A Vision to identify Architectural Smells in Self-Adaptive Systems using Behavioral Maps

Edilton Lima dos Santos
edilton.limados@unamur.be
Sophie Fortz
Gilles Perrouin
Pierre-Yves Schobbens

CASA@ECSA 2021, Växjö, Sweden, September 13th - 14th, 2021.

Agenda

Motivation

Proposed approach

Behavioral Map

Processes

Algorithm

Framework

CASE STUDY

Motivation

- Self-adaptive systems **change** their **behavior depending** on **environmental changes** and **reconfiguration plans**.

- Dynamic Software Product Line (**DSPL**) engineering **implements self-adaptive systems** by dynamically **binding or unbinding features**.

- **Binding or unbinding features** as prescribed by a **feature model.**

Motivation

- **Challenges**
  - **Number of possible configuration** goes exponentially with the number of features;
  - The **(re)configuration** process **may add** a **new architectural solution** in an **inappropriate context** via new features loaded;
  - The **(re)configuration** process **may combine architectural fragments** with **undesirable behaviors.**
- **Problem**
  - Architectural Smell**.**

Motivation

- **Studies about Architectural Smell in SAS:**
  - C. Raibulet, F. A. Fontana, S. Carettoni, **A preliminary analysis of self-adaptive systems according to different issues**, Software Quality Journal (2020) 1–31.
  - M. A. Serikawa, A. d. S. Landi, B. R. Siqueira, R. S. Costa, F. C. Ferrari, R. Menotti, V. V. De Camargo, **Towards the characterization of monitor smells in adaptive systems**, in: X Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS), IEEE, 2016, pp. 51–60.

Proposed approach

- We introduce **the** Behavioral Map

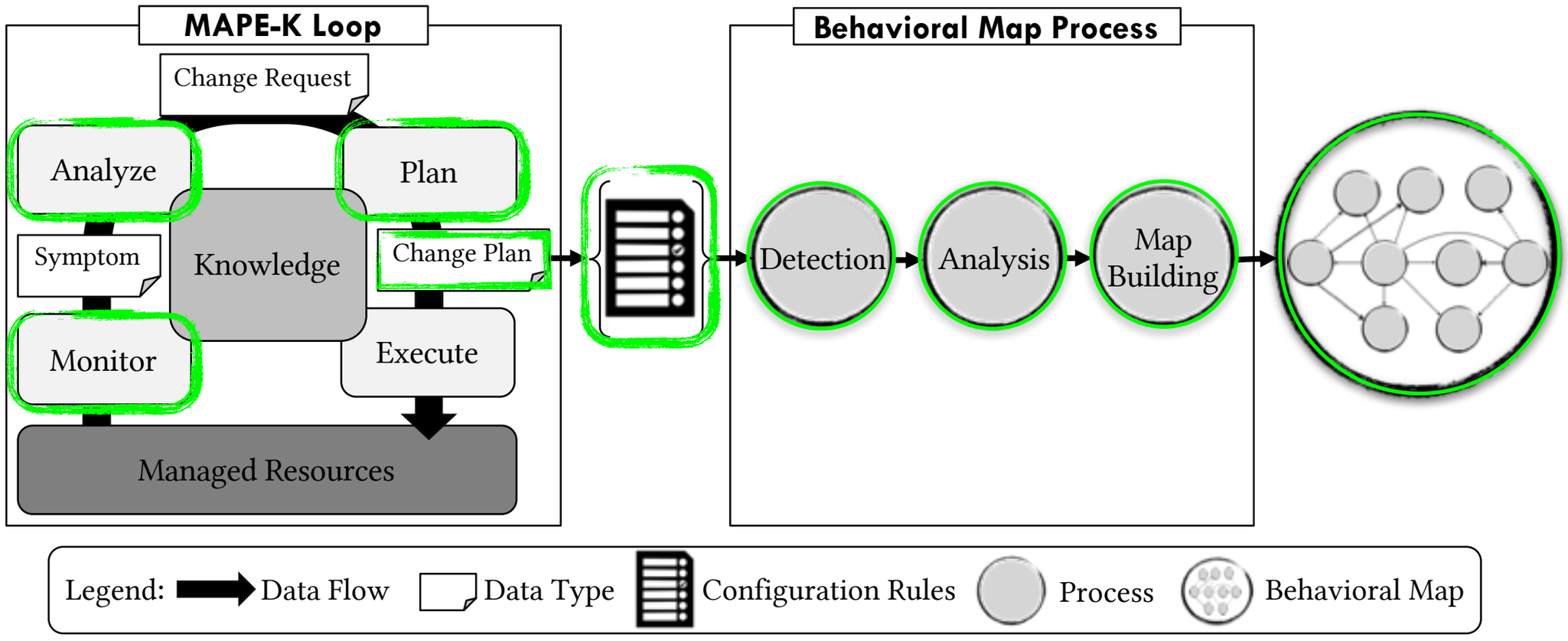
- A Behavioral Map (BM) can be seen as a *hybrid structure*, data, and control information about the self-adaptive system.

- BM maps the interactions and influences that a feature has on other features in a specific configuration for a given runtime context.

- The BM can support architectural bad smell identification at runtime.

