Detection of SOA Patterns

Anthony Demange  Naouel Moha  Guy Tremblay

Département d’informatique, LATECE Lab.
Université du Québec à Montréal

anthonydemange@gmail.com, \{moha.naouel, tremblay.guy\}@uqam.ca

December 4, 2013
SOA Systems Complexity

- Constant evolution of SOA systems
- SOA Patterns help solve common design problems
- Detection techniques assess systems quality
SOA Patterns: An Example

Intermediate Routing (Erl, 2009)
Goal = Assess the design and QoS of SOA systems through detection of SOA Patterns.

SODOP Approach

- Service Oriented Detection Of Patterns
- Extension of SODA approach (Moha et al., ICSOC 2012)
Outline

1. How to assess SOA systems quality?

2. Related Work

3. SODOP Approach to Pattern Specification and Detection

4. Pattern Detection Experiments on Two SOA systems

5. Results

6. Conclusion
## (Anti-)Patterns Specification and Detection

<table>
<thead>
<tr>
<th></th>
<th>Specification</th>
<th>Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OO Paradigm</strong></td>
<td>Gamma et al. (1994)</td>
<td>Antoniol et al. (1998)</td>
</tr>
<tr>
<td></td>
<td>Brown et al. (1998)</td>
<td>Munro et al. (2005)</td>
</tr>
<tr>
<td></td>
<td>Fowler et al. (1999)</td>
<td>Rasool et al. (2010)</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>Rotem-Gal-Oz (2012)</td>
<td></td>
</tr>
</tbody>
</table>
Summary

*OO techniques cannot be directly applied on SOA systems:
  * classes vs. services
  * mainly static detection vs. highly dynamic detection

⇒ No approach for the detection of SOA patterns.
Outline

1. How to assess SOA systems quality?
2. Related Work
3. SODOP Approach to Pattern Specification and Detection
4. Pattern Detection Experiments on Two SOA systems
5. Results
6. Conclusion
SODOP Approach to SOA Patterns Detection

1. **Rule Cards Specification**
   - SOA Patterns informal description
   - DSL
   - Rule cards

2. **Detection Algorithms Generation**
   - <Automatic>
   - Detection algorithms

3. **Patterns Detection**
   - Services identified as Patterns
   - <Automatic>
   - SOA system
SOA Patterns Specification

SOA Patterns

1. Basic Service
2. Facade
3. Proxy
4. Router
5. Adapter

Anthony Demange (UQAM)
Detection of SOA Patterns
December 4, 2013
Facade Pattern

1 RULE_CARD: Facade {
2 RULE: Facade {INTER HighDR
3 LowOCR HighRT};
4 RULE: HighDR {DR ≥ HIGH};
5 RULE: LowOCR {NIC/NOC ≤ LOW};
6 RULE: HighRT {RT ≥ HIGH};
7 };

Anthony Demange (UQAM)  Detection of SOA Patterns  December 4, 2013
Router Pattern

1 RULE_CARD: Router {
2   RULE: Router {HighPOPC};
3   RULE: HighPOPC {POPC ≥ HIGH};
4 }

Detection of SOA Patterns
Domain Specific Language for Pattern Specification

1. rule_card ::= RULE_CARD: rule_card_name { (rule)^+ };

2. rule ::= RULE: rule_name { content_rule };

3. content_rule ::= metric | set_operator rule_type (rule_type)^+

4. rule_type ::= RULE_CARD: rule_card_name

5. set_operator ::= INTER | UNION | DIFF | INCL | NEG

6. metric ::= metric_value comparator (metric_value | ordi_value | num_value)

7. metric_value ::= id_metric (num_operator id_metric)?

8. num_operator ::= + | - | * | /

9. id_metric ::= ANAM | ANIM | ANP | ANPT | COH | NID | NIR | NMD | NOR | NSC | TNP

10. | A | DR | ET | NDC | NIC | NOC | NTMI | POPC | PSC | SR | RT

11. ordi_value ::= VERY_LOW | LOW | MEDIUM | HIGH | VERY_HIGH

12. comparator ::= < | ≤ | = | ≥ | >

13. rule_cardName, ruleName ∈ string

14. num_value ∈ double
**Eight New Dynamic Metrics**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ET</strong></td>
<td>Execution Time</td>
</tr>
<tr>
<td><strong>NDC</strong></td>
<td>Number of Different Clients</td>
</tr>
<tr>
<td><strong>NIC</strong></td>
<td>Number of Incoming Calls</td>
</tr>
<tr>
<td><strong>NOC</strong></td>
<td>Number of Outgoing Calls</td>
</tr>
<tr>
<td><strong>DR</strong></td>
<td>Delegation Ratio</td>
</tr>
<tr>
<td><strong>SR</strong></td>
<td>Service Reuse (NIC/TotalCalls)</td>
</tr>
<tr>
<td><strong>POPC</strong></td>
<td>Proportion Of Path Changes</td>
</tr>
<tr>
<td><strong>PSC</strong></td>
<td>Proportion of Signature Changes</td>
</tr>
</tbody>
</table>
Generation Process

Two steps generation process

1. **EMF** meta-model
2. **Acceleo** code generation tool
Detection Process: SOFA framework

Detection of SOA Patterns

December 4, 2013 16 / 29
Outline

1. How to assess SOA systems quality?
2. Related Work
3. SODOP Approach to Pattern Specification and Detection
4. Pattern Detection Experiments on Two SOA systems
5. Results
6. Conclusion
Experiments Setup

**Goal:**
Show the **usefulness** and **efficiency** of SODOP

**Subjects:**
Five SOA Patterns

**Objects:**
*Home-Automation* (13 components)
*FraSCAti* (91 components - 130 services)

**Validation** (precision + recall):
Two external engineers
FraSCAti core team
Hypotheses

H1. Extensibility:
The proposed extended DSL is flexible enough to define SOA patterns.

