Feature Location in Model Driven Software Engineering: Industrial Experiences

Feature location is a key activity to reengineer a set of products into a Product Line. In the context of Model Driven Software Engineering, models are the cornerstone artefact where feature location must be performed. In this talk, we are going to go through the efforts performed at two industrial case studies (Induction Hobs of BSH Group, and Train Control & Management Software of CAF) to achieve feature location in models. These feature location efforts range from Information Retrieval to Machine Learning, and include the dimension of Search-based Software Engineering. Results are not perfect, but we are going to discuss if they are up to the task of Product Line Reengineering.
Hi!, I am Carlos Cetina

PhD in Computer Science (Universidad Politécnica de Valencia)
Associate Professor (Universidad San Jorge)
Head of a young research group: svit.usj.es

My background is in Model Driven Engineering and Software Product Lines.
Models abstract from implementation details and Systematic reuse

Intersection: Reuse of model fragments

Before reusing model fragments, we must find them.

I am new to SBSE, but it is helping me to find the model fragments in industrial environments:

Past: Home automation, KNX, B/S/H/ Induction Hobs
Present: CAF, Train Control & Management
Future: Bosch, BOSCH, teltronic, Integrated command and control
This is about: **Locating Model Fragments on Models**

We locate model fragments to reuse them as features in other products.
Locating Model Fragments on Models

Models: abstract specification of a part of a software system.

Popular modelling languages (among others):

- UML Class Diagram
- BPMN
- Entity Relationship
- Grafcet
- Ladder
- Simulink
- Textual modeling
Locating Model Fragments on Models
Locating Model Fragments on Models: Industrial Experiences

Induction Hobs of B/S/H/
(produced under the Bosch, Siemens, Balay, Neff, Gaggenau brands, among others)

Rolling stock of CAF
(Trains, Trams, High-speed, and Underground)

Both:
• SW development is not their main activity, but their products are intense on SW:
  - main source of new functionality,
  - and SW compensates for HW.
Family of products
Family of products
Family of products
Family of products
Family of products
Family of products
Family of products
Family of products
Family of products
Family of products
Domain Specific Language Model for Induction Hobs: IHDSL
(Basic concepts)
Model Driven Engineering

IHDSL Model

M2T

Induction Hob firmware
Starting point: variability is not formalized.
Goal: Locating Features (Model Fragments) on Models
Less noise:
Models are less bound to the underlying implementation details and are much closer to the problem domain.

Metamodel helps:
Metamodel of DSL encode domain knowledge. Inductor in IHDSL as oppose to Class in UML or in Java grammar.

From code to models

IR techniques and models?
Intuition: as text documents. But models are not text only. But models have model and metamodel.

ML and models?
Feature encoding is the first challenge.
Locating Features (Model Fragments) on Models

Our journey
“Blood, sweat and tears” Approach

- From focus groups to discussions (you name it).
- Eventually you identify reusable model fragments 😊
- BUT, extracting all model fragments takes more time than is available.
“Machine replaces humans” Approach

- Mechanical model comparisons.
- Time is not a problem anymore.
- BUT, results.....

Building up on:
Xiaorui Zhang, Øystein Haugen, Birger Møller-Pedersen: 
Model Comparison to Synthesize a Model-Driven Software Product Line. 
SPLC 2011: 90-99
Results…

Comparisons

Feature 1

Why not?

University

Feature 2
Results... are not for university, they are for industry.
“Machine and human complement each other” Approach

- Human has domain knowledge.
- Machine capability to search very large problem spaces.
Domain experts guide the search providing its knowledge about the feature:

**<<Feature Knowledge>>**

**Hotplate:** group of inductors that can work in conjunction to heat the cookware. Each hotplate is controlled by a power level that is then translated to different power outputs for each inductor depending on their size and position. Inductors activated depend on the detection of cookware.

**Fragment Seed**

**Textual Description**
<table>
<thead>
<tr>
<th>#</th>
<th>Model Fragment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MF2</td>
</tr>
<tr>
<td>2</td>
<td>MF1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>n</td>
<td>MF5</td>
</tr>
</tbody>
</table>
### Model Fragments

<table>
<thead>
<tr>
<th>Terms</th>
<th>MF1</th>
<th>MF2</th>
<th>MF3</th>
<th>MF4</th>
<th>MF5</th>
<th>MF6</th>
<th>MF7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inverter</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Provider</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Power</td>
<td>0</td>
<td>4</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Inductor</td>
<td>0</td>
<td>10</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### Query

- Inverter: 0
- Provider: 0
- Power: 2
- Inductor: 0

### LSA Results

The LSA results are visualized in a 3D space with vectors MF1, MF2, and MF3.

### Fitness

\[
\text{fitness}(p_1) = \cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|}
\]
FLiMEA produces a ranking of feature realizations matching the feature description provided.

The feature realizations are ordered based on the fitness calculated.