Behavioral Map

Processes



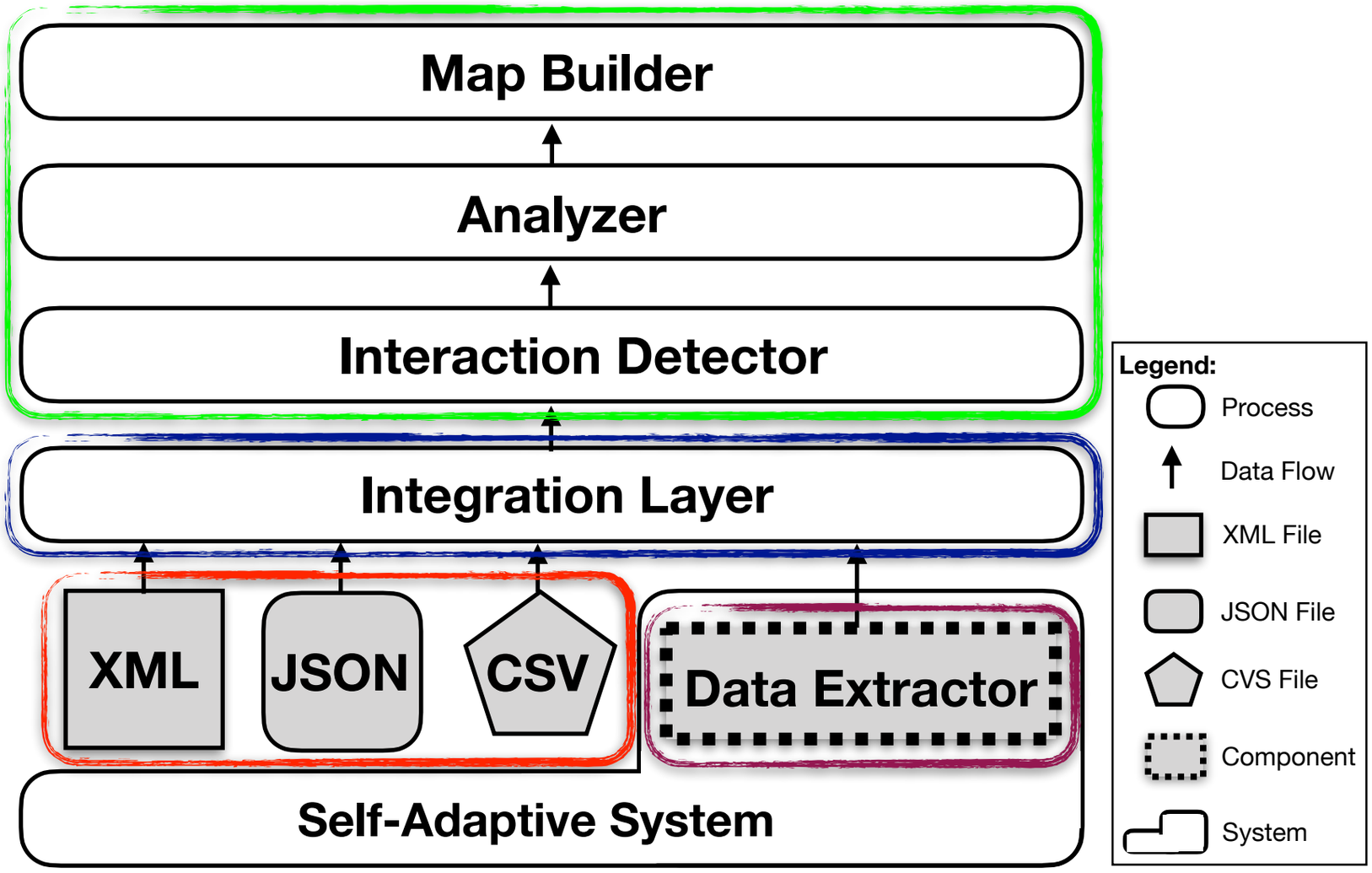
Behavioral Map

Algorithm

```
1  table ← loadConfigurationRulesFile(ℂℝfile);
2  verticesOnMap ← createVerticesOnMap(table);
3  foreach vertex in verticesOnMap do
4      foreach row in table do
5          if row.name.equals(vertex.name) then
6              foreach relation in row.getAllRelationships() do
7                  if relation.relationship is not null then
8                      createEdge(vertex, relation.relationship_type, relation.featureName);
9                  end
10             end
11         end
12     end
13 end
```

