# A DL Semantics for Reasoning over OVM-based Variability Models

Germán Braun[1,3], Matias Pol'la[1,3], Laura Cecchi[1], Agustina Buccella[1,3], Pablo Fillottrani[2,4], and Alejandra Cechich[1]

[1]UNIVERSIDAD NACIONAL DEL COMAHUE [2]UNIVERSIDAD NACIONAL DEL SUR
[3]*Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)*
[4]*Comisión de Investigaciones Científicas de la Provincia de Buenos Aires (CIC)*

**Abstract** Software Product Line (SPL) development has traditionally included Variability Management as a way of defining, modelling, implementing and testing variability. In this context, we have created a framework, SeVaTax, based on extensions of the Orthogonal Variability Model (OVM), and aimed at analysing properties of variability models and deriving products from an SPL. Despite several approaches have proposed techniques for automatically analysing the variability models' potential for deriving products, a formalisation of OVM-based diagrams has not been addressed yet. In this paper, we formally define the syntax and semantics of SeVaTax diagrams expressed in first order logic, introduce a DL encoding and establish the EXPTIME-membership of reasoning over SeVaTax. Based on these results, we are developing a prototype providing a back-end for the automated support of consistency checks of SeVaTax variability diagrams along with its encoding into DL knowledge bases.

## 1 Introduction and Motivation

Variability management [22] is one the most challenging and important activities of the *domain* and *application engineering* phases of an SPL development. During the domain engineering, variability must be defined, modelled, implemented, and validated taking into account domain constraints and the flexibility the SPL should provide. Then, during the application engineering, this variability is instantiated in order to derive new products from the SPL. These products must consider the application requirements, which must be validated according to specific combinations of variability constraints imposed by the model. Particularly, a new research topic has emerged for this validation as *automated variability analysis*, which focuses on a set of techniques for translating and validating variability models by considering the anomalies or mismatches these models may contain [16, 20, 27, 3]. On this line, checking consistency of variability models (VM) is still a critical problem. In previous works, we have introduced a process for implementing variability analysis [5, 6]; and we have defined a framework, SeVaTax [23], which includes selecting variants according to the needs and characteristics of new products, and assembling software components for those products.

In the literature, several works propose different techniques and methods for the analysis process, many of them focused on the analysis of Feature Models (FMs) [14] or OVMs but using back-end SAT solvers [3, 31, 20, 32, 18, 13, 28, 19]. In this respect,

SAT-based approaches inherit some limitations from more restrictive logics and thus failing to reflect the finer logical structure of variability models. Particularly, SAT-based proposals miss the relationships amongst services when embedding variability models in CNF. As a consequence, the domain validation process involves several product-by-product SAT checks, i.e. setting CNF input formulae according to each possible instantiation derived from the selection of services in the domain model. Therefore, identifying inconsistency sources is hard in such approaches as has also been evaluated in [23]. Aiming at addressing these drawbacks and improving the variability analysis process, we consider the well-known benefits of using DL to conceptual representation in terms of classes and relationships admitting decidable reasoning. Even though it seems to fit with our requirements, a formalisation of OVM-like diagrams is still missing and therefore validations in existing approaches are not reliable enough either.

Based on this motivation, in this work we propose the formalisation of SeVaTax diagrams expressed in first order logic, and introduce an $\mathcal{ALCI}$ encoding allowing extended reasoning capabilities. We also conclude that reasoning on SeVaTax diagrams is EXPTIME w.r.t. service consistency. From these results, we are developing a prototype providing a back-end for the automated support of consistency checks of SeVaTax variability diagrams along with its encoding into DL knowledge bases.

This work is structured as follows. Section 2 briefly describes the SeVaTax framework, its graphical primitives and a required set of reasoning services to check them. Section 3 introduces the FOL formalisation of SeVaTax diagrams and section 4 presents the corresponding $\mathcal{ALCI}$ encoding together with complexity results. Finally, section 5 shows some related works and section 6 concludes with the main contributions and future works.