The domain expert can decide which one fits better to their understanding.
Locating Features (Model Fragments) on Models

Our journey

Human (only)  Computer (only)  EVO+IR

Result?
“Machine and human complement each other” Approach

- Human has domain knowledge.
- Machine capability to search very large problem spaces.
- Result: Library of model fragments.
“Classic” models…
“Classic” models… but this is all about model fragments…
This is useful here:
Øystein Haugen, Andrzej Wasowski, Krzysztof Czarnecki: CVL: common variability language. SPLC 2013: 277
You can see this in action:

http://carloscetina.com/variabilitytool.htm
Now, models are in SPL department.
Outperforms baselines...

but never ever 100% recall and 100% precision!
Jane Cleland-Huang - Towards Effective Software and Systems Traceability

https://www.youtube.com/watch?v=1C3Di2iNh0&t=3179s
Locating Features (Model Fragments) on Models

Our journey
The system will turn on the LED of the button that closes the doors on one side of the train if all the doors of the correspondent coupling are closed or blocked.
Locating Features (Model Fragments) on Models

Our journey

- Human (only)
- Computer (only)
- EVO+IR
- Dynamic Analysis
- EVO+ML
Dynamic Analysis

Context of the induction hob

Reconfigurations of the induction hob

R1: `newChannelForDoubleInductor(upperPowerGroup, secondaryInductor)`
R2: `redirectPowerFromInverter(secondInverter, upperPowerGroup)`

Scenario: 1. Dynamic Analysis → Model Trace → Model Retrieval in the Model Trace → Most Relevant Model → Information Retrieval in the Model → Ranked Model Elements
Locating Features (Model Fragments) on Models

Our journey

Human (only) -> Computer (only) -> EVO+IR -> Dynamic Analysis

Reformulations

EVO+ML
<table>
<thead>
<tr>
<th>Author</th>
<th>Base query</th>
<th>Relevant documents</th>
<th>k Value</th>
<th>Industrial domain</th>
<th>Artifact</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Rivas et al., 2014]</td>
<td>User</td>
<td>Biomedical articles</td>
<td>10</td>
<td>No</td>
<td>Text</td>
</tr>
<tr>
<td>[Hill et al., 2009]</td>
<td>Developer</td>
<td>Source code</td>
<td>LoC</td>
<td>No</td>
<td>Code</td>
</tr>
<tr>
<td>[Lu et al., 2015]</td>
<td>Developer</td>
<td>Internet site</td>
<td>20</td>
<td>No</td>
<td>Code</td>
</tr>
<tr>
<td>[Marcus et al., 2004]</td>
<td>User</td>
<td>Source code</td>
<td>5</td>
<td>No</td>
<td>Code</td>
</tr>
<tr>
<td>[Gay et al., 2009]</td>
<td>Developer</td>
<td>Source code</td>
<td>1, 3, and 5</td>
<td>No</td>
<td>Code</td>
</tr>
<tr>
<td>[Haiduc et al., 2013]</td>
<td>User</td>
<td>Source code</td>
<td>5</td>
<td>No</td>
<td>Code</td>
</tr>
<tr>
<td>[Dumitru et al., 2011]</td>
<td>User</td>
<td>Internet sites</td>
<td>25</td>
<td>No</td>
<td>Product specifications</td>
</tr>
<tr>
<td>[Tian et al., 2014]</td>
<td>User</td>
<td>Internet site</td>
<td>10,000</td>
<td>No</td>
<td>Text and code</td>
</tr>
<tr>
<td>[Dietrich et al., 2013]</td>
<td>Analyst</td>
<td>Requirement traces</td>
<td>-</td>
<td>No</td>
<td>Code and documents</td>
</tr>
<tr>
<td>[Lv et al., 2015]</td>
<td>User</td>
<td>Online documentation</td>
<td>10</td>
<td>No</td>
<td>Code</td>
</tr>
<tr>
<td><strong>Our work</strong></td>
<td>Domain expert</td>
<td>Domain experts</td>
<td>3</td>
<td>Yes</td>
<td>Models</td>
</tr>
</tbody>
</table>
Locating Features (Model Fragments) on Models

Our journey

- Human (only)
- Computer (only)
- EVO+IR
- Dynamic Analysis
- Reformulations
- Tuning NLP & EVO

EVO+ML
Ballarin, Manuel; Marcen, Ana; Pelechano, Vicente; Cetina, Carlos; Measures to report the Location Problem of Model Fragment Location. MoDELS. 2019
There is a lot of hard and exciting work to do. If you want to join us in this quest, just let me know.
Thanks!
More info:

Font, Jaime; Arcega, Lorena; Haugen, Øystein; Cetina, Carlos; Achieving Feature Location in Families of Models through the use of Search-Based Software Engineering. IEEE Transactions on Evolutionary computation. 2018

Marcén, Ana Cristina; Pérez, Francisca; Cetina, Carlos; Ontological Evolutionary Encoding to Bridge Machine Learning and Conceptual Models: Approach and Industrial Evaluation. 36th International Conference on Conceptual Modeling. 2017

Pérez, Francisca; Marcén, Ana Cristina; Lapeña, Raúl; Cetina, Carlos; Introducing Collaboration for Locating Features in Models: Approach and Industrial Evaluation. 25th International Conference on Cooperative Information Systems. 2017