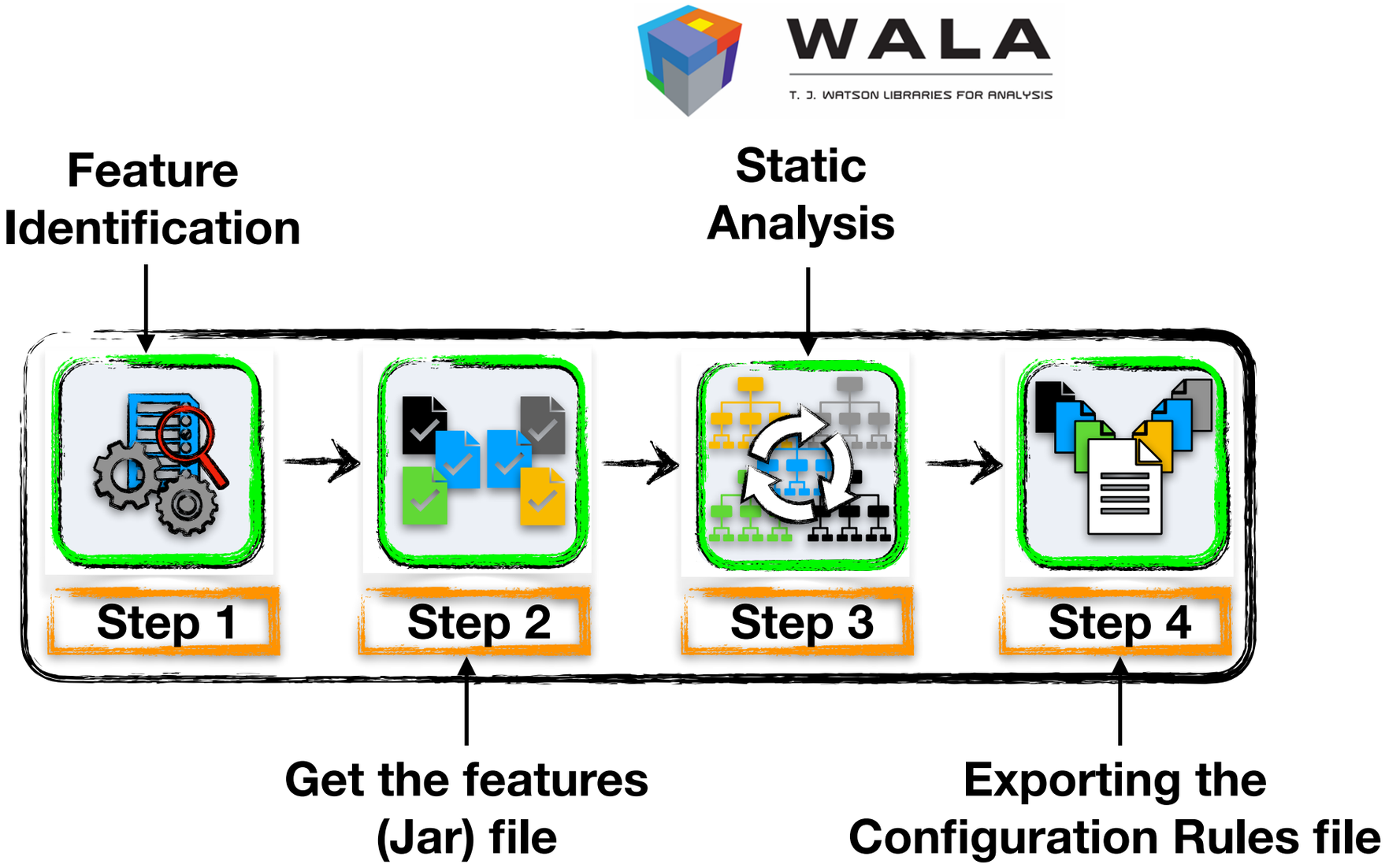
Behavioral Map

Framework Architecture



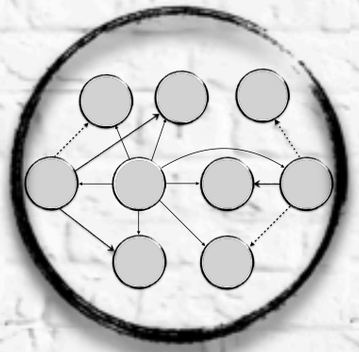
Behavioral Map

Data Extractor Process

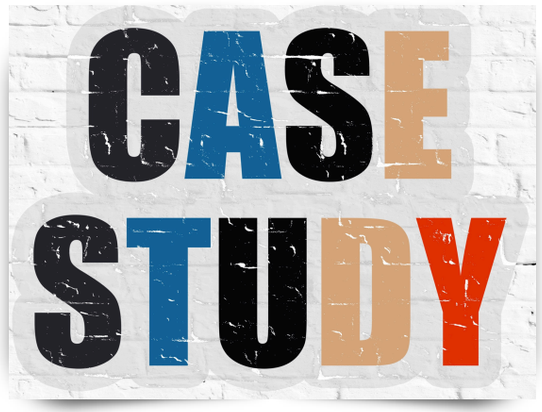


The Behavioral Map was implemented using neo4j

Behavioral Map



CASE STUDY



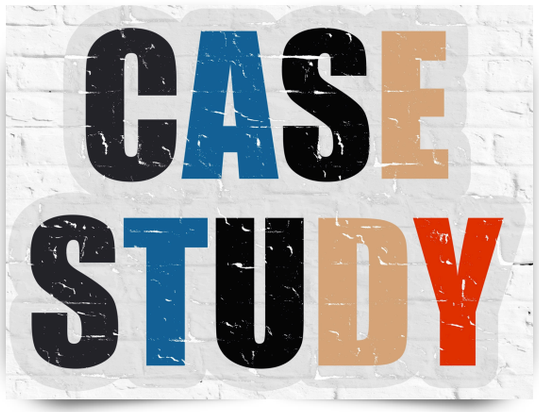
Architectural Smell List

Smell Name	Detection
Cyclic Dependency (CD) [16]	Full
Extraneous Connector (EC) [8]	Full
Hub-Like Dependency (HL) [16, 10]	Full
Oppressed Monitors (OM)[11]	Partial

C. Raibulet, F. A. Fontana, S. Carettoni, **A preliminary analysis of self-adaptive systems according to different issues**, Software Quality Journal (2020) 1–31.

M. A. Serikawa, A. d. S. Landi, B. R. Siqueira, R. S. Costa, F. C. Ferrari, R. Menotti, V. V. De Camargo, **Towards the characterization of monitor smells in adaptive systems,** in: X Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS), IEEE, 2016, pp. 51–60.

Behavioral Map



The SHE Smart Home System

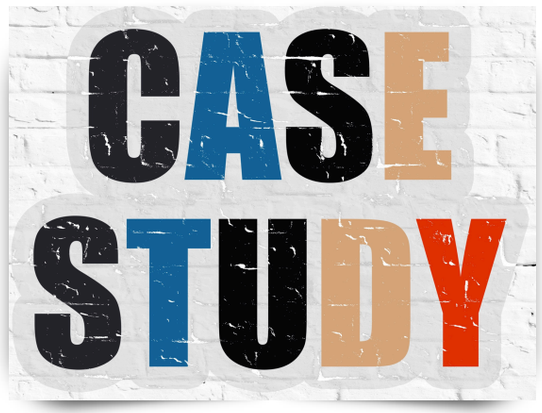
- We applied the **BM** framework on SHE smart home system based on MAPE-K loop and implemented using the publish-subscribe architecture.