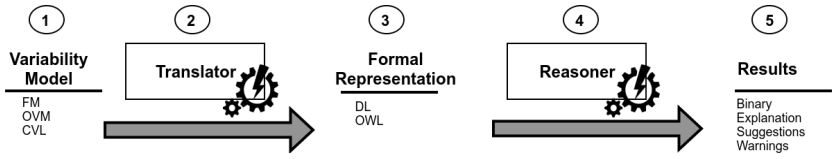


**Figure 1.** General Process for Automated Variability Analysis Process

## 2 The SeVaTax Framework

The automated variability analysis process includes five components, as depicted in Fig. 1. The first component is the variability model, which may be specified by some of the different modelling languages, such as Feature Model (FM), OVM or Constraint Variability Language (CVL) [11], among others. The second component is a translator for transforming the graphical model into a formal representation. The output of this translator is sent to a reasoner responsible for validating the model. The results of this process are defined according to the next most common cases extracted from the literature: Valid Model (VM), Valid Instantiation (VI), All Products (AP), Instantiation Number (IN) and Problem Detection (PD). The first one, *VM*, determines whether is possible to

generate a product from a variability model; *VI* checks if the instantiation might generate a valid product given a model and one of its configurations; *AP* determines all the possible instantiations; and *IN* specifies the number of possible instantiations. Finally, *PD* is responsible for identifying the causes which determine that a model is invalid.

One of the main components of the SeVaTax framework is the graphical language based on an OVM-extended notation for representing each SPL functionality. In previous works [6, 5], we have already defined this model based on a functionality-oriented methodology. That is, each functionality of the SPL is modelled as a functional datasheet represented by a set of services, common[1] and variant ones, interacting among them for performing the functionality. Each datasheet includes a set of hierarchical structures, where each structure is composed by a root service and a set of services (variation points or variants). Then, during the derivation process, these variants are selected configuring a new product and providing flexibility to the SPL.

We have defined a set of interactions that allows us to represent the functionality as graphical notation. These interactions involve common and variant services, and are divided into *variability types* for denoting the variant interactions among services; *dependencies* for denoting interaction between services; and *scope* for specifying the scope of each variant point [4]. The complete set of interactions supported in this work (and graphically depicted in Fig. 2) are:

Variability Types:
- **Mandatory variation point**: it determines the selection of a variant service when the variation point is included in the product being derived.
- **Optional variation point**: it specifies that zero or more variant services, associated to the variation point, can be selected.
- **Alternative variation point**: it defines that only one variant service, of the set of associated variants of the variation point, must be selected (XOR relation).
- **Variant variation point**: it defines that at least one variant service, of the set of associated variants of the variation point, must be selected (OR relation).

Dependencies:
- **Requires**: it specifies a relation between two variant services in which the selection of one variant service requires the selection of the other one.
- **Excludes**: it is the opposite of the requires dependency specifying the exclusion of a variant when another one is selected.

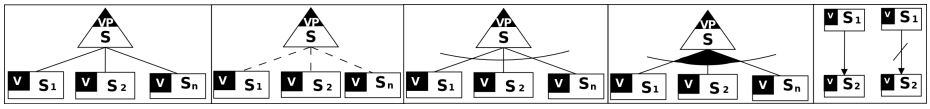


**Figure 2.** Left-Right: Mandatory; Optional; Alternative; Variant; Requires; Excludes. VP = Variation Point. V = Variant.

[1] Common services are services which will be part of every product derived from the SPL

The SeVaTax framework also provides the following primitive and scope operators, which are out of scope of this work:
- **Uses** ( ◄——► ) dependency: it specifies a dependence between common services, which are not necessarily associated with a variation point.
- **Global Variation Point (Global VP)** (GV): it specifies that if the variation point is instantiated in a specific way, it will be applied in that way for all functionality including that variation point.
- **Specific Variation Point (Specific VP)**(SV): it specifies that the instantiation of the variation point is particular for each functionality including that variation point.

A SeVaTax model provides a graphical notation depicting associations between two specific types of services, variation point (VP) and variant (V). In this work, associations may be variability types and dependencies. Each variability must relate one VP with a set of at least one V or VP. Moreover, each V must be only associated to one and only one VP.

