Rigorous Design and Deployment of IoT Applications

Ajay Krishna | Michel Le Pallec | Radu Mateescu | Ludovic Noirie | Gwen Salaün


27 May 2019, Montreal, QC, Canada
Context

Internet of Things (IoT) applications are heterogeneous and concurrent.

Difficult for end-users to build correct applications by composing various objects.

Formal modeling and verification can help users build correct applications.
Modelling behavior: Switch
Modelling behavior: Philips Hue Lamp

Off
On
Change Color
Interface-based Modeling

https://meethue.com/api/key/lights/1/state

```json
{
  "on": false
}
```

```json
{
  "on": true
}
```

```json
{
  "hue": 50000,
  "bri": 200
}
```
IoT Object Model

0 = < I_{in}, I_{out}, LTS >

LTS = < S, A, T, s_0 >

A \subseteq I_{in} \cup I_{out} \cup \{\tau\}

A \subseteq I_{in} \cup I_{out} \cup \{\tau\} \ (s_1, a, s_2) \in T

s_0 \in S

I_{in} = \{ S\_ON, S\_OFF, MOTION \}

I_{out} = \{ ALERT \}
Bindings

Binding $\beta = (i_{out}^{o_1}, i_{in}^{o_2})$

$\text{in}(\beta) = i_{in}^{o_2}$

$\text{out}(\beta) = i_{out}^{o_1}$

Notion of Strong and Weak Bindings

**Strong**: Functionally important

**Weak**: Optional (can provide additional service)
Composition

\[ C = \langle B, \Sigma, I_{in}^U, I_{out}^U, LTS, W \rangle \]

\[ B = \{ \beta_1, \beta_2, \ldots \beta_n \} \]

\[ \Sigma = \{ O_1, O_2, \ldots O_n \} \]

\[ I_{in}^U = (I_{in_1} \cup I_{in_2} \cup \ldots \cup I_{in_n}) \setminus \text{in}(B) \]

\[ I_{out}^U = (I_{out_1} \cup I_{out_2} \cup \ldots \cup I_{out_n}) \setminus \text{out}(B) \]

\[ LTS = \text{ren}_{\rho_B}(LTS_1) \otimes_B \ldots \otimes_B \text{ren}_{\rho_B}(LTS_n) \]

\[ W \subseteq B \quad \text{//weak bindings} \]
A service composition is correct if all bindings can effectively be executed and if all reachable actions unbound in the composition do not prevent the bindings to be executed
Compatibility: Example

\[
\text{hide}_{(I_{\text{in}} \cup I_{\text{out}})(\text{LTS})}(\text{LTS}) \equiv_{br} \text{LTS} \otimes_A \text{chaos}_B
\]
Deployment

- Bindings describe dependencies among the objects

- A composition can be viewed as a directed graph. Objects as nodes and bindings as edges

- Dependencies can be identified by inverse topological sorting

- Discard weak bindings in case of cyclic dependencies
Deployment Plan

A deployment plan consists of sequence of steps involving 3 operations at network level (SDN)

**ADD** – Provisioning of connected objects

**BIND** – Network configuration to allow communication between app interfaces

**START** – Enables app interfaces
Tool Support: IoT Composer
LNT Specification

**Object**

```plaintext
module mediadevice is
    process mediadevice_idle [on, video, aux, audio: any] is
        select
            on; mediadevice_video [on, video, aux, audio]
        []
            aux; mediadevice_audio [on, video, aux, audio]
        end select
    end process
end module
```

**Composition**

```plaintext
module prodall (phone, mediadevice, speaker) is
    process prod [on, video, aux, audio: any] is
        par
            on, video -> phone_idle [on, video]
        ||
            on, video, audio -> mediadevice_idle [on, video, aux, audio]
        ||
            audio -> speaker_idle [audio]
        end par
    end process
end module
```
Objects can interact with different IoT services but preserves network isolation and discovery properties
Experiments

<table>
<thead>
<tr>
<th>Use case</th>
<th>Obs</th>
<th>Bind</th>
<th>LTS States</th>
<th>Trans.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Media</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Thermo</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>SmartDoor1</td>
<td>3</td>
<td>4</td>
<td>8</td>
<td>13</td>
</tr>
<tr>
<td>LightSense</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>SmartDoor2</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>BabyMonitor</td>
<td>4</td>
<td>6</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>SmartAccess</td>
<td>5</td>
<td>8</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>MultiDoor</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>MultiCase</td>
<td>10</td>
<td>12</td>
<td>36</td>
<td>154</td>
</tr>
<tr>
<td>MultiCase2</td>
<td>13</td>
<td>15</td>
<td>176</td>
<td>892</td>
</tr>
<tr>
<td>IndepCase1</td>
<td>16</td>
<td>18</td>
<td>876</td>
<td>5134</td>
</tr>
<tr>
<td>IndepCase2</td>
<td>20</td>
<td>24</td>
<td>10501</td>
<td>79886</td>
</tr>
</tbody>
</table>

Number of objects (n)  Compatibility check (secs)
Summary

- IoT Objects: $O_1, O_2, O_3...$
- SDN-based Deployment

**Abstraction**
- Choose
- Physical Env.

**Behavioural Model**
- $M_1$, $M_2$, $M_3$
- $\beta_1$, $\beta_2$, $\beta_3$
- Generate

**Composition**
- $S_0$, $S_1$
- OFF, ON
- Verify

**Compatibility Check**
- Equivalence Checking
- Generate

**Deployment Plan**
- Dependency Model
- Deploy

Incompatible
Concluding Remarks

Formal modelling and analysis contributes to correct composition and deployment.

Proposals implemented as a tool for end-users.

Future work on Thing Description, QoS and performance analysis of IoT services, and reconfiguration scenarios.