- The SHE is composed by:

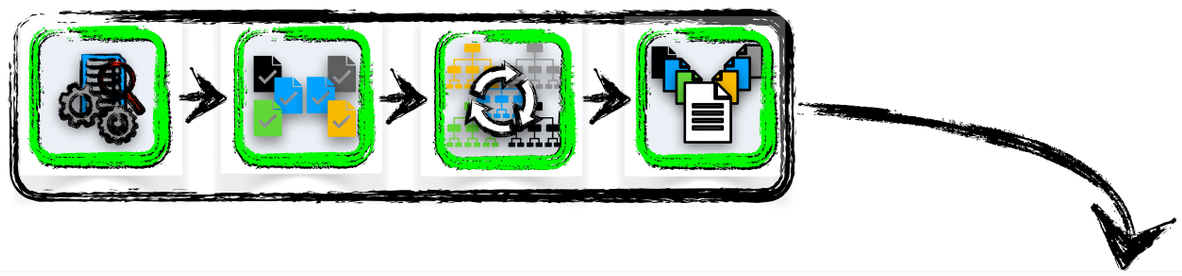
  - **Core features:** Manager, Listener, Loader, Installer, and Presentation Layer.
  - We included **four optional features** as follows:

    - **Luminosity:** used to read data from the luminosity sensor;
    - **Presence:** used to read data from the presence sensor;
    - **lampController:** responsible for controlling Lamp feature's behavior using the information read from Luminosity and Presence features;
    - **Lamp:** an actuator used to switch on and off lights based on the lampController feature's data.

Case study available in: https://github.com/edilton-santos/BehavioralMapExample

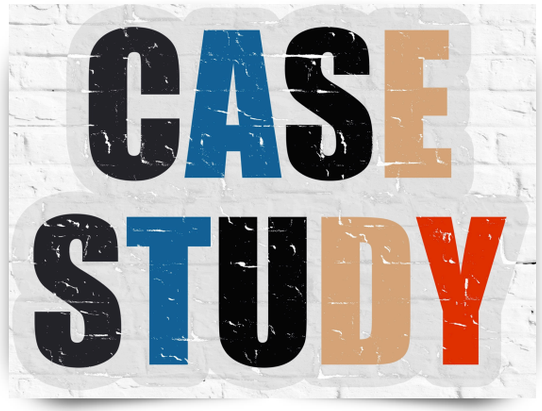


Running the Data Extractor




```
{
"imported_packages":["com.she.core.listener"],
"relationships":[
    {"relationship_type":"Requires","feature_name":"listener"},
    {"relationship_type":"Reads","feature_name":"lampController"},
    {"relationship_type":"Reads","feature_name":"luminosity"},
    {"relationship_type":"Reads","feature_name":"presence"},
    {"relationship_type":"Reads","feature_name":"lamp"}
   ],
"name":"presentation layer",
"type":"Core",
"version":"1.0.0",
"status":"Ativo",
"exported_packages":["com.she.core.presentation.layer","com.she.core.presentation.view"]
}
```

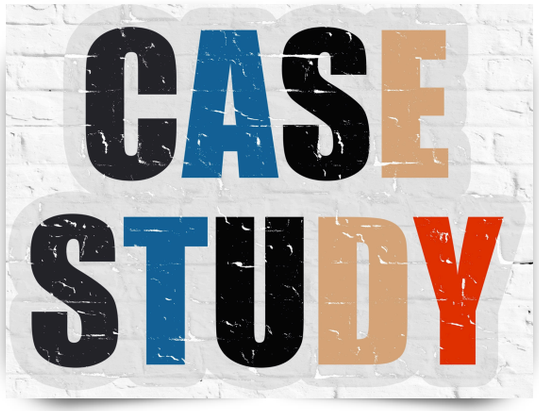

Behavioral Map



Identifying Architectural Bad Smells

Feature Types	Core(5) Feature(9) Controller(1) Sensor(2) Actuator(1)
Relationships	Requires(11) Reads(6) Controls(1)

Behavioral Map

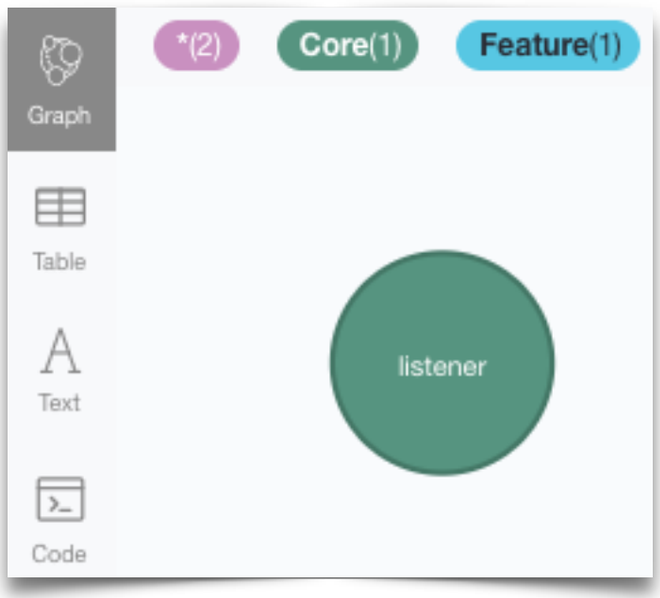


Identifying Architectural Bad Smells

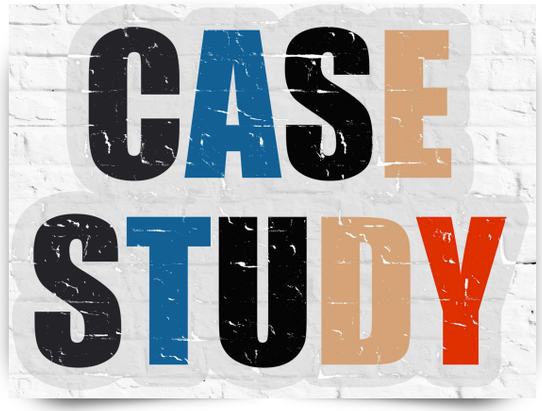
Cypher query:

```
// Hub-Like Dependency
MATCH (f:Feature)-[r:Requires]→(f2:Feature)
OPTIONAL MATCH (f2)-[r2:Requires]→(:Feature)
WITH f2, count(r) As Rtotal, count(r2) As Rtotal2
WHERE Rtotal ⩾ 5 and Rtotal > Rtotal2
RETURN f2, Rtotal, Rtotal2
```

