Dynamical Systems on the Web: Classification and Challenges

Prof. Dr. Andreas Harth
Friedrich-Alexander-University Erlangen-Nuremberg and Fraunhofer IIS-SCS

Joint work with colleagues from KIT: Tobias Kaefer, Sebastian Speiser and Steffen Stadtmueller

Fourth Stream Reasoning Workshop, Linköping University, April 2019
Agenda

1. Internet Architecture vs. Web Architecture vs. Linked Data
2. User Agents in Dynamical Web Environments
3. Scenarios and Evaluation
4. Conclusion
Internet Architecture: Sockets

- Host
- IP address
- Port
- Channel
- Port
- IP address
The Web vs. the Internet

https://www.w3.org/20/
Web Architecture

- URIs act as names for resources: RFC 1630 (1994), now RFC 3986
- HTTP to interact with resources/resource state: RFC 1945 (1996), now RFC 7230 - 7235

- Web architecture assumes a strict separation between user agents and servers
- User agents emit requests, receive response
- Servers answer to incoming requests with a response

https://www.w3.org/DesignIssues/diagrams/history/Architecture_crop.png
### Semantics of HTTP Messages

<table>
<thead>
<tr>
<th>HTTP Request Method</th>
<th>HTTP Request, or Response Code</th>
<th>HTTP Message Semantics: The HTTP Message Body contains...</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Request</td>
<td>Nothing</td>
</tr>
<tr>
<td>PUT</td>
<td>Request</td>
<td>State of the resource</td>
</tr>
<tr>
<td>POST</td>
<td>Request</td>
<td>State of the resource or arbitrary data</td>
</tr>
<tr>
<td>DELETE</td>
<td>Request</td>
<td>Nothing</td>
</tr>
<tr>
<td>Any</td>
<td>Non-2xx</td>
<td>State of the request</td>
</tr>
<tr>
<td>GET</td>
<td>2xx</td>
<td>State of the resource</td>
</tr>
<tr>
<td>PUT</td>
<td>2xx</td>
<td>State of the resource or empty</td>
</tr>
<tr>
<td>POST</td>
<td>2xx</td>
<td>State of the request (referring to new resource)</td>
</tr>
<tr>
<td>DELETE</td>
<td>2xx</td>
<td>State of the request or empty</td>
</tr>
</tbody>
</table>

RDF Dataset

**Definition**  (Named Graph, RDF Dataset). Let $\mathcal{G}$ be the set of RDF graphs and $\mathcal{U}$ be the set of URIs. A pair $\langle g, u \rangle \in \mathcal{G} \times \mathcal{U}$ is called a named graph. An RDF dataset consists of a (possibly empty) set of named graphs (with distinct names) and a default graph $g \in \mathcal{G}$ without a name.
Web Architecture/Linked Data

• User agent:
• RDF dataset $S \subseteq \text{Web}$

• Servers:
• RDF dataset Web (infinite)
Agenda

1. Internet Architecture vs. Web Architecture vs. Linked Data
2. User Agents in Dynamical Web Environments
3. Scenarios and Evaluation
4. Conclusion
Cognitive Architectures

• SOAR (initially: State, Operator, Apply, Result),
• ACT-R (Adaptive Control of Though – Rational)
• Goal: to create „intelligent agents“
• For starters we only consider user agents that are
  • „simple reflex agents“ (Russel & Norvig, see figure),
  • aka „tropistic agents“ (Genesereth & Nilson)
• We use rules to control the agent’s behaviour
• What the world is like now:
  • safe HTTP methods (GET)
• What action should I do now:
  • unsafe HTTP methods

### Some Models of Computation

#### Candidates

<table>
<thead>
<tr>
<th>Model of Computation</th>
<th>Main Mismatch to Linked Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lambda Calculus</td>
<td>Based on Events / Functions</td>
</tr>
<tr>
<td>Pi Calculus</td>
<td>Based on Events / Channels</td>
</tr>
<tr>
<td>Petri Nets</td>
<td>Based on Events</td>
</tr>
<tr>
<td>Graph Rewriting</td>
<td>Unclear data access + FOL-handling</td>
</tr>
<tr>
<td>Turing Machine</td>
<td>Abstraction too low-level</td>
</tr>
<tr>
<td>Finite State Machines</td>
<td>Unclear state + condition representation</td>
</tr>
<tr>
<td>Abstract State Machines</td>
<td>?</td>
</tr>
</tbody>
</table>

