Appendix

Jude Gyimah*, Henriette Knopp*, Thorsten Berger*, Patrizio Pelliccione†

I. Denotational Semantics

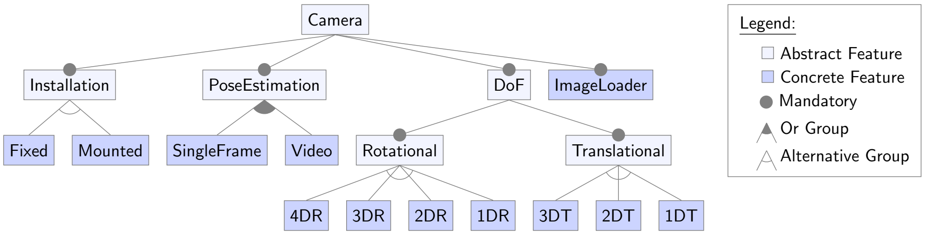

Fig. 1: running example: camera subsystem

Let ID be the set of feature names in the form of string identifiers, $\mathsf{Group} = \{\mathsf{OR}, \mathsf{XOR}\}$ the type of feature groups supported by a feature model, and $\mathsf{BTime} = \{\text{Early, Late}\}$ the supported binding times and $\mathsf{BMode} = \{\text{Static, Dynamic}\}$ the supported binding modes.

Now, let FM be the set of all possible feature models, where each model $fm \in \mathsf{FM}$ is a set of features $\mathcal{F}$, so $\mathsf{FM} = P(\mathcal{F})$, where[1]:

$$\mathcal{F} = \mathsf{ID} \times \lceil \mathsf{ID} \rceil \times \lfloor \mathsf{Group} \rfloor \times \mathbb{B} \times \mathcal{P}(\mathsf{BTime})\backslash\{\emptyset\} \times \mathcal{P}(\mathsf{BMode})\backslash\{\emptyset\} \times \lfloor P(\mathsf{ID}) \rfloor \times \lfloor P(\mathsf{ID} \rfloor)$$

A feature $f \in \mathcal{F}$ is characterized by its attributes, denoted as a tuple. If $f = (n, p, o, g, t, m, i, e)$, then n is the name, p is the parent of the feature with $\top$ for the root element, o determines whether a feature is optional (True) or mandatory (False), t and m are the allowed binding time and the allowed binding mode, i are cross-tree inclusion constraints, and e are cross-tree exclusion constraints.

As an alternative way to look at the definition of the feature model we also provide the abstract syntax in the for of a meta-model in Figure 2.

Semantic Domain: The configuration of a feature model $fm \in \mathsf{FM}$ is an assignment of configuration triple δ to all features $f \in fm$. Thus, the set of all possible configurations is defined as

$$\mathsf{Confs} : \mathsf{Id} \longrightarrow (\mathbb{B} \times \mathsf{BTime} \times \mathsf{BMode})$$

The configuration triple $\delta \in \mathsf{Configs}, \delta(n)$ consists of the selection status, binding time, and binding mode. A valid configuration of the modelled DoF feature could be: $\delta(DoF) = (\text{True, Early, Static})$, with $\delta(DoF)_{\text{sel}} = \text{True}$, $\delta(DoF)_{\text{tim}} = \text{Early}$, and $\delta(DoF)_{\text{mod}} = \text{Static}$.

```
{ "name": "DoF",
  "group": "",
  "isOptional": false,
  "includes": "",
  "excludes": "",
  "bindingTimeAllowed": "Early",
  "bindingModeAllowed": "Static" }
```

Listing 1: DoF feature concrete syntax

```
{ "name": "DoF",
  "isSelected": true,
  "bindingMode": "Static",
  "bindingTime": "Early" }
```

Listing 2: DoF configuration concrete syntax

We define the semantic function $[\![\cdot]\!]_{FM}$ as

$$[\![\cdot]\!]_{FM} : \mathsf{FM} \longrightarrow \mathcal{P}(\mathsf{Confs})$$

* Ruhr University, Bochum, Germany
† Gran Sasso Science Institute (GSSI), L'Aquila, Italy