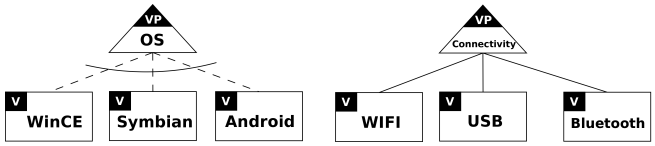


**Figure 3.** Left-Right: Alternative variability type for OS; Mandatory variability type for Connectivity.

*Example 1.* Fig. 3 depicts alternative and mandatory variability types modelling the variant of an operating system (OS) and the connectivity for a mobile phone, respectively. The OS variation point allows configuring three possible products: {OS, WinCE}, {OS, Symbian} and {OS, Android}. The Connectivity VP configures only the following product: {Connectivity, WIFI, USB, Bluetooth}.

Towards a new implementation of the SeVaTax framework, we present a formalisation of the OVM-based models and introduce a novel DL encoding to check diagrams for consistency, which have not been yet combined into the very same framework.

## 3 Formalising SeVaTax Diagrams

In this section we introduce the semantics of each variability type: Mandatory, Optional, Alternative and Optional and dependencies: Requires and Excludes, in terms of first order logic (FOL). We do not deal with either Uses dependency or scope operators. A FOL encoding is firstly proposed for analysing general properties of models, and easing comparisons with related approaches. In next section, this semantics is instantiated in $\mathcal{ALCI}$.

**Definition 1.** *Let $\Sigma$ be a SeVaTax diagram. We define $\Sigma^{FOL}$ as the corresponding set of FOL assertions representing such diagram and constructed as follows. The signature of the $\Sigma^{FOL}$ knowledge base includes the followings domain predicates:*

- *$S_{v_1}, ..., S_{v_m}, S_{vp_1}, ..., S_{vp_n}$ are FOL unary predicates for services, $S_{v_j}$ for variant services and $S_{vp_i}$ for variation point services,*
- *MAND, OPT, ALT and VAR are binary relations. However, we formalize such variability types as r+1-ary predicates denoted as MAND$^{r+1}$, OPT$^{r+1}$, ALT$^{r+1}$ and VAR$^{r+1}$, respectively. For instance, if $x$ MAND $\{y_1, ..., y_r\}$ is an association defined in $\Sigma$, we characterise it as the FOL r+1-ary predicate: MAND$^{r+1}(x, y_1, ..., y_r)$.*
- *Each dependency REQ and EXC corresponds to the FOL binary predicate REQ and EXC, respectively.*

*Now, we proceed to formalise $\Sigma$ as a set of FOL assertions. Firstly, we introduce the following background assertion imposing that each service may be instantiated by at most one element. Such formulae must be specified for each variation point and variant service in $\Sigma$. Then, we introduce the domain axioms describing the variability types and dependencies of SeVaTax.*

$$\exists x.S_{vp_k}(x) \wedge \neg\exists y.(S_{vp_k}(y) \wedge x \neq y) \qquad \exists x.S_{v_k}(x) \wedge \neg\exists y.(S_{v_k}(y) \wedge x \neq y)$$

$$\neg\exists x.(S_{vp_i}(x) \wedge S_{vp_l}(x)),\ 1 \leq i, l \leq n, \qquad \neg\exists x.(S_{v_j}(x) \wedge S_{v_l}(x)),\ 1 \leq i, l \leq m,\ i \neq l$$

***Mandatory**: for each MAND$^{m+1}$ in the signature of $\Sigma^{FOL}$*

$$\exists x.\mathit{MAND}^{m+1}(x, y_1, ..., y_m) \wedge S_{vp_k}(x) \rightarrow \bigwedge_{i=1}^{m}(S_{vp_i}(y_i) \vee S_{v_i}(y_i)), \quad m \geq 1$$

$$\exists y_1...y_m.\mathit{MAND}^{m+1}(x, y_1, ..., y_m) \wedge \bigwedge_{i=1}^{m}(S_{vp_i}(y_i) \vee S_{v_i}(y_i)) \rightarrow S_{vp_k}(x), \quad m \geq 1$$

$$\exists x, y_1...y_m.\Big(\mathit{MAND}^{m+1}(x, y_1..., y_m) \rightarrow \bigwedge_{i=1}^{m} x \neq y_i\Big), \quad m \geq 1$$