#### Abstract State Machines [G]

- Provide a good fit to Linked Data
  - First-order logic-based (cf. RDF(S)/OWL)
  - State as first-class citizen (HTTP)
- About the evolution of first-order structures (aka. states)
  - Specifically, how the interpretation of function names changes over time

![Diagram](https://via.placeholder.com/150)

- Evolution (so-called transition function) in rules:
  - If condition(s) hold then update the interpretation(s)
- Execution in ASM Steps:
  - Collect all updates, execute updates in bulk

---

Abstract State Machines for Linked Data

Basic Definitions / Simplifications:
- Ground graphs (i.e. no blank nodes)
- $U$ – the set of all URIs, $L$ – the set of all Literals, interpreted to the same resources in all graphs
- $IP \subseteq IR$ (required in RDF and more constrained interpretations)
- No HTTP redirects

1. Define RDF model theory for Linked Data using RDF datasets
   - Different extension functions ($IEXT$) in [Z] for RDF datasets:
     a) For updates: Named Graphs are in a particular relation with what the graph refers to
        $$IEXT_c := \text{Extension function of the graph available at } c$$
     b) For conditions: Default graph as union or as merge
        $$IEXT^{\text{UNION}} = \bigcup_c IEXT_c$$

2. Define ASM functions for the model theoretic views on Linked Data / RDF datasets
   - $quad(\cdot,\cdot,\cdot,\cdot): IR \times IR \times IR \times IR \rightarrow \{\text{TRUE, UNDEF}\}$ the ASM characteristic function for the set of all quads $\sim IEXT$ in a)
   - $statement(\cdot,\cdot,\cdot): IR \times IR \times IR \rightarrow \{\text{TRUE, UNDEF}\}$ the ASM characteristic function for the set of all triples $\sim IEXT$ in b)

Abstract State Machines for Linked Data

3. Define an ASM transition function $T$ for the Linked Data ASM functions
   • If conditions hold in $\text{statement}(;,;) \text{ then update } \text{quad}(;,;,;)
   • Conditions in $\text{statement}(;,;) \sim \text{SPARQL BGP Queries}$

4. Define how the ASM evaluation of the ASM functions maps to the HTTP request semantics
   • $\text{statement}(;,;) \text{ in conditions } \sim \text{GET request to all sources}$
   • $\text{quad}(;,;,;) \text{ in updates } \sim \text{PUT request to given source(s)}$

5. Define the ASM $(Y, X, I, T)$ using the semantic conditions
   • $Y := U \cup L \cup \{\text{true, undef}\} \cup \{\land\} \cup \{\text{quad, statement}\}$
   • $X := IR \cup IP \cup \{\text{true, undef}\} \cup \{f : X^n \rightarrow X\}$
     $$I_S(y) \quad \text{if } y \in U$$
     $$I_L(y) \quad \text{if } y \in L$$
   • $I_t(y) :=
     \begin{cases} 
     \text{TRUE} & \text{if } y = \text{true} \\
     \text{UNDEF} & \text{if } y = \text{undef} \\
     \in \{f | f: X^n \rightarrow X\} & \text{if } y \in \{\text{quad, statement, \land}\} \\
     \end{cases}$
   • Execute following ASM steps: First evaluate all conditions, then apply the collected updates in bulk
Result: Operational Semantics for the Linked Data-Fu Language [SSHS]

@prefix ... # See prefix.cc

Assertions

```
<http://building3.example/lamps/0#l>
    rdf:type saref:LightingDevice;
    ssn:hasProperty <http://building3.example/lamps/0/power#p> .
```

Deductions

```
{ ?thing ssn:hasProperty ?prop . }
=> { ?prop ssn:isPropertyOf ?thing . } .
```

Conditional GET requests

```
{ ?y ssn:isPropertyOf ?x . }
```

Conditional PUT/POST/DELETE requests

```
{ ?lamp a saref:LightingDevice .
  ?property ssn:isPropertyOf ?lamp ; rdf:value "off" . }
=> { [] http:mthd httpm:PUT; http:requestURI ?property;
    http:body { ?property rdf:value "on" . } . } .
```

Agenda

1. Internet Architecture vs. Web Architecture vs. Linked Data
2. User Agents in Dynamical Web Environments
3. Scenarios and Evaluation
4. Conclusion
Scenario: Data Processing within Virtual and Augmented Reality Environments

Scenario “Virtual Airplane Pilot Training”

- **Workflows** of the pilot in a plane are derived by human factor methods
  - E.g., behavior during emergency landing

- **Decisions** depend on a multitude of environment conditions, including the behavior of the actual pilot