[1] Hint: $\lceil X \rceil = X \cup \{\top\}$and $\lceil X \rceil = X \cup \{\bot\}$

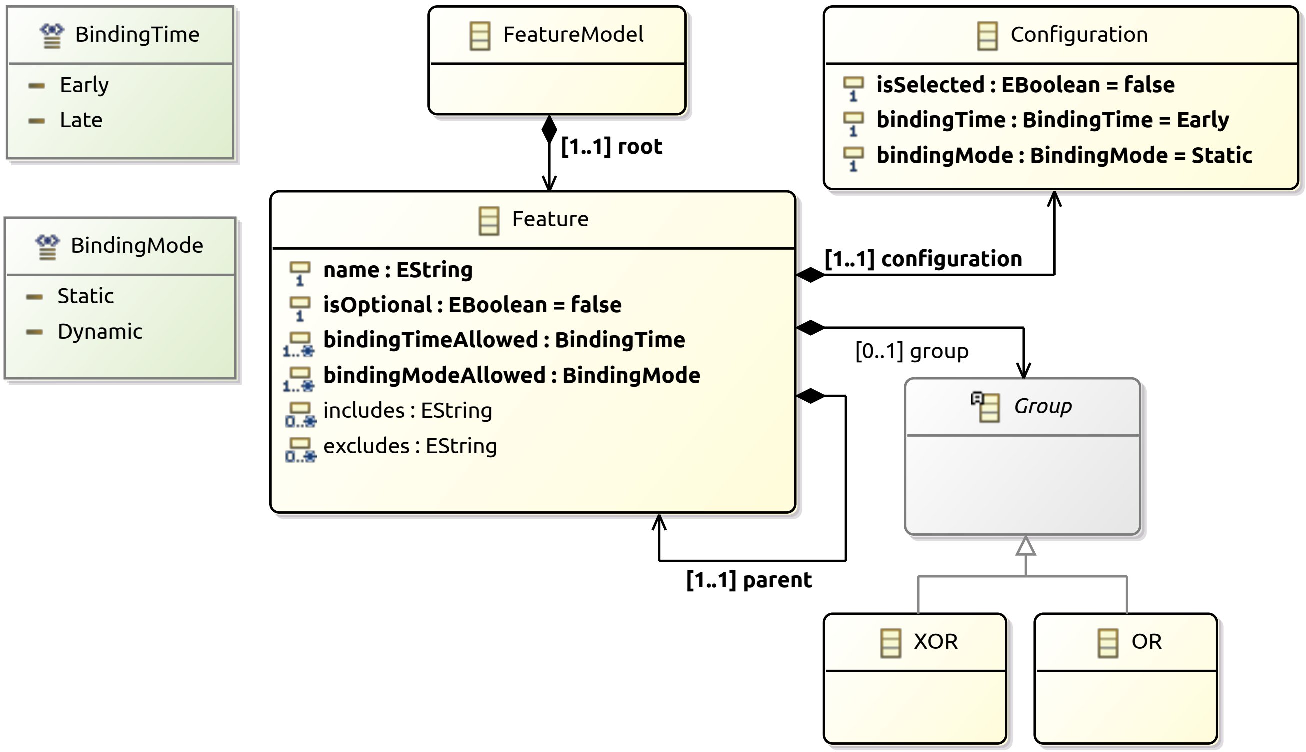

Fig. 2: Abstract Syntax of Feature and Feature Configuration

The semantic function evaluates a feature model $fm \in FM$ and returns all configurations Confs from $P(\mathsf{Confs})$.

Helper Functions Before going further in details on the semantics we first introduce some functions that are helpful in the following definitions. To define the rest of the semantics we rely on the relationships between features characterized in the form of requires and exclusion criteria.

