A Case Study for Workflow-based Automation in the Internet of Things

IoT-ASAP @ ICSA 2018

Ronny Seiger, Steffen Huber, Uwe Aßmann

Seattle, 30.04.2018
Workflows in IoT

Workflow Layer

Service Layer

Middleware Layer

Driver/Control Software Layer

Hardware Layer
Smart Home Example Processes

Morning Routine Process

1. Move Robot to Paperboy
2. Robot arrived?
3. Place Paper on Robot
4. Move Robot to Reader
5. Robot arrived?
6. Switch on Light
7. Brew Coffee

Emergency Process

1. Ask about Health Status
2. Healthy AND Responsive?
3. Call Emergency Service
4. Authentication successful?
5. Unlock Door
6. Medical Personnel arrives
7. Health Alarm triggered
8. 24 Hour Emergency Service
• Modelling and execution of workflows/processes in IoT
  → describe interactions of IoT elements on business process level
    – Heterogeneous devices
    – Complex sensor networks
    – Resource-constraint, mobile or stationary actuators
    – Humans
    – Smart objects
    – Software services and applications

• Resilient workflow execution
  → detect and handle errors and unanticipated situations
    – Interactions with the physical world
    – Mutual influence between physical and cyber world (CPS)
- Complex event processing, Web service invocations, Human Tasks
- Dynamic service selection (Ontology + Semantic Queries): SAL\(^2\) [4]
- Self-adaptive workflow execution: Feedback Service\(^3\) [5,6]
  - Goals define success or error criteria
  - Analysis of external sensor data
  - Process adaptation in case of errors

(1) https://github.com/IoTUDresden/proteus
(2) https://github.com/IoTUDresden/openhab2-addons
(3) https://github.com/IoTUDresden/feedback-service
Real world case study based on scenario processes
- Morning Routine process and Emergency process
- Coffee process and Robot Navigation process

Controlled lab experiments
- 1 control computer (Ubuntu Linux) with PROtEUS WfMS + associated services, middleware (OpenHAB)
- Sensors and actuators from various vendors (Homematic, Tinkerforge, ...)
- Turtlebot 2 robots, Android tablets
- BeSpoon tracking system
**Morning Routine Process**

- **Dynamic sensor selection**
  - Position
- **RESTful services**
  - Send robot to target
  - Trigger light and coffee
- **Event processing**
  - Robot arrived
- **Human Task**

---

**Process Model [1]**

- **OpenHabSemanticInvoke**
  - name: GetPositionForPaperBoy (P1)
    - type: LocationQuery
  - name: GetPositionForReader (P2)
    - type: LocationQuery

- **RESTInvoke**
  - name: MoveRobotToPaperBoy (P3)
    - type: RobotMovement
    - resource: TurtleBot
    - server: 192.168.1.3:8090/rest
    - HTTP method: POST
  - name: MoveRobotToReader (P6)
    - type: RobotMovement
    - resource: TurtleBot
    - server: 192.168.1.3:8090/rest
    - HTTP method: POST
  - name: BrewCoffee (P9)
    - type: CoffeeBrew
    - resource: SmarterCoffee
    - server: 192.168.3.8090/rest
    - HTTP method: GET
  - name: SwitchOnLight (P8)
    - type: LightControl
    - resource: HueLamp
    - server: 192.168.3.8090/rest
    - HTTP method: GET

- **TriggeredEvent**
  - name: WaitForRobotToArriveAtPaperBoy (P4)
    - type: LocationCheck
    - resource: ROSLocation
    - EPL: select * from SensorEvent where cast(value?, string) like ‘ARRIVED%’ and deviceUid = ‘proteus_turtle_movement_1’
  - name: WaitForRobotToArriveAtReader (P7)
    - type: LocationCheck
    - resource: ROSLocation
    - EPL: select * from SensorEvent where cast(value?, string) like ‘ARRIVED%’ and deviceUid = ‘proteus_turtle_movement_1’

- **HumanTask**
  - name: PlacePaperOnTurtle (P5)
    - type: HumanTask
    - resource: PaperBoy
Results: Morning Routine Process

- Fast virtual process executions (near real-time, < 100 ms)
- Cyber-physical process steps much longer
- Asynchronous service invocations → require event listeners
A Case Study for Workflow-based Automation in IoT