Result:



Behavioral Map

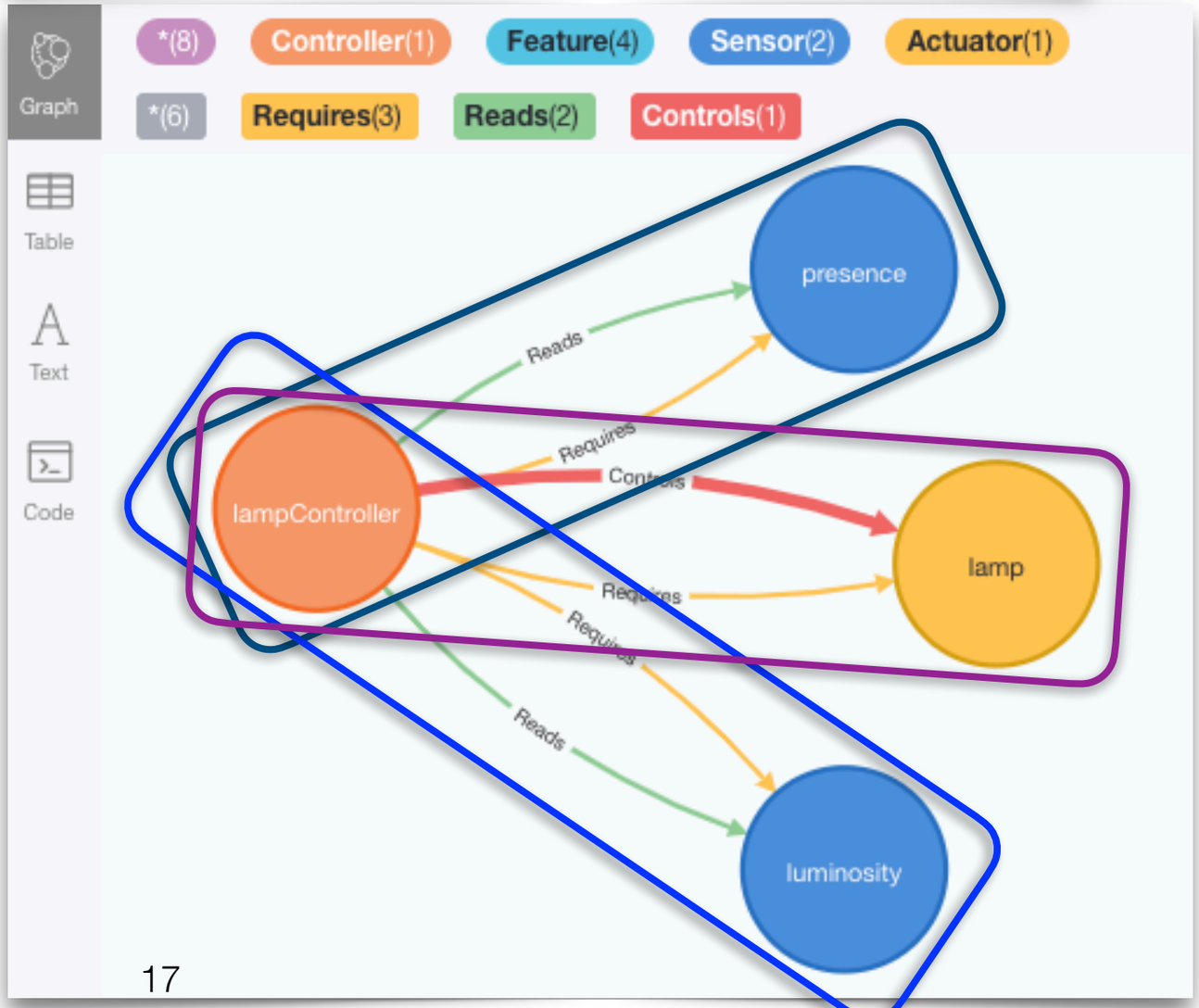


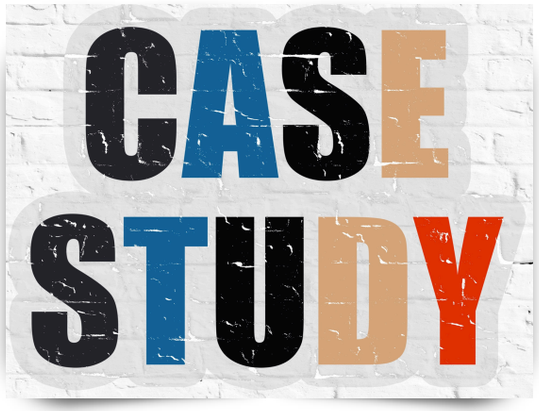
Identifying Architectural Bad Smells

Cypher query:

```
// Extraneous Connector
MATCH (f:Feature)-[r:Requires]→(f2:Feature)
WHERE exists((f)-[:Reads|Controls]→(f2))
RETURN f, r, f2
```