If a feature X requires a feature Y this means that when feature X is present in the system feature Y needs to be present as well. We define:

$$\text{req}(\delta(n),\delta(m)) = \begin{cases} \text{true if } \delta(n)_{\text{sel}} = \text{False} \\ \text{true if } \delta(n)_{\text{sel}} = \text{True and } \delta(m)_{\text{mod}} = \text{Static and } \delta(m)_{\text{sel}} = \text{ True and if } \delta(n)_{\text{tim}} = \text{Late} \\ \text{true if } \delta(n)_{\text{sel}} = \text{True and } \delta_{\text{mod}}(m) = \text{Static and } \delta(m)_{\text{sel}} = \text{ True} \\ \qquad \text{and if } \delta(n)_{\text{tim}} = \text{Early and } \delta(m)_{\text{tim}} = \text{Early} \\ \text{false otherwise} \end{cases}$$

If two features X and Y exclude each other this means, that they can not be active at the same time. Thus, we define:

$$\text{excl}(\delta(n),\delta(m)) = \begin{cases} \text{true if } \delta(n)_{\text{sel}} = \text{False} \\ \text{true if } \delta(n)_{\text{sel}} = \text{True and } \delta(m)_{\text{sel}} = \text{False} \\ \text{false otherwise} \end{cases}$$

Feature Model Semantics: We define our semantic function as an intersection of denotations.

$$[\![fm]\!]_{\mathsf{FM}} = \left(\bigcap_{f\in fm}[\![f]\!]_{\text{timModConsistency}}\right) \cap \left(\bigcap_{f\in fm}[\![f]\!]_{\text{tree}}\right) \cap \left(\bigcap_{f\in fm}[\![f]\!]_{\text{isMandatory}}\right) \cap \left(\bigcap_{f\in fm}[\![f]\!]_{\text{group}}\right) \cap \left(\bigcap_{f\in fm}[\![f]\!]_{\text{excludes}}\right)$$
$$\cap \left(\bigcap_{f\in fm}[\![f]\!]_{\text{includes}}\right) \cap \{\delta \in \mathsf{Configs} \mid \delta(f) = \bot \forall f \in \mathsf{Id}\backslash\mathsf{Id}(fm)\}$$

Valuations for the above denotations are defined as follows:

Model Matching Configuration:

$$\llbracket (n,\_,\_,\_,t,m,\_,\_) \rrbracket_{\text{timModConsistency}} = \left\{ \delta \in \mathsf{Confs} \;\middle|\; \delta(n)_{\text{tim}} \in t \;\wedge\; \delta(n)_{\text{mod}} \in m \right\}$$

For every feature of a model, its configured binding time and mode needs to match the allowed binding time and the allowed binding mode.

Tree Implication:

$$\llbracket (n,p,\_,\_,\_,\_,\_,\_) \rrbracket_{\text{tree}} = \left\{ \delta \in \mathsf{Confs} \;\middle|\; \mathit{req}(\delta(n),\delta(p)) \right\}$$

In the feature model a feature always depends on the feature that is above it in the hirachy (parent feature). Therefore, a feature n requires its parent feature p.

Optional Feature Implication:

$$\llbracket (n,\_,\mathsf{False},\_,t,m,\_,\_) \rrbracket_{\text{isMandatory}} = \left\{ \delta \in \mathsf{Confs} \;\middle|\; \delta(n)_{\text{sel}} = \text{True} \wedge \delta(n)_{\text{mod}} = \text{Static} \right\}$$

If a feature is mandatory this limits its possible configurations. Mandatory features need to be selected throughout the whole execution and therefore need to be static.

Group Feature Selection:

$$\llbracket (n,p,\_,OR,\_,\_,\_,\_) \rrbracket_{\text{group}} = \left\{ \delta \in \mathsf{Configs} \;\middle|\; \bigvee_{(x=(\_,n,\_,\_,\_,\_,\_,\_))\in\mathcal{F}} \delta(x)_{sel} \right\}$$

For a feature that denotes an *OR* group, there needs to be at least one child feature that is selected.

$$\llbracket (n,p,\_,XOR,\_,\_,\_,\_) \rrbracket_{\text{group}} = \llbracket (n,p,\_,OR,\_,\_,\_,\_) \rrbracket_{\text{group}} \cap \left\{ \delta \in \mathsf{Configs} \;\middle|\; \bigwedge_{(x,y=(\_,n,\_,\_,\_,\_,\_,\_))\in\mathcal{F},x\neq y} \text{excl}(\delta(x),\delta(y)) \right\}$$

If two features n_1 and n_2 exist in a configuration, such that they do not reference the same feature, then both features cannot exist simultaneously within the same *XOR* group.