Event processing

- Human Task
- Dynamic service selection

Emergency Process

30.04.2018
Results: Emergency Process

<table>
<thead>
<tr>
<th>ID</th>
<th>Process Step</th>
<th>Duration (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>HealthChanged</td>
<td>27916</td>
</tr>
<tr>
<td>P2</td>
<td>ShowQuestionOnScreens</td>
<td>181</td>
</tr>
<tr>
<td>P3</td>
<td>ShowQuestionOnDisplays</td>
<td>47</td>
</tr>
<tr>
<td>P4</td>
<td>SwitchLcdBacklightsOn</td>
<td>57</td>
</tr>
<tr>
<td>P5</td>
<td>AllLightsOn</td>
<td>111</td>
</tr>
<tr>
<td>P6</td>
<td>AskIfHumanOk</td>
<td>15046</td>
</tr>
<tr>
<td>P7</td>
<td>OR</td>
<td>7</td>
</tr>
<tr>
<td>P8</td>
<td>ShowCallEmergency</td>
<td>131</td>
</tr>
<tr>
<td>P9</td>
<td>ShowHelpsCalled</td>
<td>55</td>
</tr>
<tr>
<td>P10</td>
<td>WaitForNfc</td>
<td>21213</td>
</tr>
<tr>
<td>P11</td>
<td>OpenDoor</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Emergency</td>
<td>64389</td>
</tr>
</tbody>
</table>

30.04.2018
Video: Emergency Process
- Process invokes coffee brewing service
- Use MAPE-K loop to verify process execution
  - Monitor, Analyze, Plan, Execute
- External sensor (infrared) + Feedback Service
- Goal defines success and error criteria

```json
"MakeCoffee" : {
  "name":"Coffee is ready",
  "objectives": [ {
    "name":"coffee temperature > 37 degrees within 3 minutes",
    "satisfiedCondition":"#coffeeTemp > 37",
    "compensationCondition":"#objective.created.
      isBefore(#now.minusSeconds(180))",
    "contextPaths": [
      "MATCH (ctemp {name: 'State_tinkerforge_irTemp_irTemp_1'})[:
        hasStateValue]->(value)",
      "WHERE toFloat(value.realStateValue)>0",
      "RETURN toFloat(value.realStateValue) AS
        coffeeTemp, id(ctemp) AS stateId"
    ] } ] }
```

30.04.2018
Results: Coffee Process

- Link external sensor data to workflow execution
- Verification of process execution
- Fast execution times

**Results:**

<table>
<thead>
<tr>
<th>ID</th>
<th>Process Step</th>
<th>Duration (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>MakeCoffee</td>
<td>114.940</td>
</tr>
<tr>
<td>P2</td>
<td>IF</td>
<td>38</td>
</tr>
<tr>
<td>P3</td>
<td>CoffeeSuccess</td>
<td>35645</td>
</tr>
<tr>
<td>P4</td>
<td>OR</td>
<td>9</td>
</tr>
<tr>
<td>Total</td>
<td>CoffeeProcess</td>
<td>150.632</td>
</tr>
</tbody>
</table>

**Phase Metrics:**

<table>
<thead>
<tr>
<th>Metrics</th>
<th>M</th>
<th>A</th>
<th>P</th>
<th>E</th>
<th>Loop</th>
<th>FB Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations (#)</td>
<td>148</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>148</td>
<td>1</td>
</tr>
<tr>
<td>Duration ((\bar{\phi}) in ms)</td>
<td>1023</td>
<td>112.135</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Robot Navigation Process

- Process invokes robot service to drive to a target
- Robot’s internal SLAM localization prone to errors
- External sensor data (BeSpoon localization) to verify correct position
- Verification and compensation planning in MAPE-K loop (Goal)
  - Success: robot reached specified coordinates and publishes “arrived” event
  - Error: robot publishes “arrived” event and has not reached coordinates
- Error: Planner decides to cancel MAPE-loop and reports error → Process loop is repeated → robot is re-instructed
Results: Robot Navigation Process

- Process verification with external sensors
- Detect and remedy errors
- Simple planning → repeat process step
- Extensible strategies (replace resources)
• Workflows for automation in IoT → on top of sensors, actuators, humans, objects, services
• Active and reactive interactions on the business process level
• Smart home case study → real world experiments
• Fast virtual computations
• Long running physical executions
• Sensor data ↔ workflow execution → execution verification
• MAPE-K loop-based self-management for resilient processes

Questions?
References


