i>	Uri-Query
<i>Response Status Code</i>	oneM2M-RSC
<i>Content</i>	<i>payload</i>

Protocol Binding

- Copper is the Firefox add-on
 - Send a CoAP request and receives a response



- MQTT Binding
 - MQTT has no header concept
 - The whole request/response primitive representation in XML or JSON is carried in the message body
 - However, some parameters are carried in a MQTT topic
 - E.g. “/oneM2M/req/<originator>/<receiver>”
 - MQTT topics
 - Registration topic using credential
 - Request/response topics using Originator ID and Receiver ID
 - To parameter, which is the target resource address, is carried in the body, while the Receiver’s entity ID is included in the topics

Protocol Binding

- MQTT lens is the Chrome extension
 - There are more MQTT clients