Excludes Constraint:

$$\llbracket (n,\_,\_,\_,\_,\_,\_,x) \rrbracket_{excludes} = \left\{ \delta \in \mathsf{Confs} \;\middle|\; \forall f \in x, \text{excl}(\delta(n),\delta(f)) \right\}$$

All features $m \in x$ need can not be active as long as n is active. We use the helper function excl to evaluate this constraint.

Includes Constraint:

$$\llbracket (n,\_,\_,\_,\_,\_,i,\_) \rrbracket_{includes} = \left\{ \delta \in \mathsf{Confs} \middle| \forall f \in i, \text{req}(\delta(n),\delta(f)) \right\}$$

All features $m \in i$ need to be included if n is active. This means n requires all features in i. We evaluate this with the $includes$ helper function.

For arbitrary features **X** and **Y** in a feature model, that have selection state ($\mathbb{B}$), configured binding time ($\delta(n)_{\text{tim}}$) and configured binding mode ($\delta(n)_{\text{mod}}$), Table I presents the outcomes (i.e. allowed(✓) or disallowed(✗)) of all possible implications (X $\implies$ Y, Y $\implies$ X) and exclusions ($\neg(X \wedge Y)$, $\neg(Y \wedge X)$)

TABLE I: Valid feature configurations after resolving their dependencies in the feature model (requires and excludes).