*The first and second sentences define the mandatory interaction among services, which requires that one instance of a variation point is related to each other variant or variation point through the predicate MAND$^{m+1}$, and vice versa. The last one imposes that the variation point, from which the variability is defined, must not be a variant of itself.*

***Optional**: for each OPT$^{m+1}$ in the signature of $\Sigma^{FOL}$*

$$\exists x.\big(\mathit{OPT}^{m+1}(x, y_1, ..., y_m) \rightarrow S_{vp_k}(x)\big), \quad m \geq 2$$

$$\exists x, y_1...y_m.\Big(\mathit{OPT}^{m+1}(x, y_1..., y_m) \rightarrow \bigwedge_{i=1}^{m} x \neq y_i\Big), \quad m \geq 2$$

*The first imposes that one instance of a variation point may be related to zero or more variants or variation points through the predicate OPT$^{m+1}$, while the last sentence also imposes that the variation point, from which the variability is defined, must not be a variant of itself.*

***Alternative**: for each ALT$^{m+1}$ in the signature of $\Sigma^{FOL}$*

$$\exists x.\mathit{ALT}^{m+1}(x, y_1..., y_m) \wedge S_{vp_k}(x) \rightarrow \bigoplus_{i=1}^{m}(S_{vp_i}(y_i) \vee S_{v_i}(y_i)), \quad m \geq 2$$

$$\mathit{ALT}^{m+1}(x, y_1..., y_m) \wedge \Big(\bigoplus_{i=1}^{m}(S_{vp_i}(y_i) \vee S_{v_i}(y_i))\Big) \rightarrow S_{vp_k}(x), \quad m \geq 2$$

$$\exists x, y_1...y_m.\Big(\mathit{ALT}^{m+1}(x, y_1..., y_m) \rightarrow \bigwedge_{i=1}^{m} x \neq y_i\Big), \quad m \geq 2$$

*The $\oplus$ operator is semantically equivalent to the exclusive or (XOR) operation. Similarly, the first sentence requires that one instance of a variation point is related to one and only one variant or variation point through the predicate ALT$^{m+1}$. The two last sentences define the same constraints than for the mandatory variability types.*

***Variant**: for each VAR$^{m+1}$ in the signature of $\Sigma^{FOL}$*

$$\exists x.\mathit{VAR}^{m+1}(x, y_1..., y_m) \wedge S_{vp_k}(x) \rightarrow \bigvee_{i=1}^{m}(S_{vp_i}(y_i) \vee S_{v_i}(y_i)), \quad m \geq 2$$

$$\mathit{VAR}^{m+1}(x, y_1..., y_m) \wedge \Big(\bigvee_{i=1}^{m}(S_{vp_i}(y_i) \vee S_{v_i}(y_i))\Big) \rightarrow S_{vp_k}(x) \quad m \geq 2$$

$$\exists x, y_1...y_m.\Big(\mathit{VAR}^{m+1}(x, y_1..., y_m) \rightarrow \bigwedge_{i=1}^{m} x \neq y_i\Big), \quad m \geq 2$$

*Lastly, the first sentence defines that one instance of a variation point is related at least one variant or variation point through the predicate VAR$^{m+1}$. The two last sentences impose the same constraints than the previous variability types.*

*Now we define the dependencies REQ and EXC as follows. We write sentences for only one relation REQ and only one EXC, while the remaining ones are defined by replacing the types of services. Moreover, in both formulae, the $\oplus$ operator is semantically equivalent to the exclusive or (XOR) operation.*