H2. Accuracy:
The services identified as matching our SOA patterns must attain at least 80% of precision and 100% of recall.

H3. Performance:
The time needed by the detection algorithms must not impact the performance of the analyzed system.
Outline

1. How to assess SOA systems quality?
2. Related Work
3. SODOP Approach to Pattern Specification and Detection
4. Pattern Detection Experiments on Two SOA systems
5. Results
6. Conclusion
Results on Home-Automation

<table>
<thead>
<tr>
<th>Pattern Name</th>
<th>Detected Services</th>
<th>Metrics</th>
<th>Detect Time</th>
<th>Exec Time</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facade</td>
<td>mediator</td>
<td>NIC/NOC DR 0.17 1.0 2.8ms</td>
<td>10ms</td>
<td>6.66s</td>
<td>[1/1] 100%</td>
<td>[1/1] 100%</td>
</tr>
<tr>
<td>Router</td>
<td>mediator</td>
<td>POPC 0.5</td>
<td>11ms</td>
<td>6.67s</td>
<td>[1/1] 100%</td>
<td>[1/1] 100%</td>
</tr>
</tbody>
</table>
### Results on Home-Automation

<table>
<thead>
<tr>
<th>Pattern Name</th>
<th>Detected Services</th>
<th>Metrics</th>
<th>Detect Time</th>
<th>Exec Time</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy</td>
<td>patientDAO</td>
<td>NIC/NOC</td>
<td>13ms</td>
<td>6.74s</td>
<td>[1/1] 100%</td>
<td>[1/1] 100%</td>
</tr>
</tbody>
</table>

**Related Work**

Detection of SOA Patterns

Anthony Demange (UQAM)
Results on FraSCAti

<table>
<thead>
<tr>
<th>Pattern Name</th>
<th>Detected Services</th>
<th>Metrics</th>
<th>Detect Time</th>
<th>Execute Time</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facade</td>
<td>FraSCAti</td>
<td>NIC/NO</td>
<td>57ms</td>
<td>10.62s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>assembly-factory</td>
<td>DR</td>
<td></td>
<td></td>
<td>[3/3]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>composite-parser</td>
<td>RT</td>
<td></td>
<td></td>
<td>100%</td>
<td>[3/16]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NIC/NOC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Anthony Demange (UQAM)  Detection of SOA Patterns  December 4, 2013
H1: The DSL is **flexible** enough to define SOA patterns

1 RULE_CARD: Basic Service {
2    RULE: Basic Service {INTER HighSR
3        HighCOH HighA LowRT};
4    RULE: HighSR {SR ≥ HIGH};
5    RULE: HighCOH {COH ≥ HIGH};
6    RULE: HighA {A ≥ HIGH};
7    RULE: LowRT {RT ≤ LOW};
8 };

(a) Basic Service

1 RULE_CARD: Facade {
2    RULE: Facade {INTER HighDR
3        LowOCR HighRT};
4    RULE: HighDR {DR ≥ HIGH};
5    RULE: LowOCR {NIC/NOC ≤ LOW};
6    RULE: HighRT {RT ≥ HIGH};
7 };

(b) Facade

1 RULE_CARD: Adapter {
2    RULE: Adapter {INTER EqualOCR
3        HighPSC};
4    RULE: EqualOCR {NIC/NOC = 1.0};
5    RULE: HighPSC {PSC ≥ HIGH};
6 };

(d) Adapter

1 RULE_CARD: Proxy {
2    RULE: Proxy {INTER EqualOCR
3        HighSR LowPSC};
4    RULE: EqualOCR {NIC/NOC = 1.0};
5    RULE: HighSR {SR ≥ HIGH};
6    RULE: LowPSC {PSC ≤ LOW};
7 };

(c) Proxy

1 RULE_CARD: Router {
2    RULE: Router {HighPOPC};
3    RULE: HighPOC {POPC ≥ HIGH};
4 };

(e) Router
H2: At least 80% of precision and 100% of recall

<table>
<thead>
<tr>
<th>Results</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average on Home-Automation</td>
<td>93.3%</td>
<td>100%</td>
</tr>
<tr>
<td>Average on FraSCATi</td>
<td>100%</td>
<td>11.8%</td>
</tr>
</tbody>
</table>
**H3: No impact on the analyzed system performance**

<table>
<thead>
<tr>
<th>Results</th>
<th>Detect Time</th>
<th>Exec Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average on Home-Automation</strong></td>
<td>25ms</td>
<td>6.73s</td>
</tr>
<tr>
<td><strong>Average on FraSCAti</strong></td>
<td>97ms</td>
<td>10.9s</td>
</tr>
</tbody>
</table>
Outline

1. How to assess SOA systems quality?
2. Related Work
3. SODOP Approach to Pattern Specification and Detection
4. Pattern Detection Experiments on Two SOA systems
5. Results
6. Conclusion
Conclusion

SODOP Approach: 3 steps to detect SOA Patterns

1. Rule Cards Specification
2. Detection Algorithms Generation
3. Patterns Detection

SOA Patterns Specification

5 SOA Patterns:
1. Basic Service
2. Facade
3. Proxy
4. Router
5. Adapter

H1: The DSL is **flexible** to define SOA patterns

Results on Home-Automation

**Detection of SOA Patterns**

Anthony Demange (UQAM)
Thank you!

Questions?

Anthony Demange
anthonydemange@gmail.com

sofa.uqam.ca/sodop