- **Linked Data-Fu**: specification and execution of dynamic workflows in a real training scenario
**Scenario: Building Automation/Evaluation**

1. **Formal: Turing Completeness**

2. **Performance**
   - Automating Building 3 of IBM Dublin, as described using the Brick ontology [1]
   - Interpreter: Linked Data-Fu
   - W1→W5 differently complex automation
   - D1: GET building data from *one source*
   - D2: GET building data from *many sources* following links

<table>
<thead>
<tr>
<th>Rooms</th>
<th>W1</th>
<th>W2</th>
<th>W3</th>
<th>W4</th>
<th>W5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>484</td>
<td>572</td>
<td>510</td>
<td>554</td>
<td>561</td>
</tr>
<tr>
<td>5</td>
<td>480</td>
<td>582</td>
<td>501</td>
<td>574</td>
<td>582</td>
</tr>
<tr>
<td>10</td>
<td>498</td>
<td>584</td>
<td>529</td>
<td>605</td>
<td>618</td>
</tr>
<tr>
<td>20</td>
<td>537</td>
<td>631</td>
<td>562</td>
<td>719</td>
<td>687</td>
</tr>
<tr>
<td>First Floor</td>
<td>563</td>
<td>629</td>
<td>590</td>
<td>750</td>
<td>728</td>
</tr>
<tr>
<td>Wing 42</td>
<td>527</td>
<td>595</td>
<td>550</td>
<td>651</td>
<td>604</td>
</tr>
<tr>
<td>Building 3</td>
<td>605</td>
<td>734</td>
<td>613</td>
<td>794</td>
<td>788</td>
</tr>
</tbody>
</table>

**Median Time [ms] for One ASM Step in D1**

<table>
<thead>
<tr>
<th>Rooms</th>
<th>W1</th>
<th>W2</th>
<th>W3</th>
<th>W4</th>
<th>W5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
<td>38</td>
<td>38</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>10</td>
<td>85</td>
<td>80</td>
<td>79</td>
<td>88</td>
<td>88</td>
</tr>
<tr>
<td>20</td>
<td>259</td>
<td>238</td>
<td>228</td>
<td>320</td>
<td>268</td>
</tr>
<tr>
<td>First Floor</td>
<td>938</td>
<td>1690</td>
<td>891</td>
<td>1063</td>
<td>1048</td>
</tr>
<tr>
<td>Wing 42</td>
<td>1435</td>
<td>1427</td>
<td>1371</td>
<td>1664</td>
<td>1408</td>
</tr>
<tr>
<td>Building 3</td>
<td>2442</td>
<td>2187</td>
<td>2192</td>
<td>2542</td>
<td>2497</td>
</tr>
</tbody>
</table>

**Median Time [ms] for One ASM Step in D2**

**Building 3 and Benchmark Statistics**

<table>
<thead>
<tr>
<th></th>
<th>Rooms</th>
<th>Floors</th>
<th>Wings</th>
<th>Lights w/ occupancy sensors</th>
<th>Lights w/ luminance sensors</th>
<th>Triples in IBM_B3.ttl</th>
<th>Resources in the LDP container</th>
<th>Dynamic resources (sensors)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Same order of magnitude</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Raising with number of Rooms</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Balancing IO and Reasoning on LUBM-LD(100)

Batch Processing: first IO, then processing, then IO,...

Stream Processing: IO and processing intertwined

InputWorker threads: I/O, ProcessingWorker threads: reasoning/materialisation

Agenda

1. Internet Architecture vs. Web Architecture vs. Linked Data
2. User Agents in Dynamical Web Environments
3. Scenarios and Evaluation
4. Conclusion
Conclusion

• We have shown how to build systems in building automation and mixed reality systems
• Other scenarios could involve virtual assistants
• We adapted Abstract State Machines for Linked Data as a formalism for specifying user agent behaviour, and used ASM4LD to give an operational semantics to a workflow vocabulary
• We have a multithreaded implementation (Linked Data-Fu)

• Future work: How to use link-following for discovery and goal-directed user agent behaviour?
• Future work: How to leverage formalisms based on state machines to perform model checking (finite models) or simulation (infinite models)?
Further Reading

• Tobias Käfer, Andreas Harth. "Specifying, Monitoring, and Executing Workflows in Linked Data Environments". International Semantic Web Conference 2018 (ISWC 2018), October 8-12, 2018, Monterey, California, USA.


Time: Synchronised Clocks

- Synchronised clocks are difficult to achieve in distributed systems with many participants
- On the web, resource state is usually just “now”