***Requires**: $\exists x.\mathit{REQ}(x, y) \rightarrow S_{vp_k}(y) \oplus S_{v_k}(y)$*
***Excludes**: for any fixed $k$,*

$$\exists x.\mathit{EXC}(x, y) \wedge (S_{vp_k}(x) \oplus S_{v_k}(x)) \rightarrow \neg(S_{vp_k}(y) \vee S_{v_k}(y))$$

$$\exists x.\mathit{EXC}(x, y) \wedge (S_{vp_k}(y) \oplus S_{v_k}(y)) \rightarrow \neg(S_{vp_k}(x) \vee S_{v_k}(x))$$

### 3.1 Reasoning Services

We have characterised SeVaTax diagrams in FOL in order to allow modellers to use this formalisation for checking relevant properties over these diagrams. Some typical services of interest are the followings:

**Definition 2.** *Let $\Sigma$ be a SeVaTax diagram and $\Sigma^{FOL}$ be the corresponding SeVaTax Knowledge Base:*
*- **Service consistency**: A service $S$ is consistent if the SeVaTax diagram admits an instantiation in which $S$ has instances. Formally, $\Sigma^{FOL}$ corresponding to $\Sigma$ admits a model in which $S$ has a nonempty extension.*
*- **SeVaTax consistency**: A SeVaTax diagram is consistent if it admits an instantiation, i.e. an actual choice of variants or variation points matching the constraints imposed by the diagram such that some service is not empty. Formally, the set of FOL assertions corresponding to the SeVaTax diagram admits a model in which some the service has a nonempty extension. This means that at least one product might be derived from the variability model. This reasoning service corresponds with the query Valid Model as defined in section 2.*
*- **Strong SeVaTax consistency**: A SeVaTax diagram $\Sigma$ is strongly consistent if for each $S$ (variant or variation point) the set of FOL assertions corresponding to the diagram $\Sigma^{FOL}$, admits some model such that $S$ has a nonempty extension. This means that every service in the variability model might be derived as part of at least one product.*
*- **Dead Service**: A service $S$ (variant or variation point) is a dead service w.r.t. $\Sigma^{FOL}$ if there is no model such that $S$ has a nonempty extension. This means that some services might never be derived as part of some product. This service corresponds with the query Problems Detection as also defined in section 2.*

Each service defined above can be reduced to each other [2]. Moreover, the service consistency reasoning task amounts to checking if a given service can be instantiated and corresponds to the notion of concept consistency DLs.

## 4 DL Encoding of SeVaTax Diagrams

We now define a DL formalisation of SeVaTax diagrams and prove the correctness of the encoding. We only show the encoding for interactions between variation points and variants so that the below assertions must be rewritten for the remaining possible combinations. On the other hand, for sake of simplicity in this ongoing approach we consider singleton DL classes to encode services. However, we are working in other approaches using nominals or ABox facts to express atomic services and thus keeping the upper bound of complexity under control.

**Definition 3.** *Let $\Sigma$ be a SeVaTax diagram. $\Sigma^{\mathcal{ALCI}}$ is the DL Knowledge Base representing $\Sigma$ and constructed as $\phi(\Sigma) = (\mathcal{A}, \mathcal{R}, \mathcal{T})$ as follows:*

- *The set $\mathcal{A}$ of atomic concepts of $\phi(\Sigma)$ contains the elements below:*
  - *an atomic concept $\phi(S_{vp_i})$ for each variation point $S_{vp_i}$ .*
  - *an atomic concept $\phi(S_{v_j})$ for each variant $S_{v_j}$.*
- *Let us suppose services $S$ and $S_i$ in the SeVaTax diagram $\Sigma$ related by a variability type ($S$ is related to $S_i$). Then the edge connecting the nodes $S$ and $S_i$ is formalised into an $\mathcal{ALCI}$ role $r_{S_i}$ which represents the relation between $\phi(S)$ and $\phi(S_i)$. If both $S$ and $S_i$ in $\Sigma$ are related by any of the Requires and Excludes dependencies then the edge connecting the nodes $S$ and $S_i$ is formalised into $\mathcal{ALCI}$ roles $r_{R_{S_i}}$*