Result:



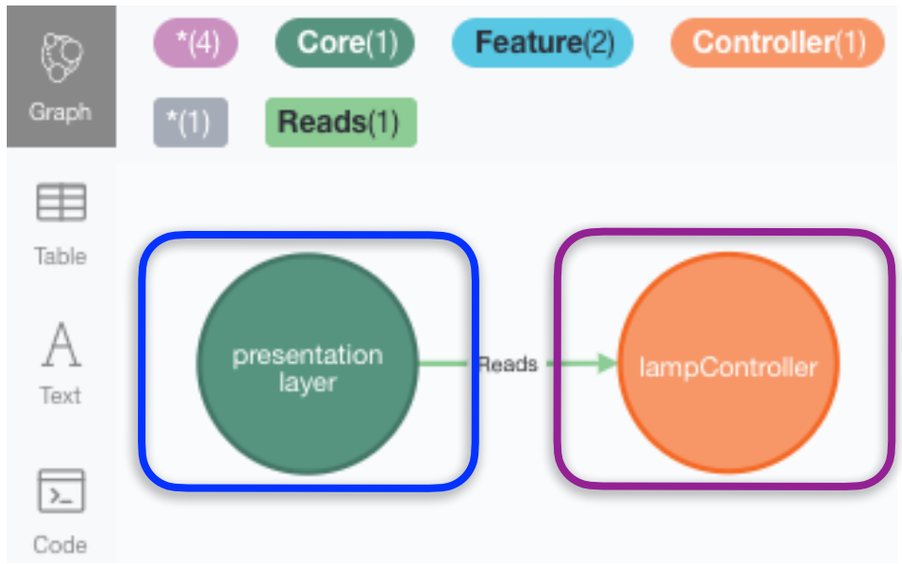


Identifying Architectural Bad Smells

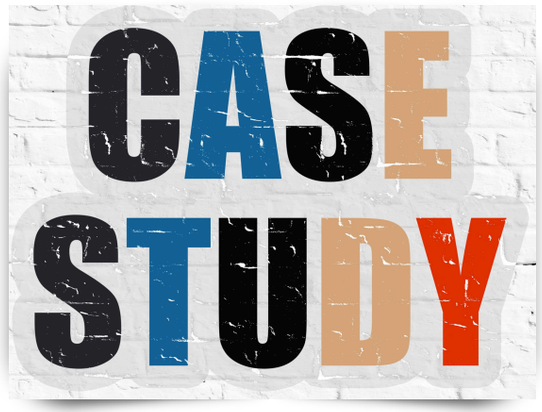
Cypher query:

```
// Look for Oppressed Monitors
MATCH (f1:Feature)-[r:Reads]→(:Feature)
WITH f1, count(r) As Rtotal
WHERE Rtotal ⩾ 2
RETURN f1, Rtotal
```

Result:



Behavioral Map

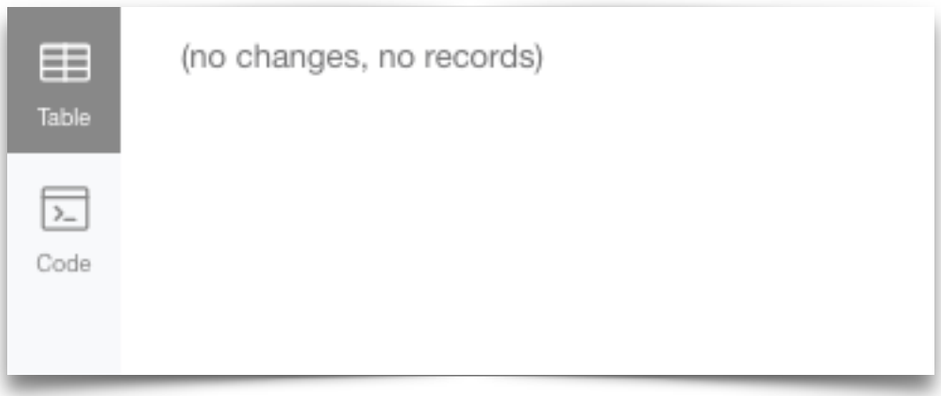


Identifying Architectural Bad Smells

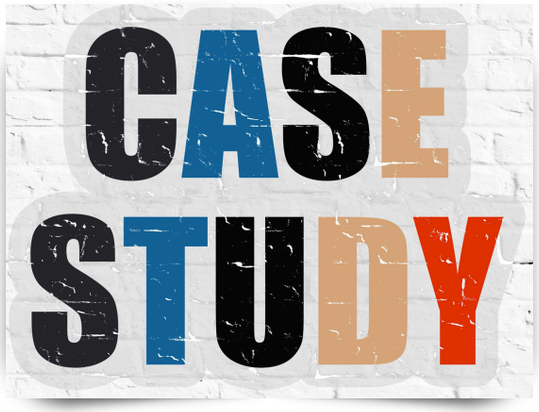
Cypher query:

```
// Cyclic Dependency
MATCH (f:Feature)-[:Requires]→(f2:Feature)-[:Requires]→(f)
OPTIONAL MATCH (f2)-[:Requires]→(f3:Feature)-[:Requires]→(f)
RETURN f, f2, f3
```

Result:

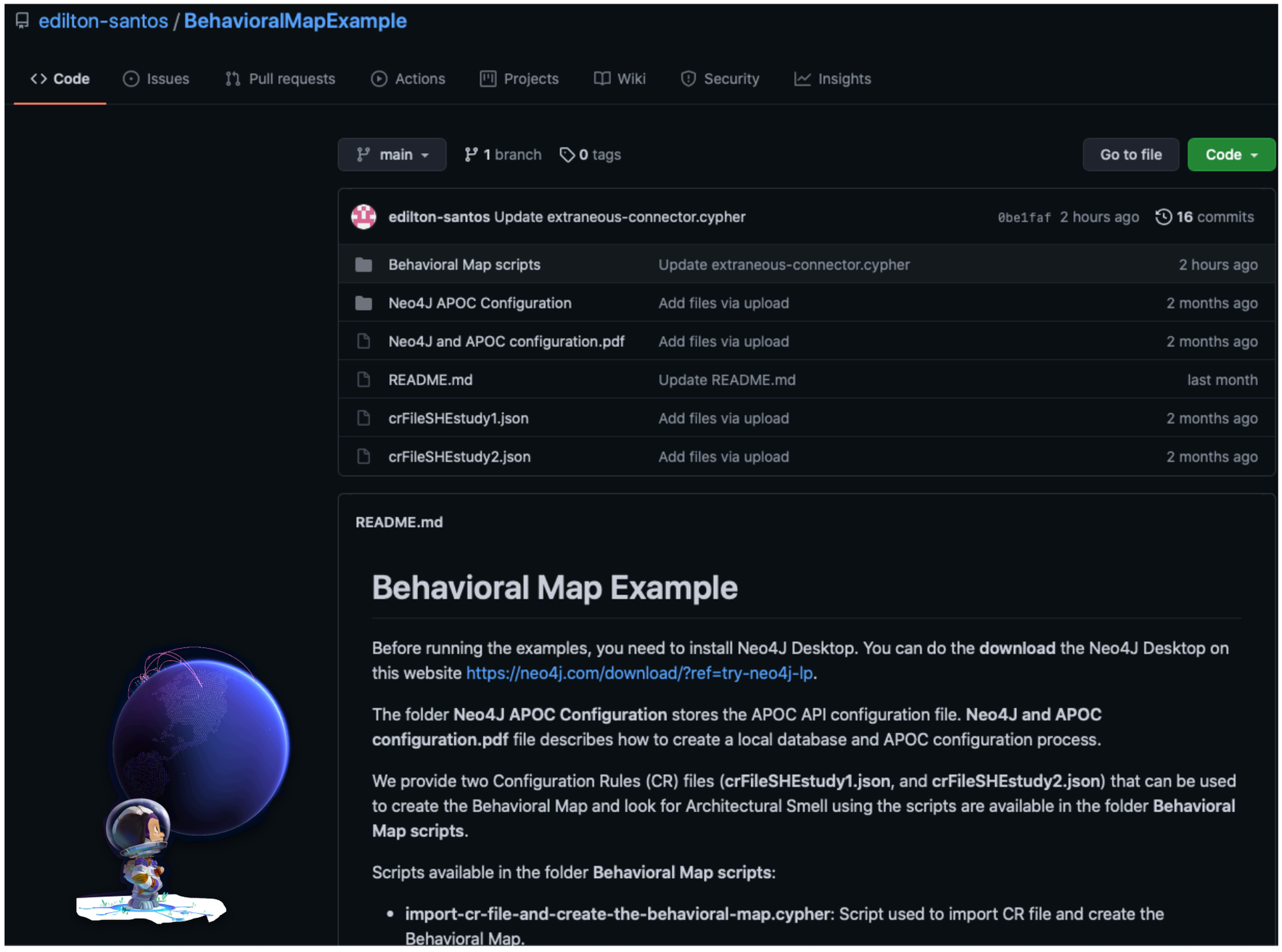


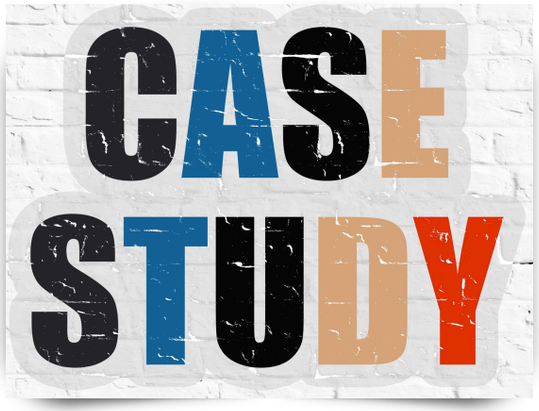
Behavioral Map



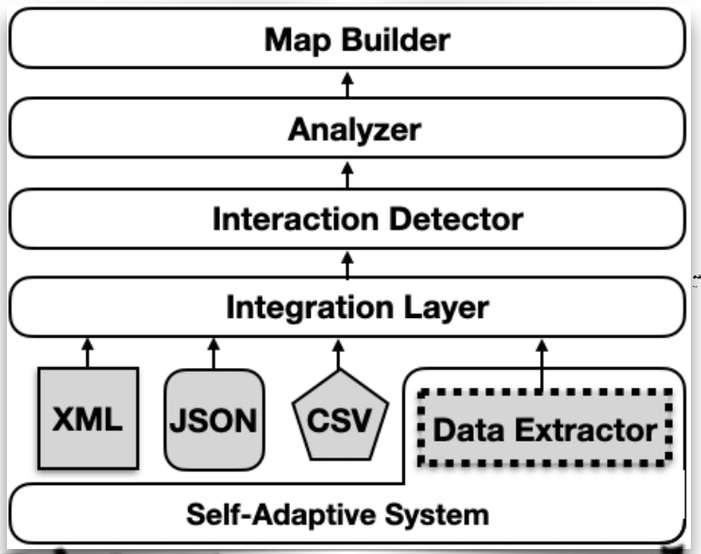
Identifying Architectural Bad Smells

https://github.com/edilton-santos/BehavioralMapExample

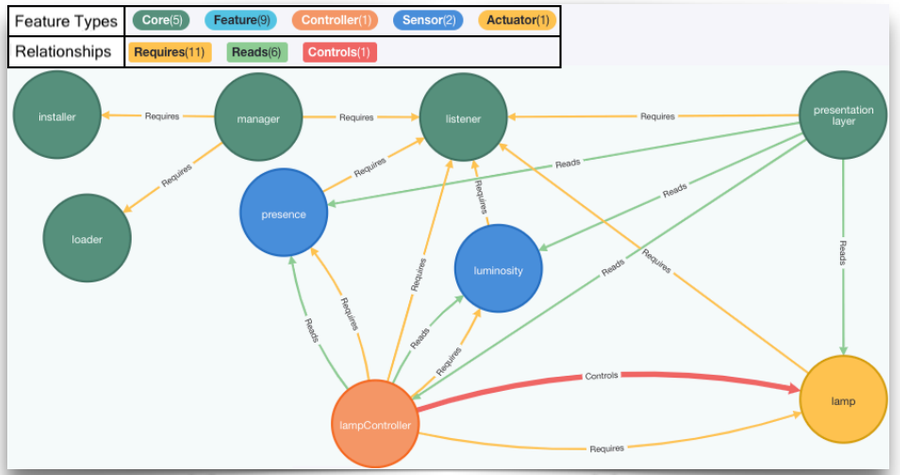




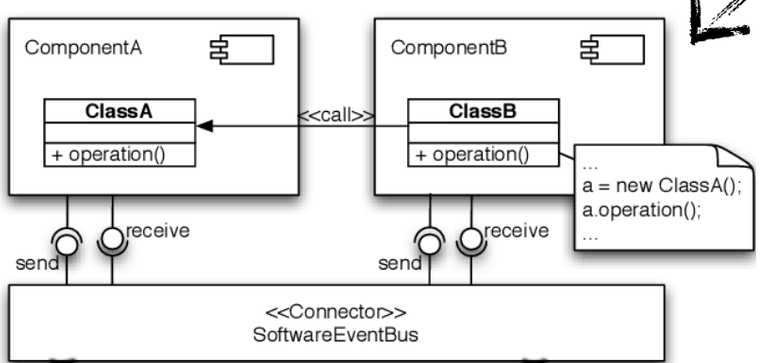
Conclusion



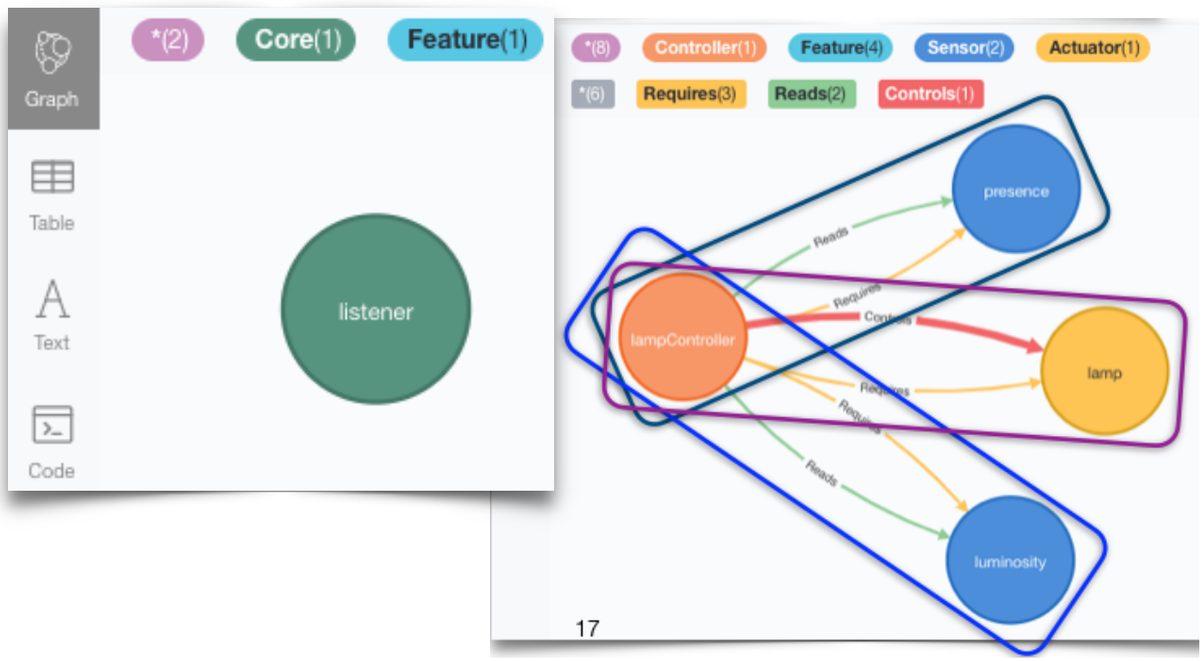
BM Framework



Behavioral Map



Architectural Smell




ДЗЯКУЙ
ESKERRIK ASKO
KÖSZÖNÖM
БЛАГОДАРЯ
谢谢
OBRIGADO
THANK YOU
DANKIE
CẢM ƠN BẠN
GRACIAS
MERCI
KIITOS
ありがとう
شكرا
DĚKUJI
DANKE
Ви благодариме
GRAZIE
GRATIAS AGIMUS TIBI
მადლობა
спасибо
TAKK
GRÀCIES
DIOLCH
DZIĘKUJĘ
TAK
FALEMINDERIT

A Vision to identify Architectural Smells in Self-Adaptive Systems using Behavioral Maps

Edilton Lima dos Santos
edilton.limados@unamur.be
Sophie Fortz
Gilles Perrouin
Pierre-Yves Schobbens

CASA@ECSA 2021, Växjö, Sweden, September 13th - 14th, 2021.