	X			Y			Dependencies in the Feature Model	
Case	$\delta(X)_{\text{mod}}$	$\delta(X)_{\text{tim}}$	$\delta(X)_{\text{sel}}$	$\delta(Y)_{\text{mod}}$	$\delta(Y)_{\text{tim}}$	$\delta(Y)_{\text{sel}}$	$X \implies Y$ (requires)	$X \implies \neg Y$ (excludes)
1	**Static**	**Early**	**True**	**Static**	**Early**	**True**	✓	✗
2	**Static**	**Early**	**True**	**Static**	**Early**	False	✗	✓
3	**Static**	**Early**	False	**Static**	**Early**	**True**	✓	✓
4	**Static**	**Early**	False	**Static**	**Early**	False	✓	✓
5	**Static**	**Early**	**True**	**Static**	**Late**	**True**	✗	✗
6	**Static**	**Early**	**True**	**Static**	**Late**	False	✗	✓
7	**Static**	**Early**	False	**Static**	**Late**	**True**	✓	✓
8	**Static**	**Early**	False	**Static**	**Late**	False	✓	✓
9	**Static**	**Early**	**True**	**Dynamic**	**Early**	**True**	✗	✗
10	**Static**	**Early**	**True**	**Dynamic**	**Early**	False	✗	✓
11	**Static**	**Early**	False	**Dynamic**	**Early**	**True**	✓	✓
12	**Static**	**Early**	False	**Dynamic**	**Early**	False	✓	✓
13	**Static**	**Early**	**True**	**Dynamic**	**Late**	**True**	✗	✗
14	**Static**	**Early**	**True**	**Dynamic**	**Late**	False	✗	✓
15	**Static**	**Early**	False	**Dynamic**	**Late**	**True**	✓	✓
16	**Static**	**Early**	False	**Dynamic**	**Late**	False	✓	✓
17	**Static**	**Late**	**True**	**Static**	**Early**	**True**	✓	✗
18	**Static**	**Late**	**True**	**Static**	**Early**	False	✗	✓
19	**Static**	**Late**	False	**Static**	**Early**	**True**	✓	✓
20	**Static**	**Late**	False	**Static**	**Early**	False	✓	✓
21	**Static**	**Late**	**True**	**Static**	**Late**	**True**	✓	✗
22	**Static**	**Late**	**True**	**Static**	**Late**	False	✗	✓
23	**Static**	**Late**	False	**Static**	**Late**	**True**	✓	✓
24	**Static**	**Late**	False	**Static**	**Late**	False	✓	✓
25	**Static**	**Late**	**True**	**Dynamic**	**Early**	**True**	✗	✗
26	**Static**	**Late**	**True**	**Dynamic**	**Early**	False	✗	✓
27	**Static**	**Late**	False	**Dynamic**	**Early**	**True**	✓	✓
28	**Static**	**Late**	False	**Dynamic**	**Early**	False	✓	✓
29	**Static**	**Late**	**True**	**Dynamic**	**Late**	**True**	✓	✗
30	**Static**	**Late**	**True**	**Dynamic**	**Late**	False	✗	✓
31	**Static**	**Late**	False	**Dynamic**	**Late**	**True**	✓	✓
32	**Static**	**Late**	False	**Dynamic**	**Late**	False	✓	✓
33	**Dynamic**	**Early**	**True**	**Static**	**Early**	**True**	✓	✗
34	**Dynamic**	**Early**	**True**	**Static**	**Early**	False	✗	✓
35	**Dynamic**	**Early**	False	**Static**	**Early**	**True**	✓	✓
36	**Dynamic**	**Early**	False	**Static**	**Early**	False	✓	✓
37	**Dynamic**	**Early**	**True**	**Static**	**Late**	**True**	✗	✗
38	**Dynamic**	**Early**	**True**	**Static**	**Late**	False	✗	✓
39	**Dynamic**	**Early**	False	**Static**	**Late**	**True**	✓	✓
40	**Dynamic**	**Early**	False	**Static**	**Late**	False	✓	✓
41	**Dynamic**	**Early**	**True**	**Dynamic**	**Early**	**True**	✗	✗
42	**Dynamic**	**Early**	**True**	**Dynamic**	**Early**	False	✗	✓
43	**Dynamic**	**Early**	False	**Dynamic**	**Early**	**True**	✓	✓
44	**Dynamic**	**Early**	False	**Dynamic**	**Early**	False	✓	✓
45	**Dynamic**	**Early**	**True**	**Dynamic**	**Late**	**True**	✗	✗
46	**Dynamic**	**Early**	**True**	**Dynamic**	**Late**	False	✗	✓
47	**Dynamic**	**Early**	False	**Dynamic**	**Late**	**True**	✓	✓
48	**Dynamic**	**Early**	False	**Dynamic**	**Late**	False	✓	✓
49	**Dynamic**	**Late**	**True**	**Static**	**Early**	**True**	✓	✗
50	**Dynamic**	**Late**	**True**	**Static**	**Early**	False	✗	✓
51	**Dynamic**	**Late**	False	**Static**	**Early**	**True**	✓	✓
52	**Dynamic**	**Late**	False	**Static**	**Early**	False	✓	✓
53	**Dynamic**	**Late**	**True**	**Static**	**Late**	**True**	✓	✗
54	**Dynamic**	**Late**	**True**	**Static**	**Late**	False	✗	✓
55	**Dynamic**	**Late**	False	**Static**	**Late**	**True**	✓	✓
56	**Dynamic**	**Late**	False	**Static**	**Late**	False	✓	✓
57	**Dynamic**	**Late**	**True**	**Dynamic**	**Early**	**True**	✗	✗
58	**Dynamic**	**Late**	**True**	**Dynamic**	**Early**	False	✗	✓
59	**Dynamic**	**Late**	False	**Dynamic**	**Early**	**True**	✓	✓
60	**Dynamic**	**Late**	False	**Dynamic**	**Early**	False	✓	✓
61	**Dynamic**	**Late**	**True**	**Dynamic**	**Late**	**True**	✗	✗
62	**Dynamic**	**Late**	**True**	**Dynamic**	**Late**	False	✗	✓
63	**Dynamic**	**Late**	False	**Dynamic**	**Late**	**True**	✓	✓
64	**Dynamic**	**Late**	False	**Dynamic**	**Late**	False	✓	✓