*and $r_{E_{S_i}}$, respectively. This means that each edge connecting services through any variability type or dependency, as defined in section 2, is translated into a DL role.*

– *The set $\mathcal{T}$ of assertions of $\phi(\Sigma)$ contains the following elements:*

$$\phi(S_{vp_i}) \sqsubseteq \neg\phi(S_{vp_l}) \quad \phi(S_{v_j}) \sqsubseteq \neg\phi(S_{v_l}) \quad 1 \leq i \leq n, \;\; 1 \leq j \leq m, \quad i, j \neq l$$

- *if $S_{vp_k}$MAND$\{S_{v_1}, ..., S_{v_m}\}$ is in $\Sigma$, then (for each $j$ and for any fixed $k$):*

$$\exists r_{S_{v_j}} \sqsubseteq \phi(S_{vp_k}) \qquad \exists r^-_{S_{v_j}} \sqsubseteq \phi(S_{v_j}) \qquad \phi(S_{vp_k}) \sqsubseteq \prod_{1 \leq j \leq m} \exists r_{S_{v_j}}.\phi(S_{v_j})$$

$$\phi(S_{v_j}) \sqsubseteq \exists r^-_{S_{v_j}}.\phi(S_{vp_k}), \textit{ for each } \phi(S_{v_j}), 1 \leq j \leq m$$

- *if $S_{vp_k}$OPT$\{S_{v_1}, ..., S_{v_m}\}$ is in $\Sigma$, then (for each $j$ and for any fixed $k$):*

$$\bigsqcup_{1 \leq j \leq m} \exists r_{S_{v_j}}.\phi(S_{v_j}) \sqsubseteq \phi(S_{vp_k})$$

*and for each $\phi(S_{v_j})$ element playing the corresponding role $r_{S_{v_j}}$:*

$$\phi(S_{v_j}) \sqsubseteq \exists r^-_{S_{v_j}}.\phi(S_{vp_k}), \textit{ for each } \phi(S_{v_j}), 1 \leq j \leq m$$

- *if $S_{vp_k}$ALT$\{S_{v_1}, ..., S_{v_m}\}$ is in $\Sigma$, then (for each $j$ and for any fixed $k$):*

$$\phi(S_{vp_k}) \sqsubseteq \bigsqcup_{1 \leq j \leq m} \left( \exists r_{S_{v_j}}.\phi(S_{v_j}) \sqcap \left( \prod_{\substack{1 \leq l \leq m \\ j \neq l}} \neg\exists r_{S_{v_l}}.\phi(S_{v_l}) \right) \right)$$

*and for the only one $\phi(S_{v_j})$ element playing the corresponding role $r_{S_{v_j}}$:*

$$\phi(S_{v_j}) \sqsubseteq \exists r^-_{S_{v_j}}.\phi(S_{vp_k}), \textit{ for each } \phi(S_{v_j}), 1 \leq j \leq m$$

- *if $S_{vp_k}$VAR$\{S_{v_1}, ..., S_{v_m}\}$ is in $\Sigma$, then (for each $j$ and for any fixed $k$):*

$$\phi(S_{vp_k}) \sqsubseteq \bigsqcup_{1 \leq j \leq m} \exists r_{S_{v_j}}.\phi(S_{v_j})$$

*For each $\phi(S_{v_j})$ element playing the corresponding role $r_{S_{v_j}}$:*

$$\phi(S_{v_j}) \sqsubseteq \exists r^-_{S_{v_j}}.\phi(S_{vp_k}), \textit{ for each } \phi(S_{v_j}), 1 \leq j \leq m$$

- *if $S_{vp_k}$REQ $S_{v_l}$ is in $\Sigma$, then:* $\phi(S_{vp_k}) \sqsubseteq \forall r^-_{R_{S_l}}.\phi(S_{v_l})$
- *if $S_{vp_k}$EXC $S_{v_l}$ is in $\Sigma$, then (for any fixed $k$):*

$$\phi(S_{vp_k}) \sqsubseteq \neg(\exists r_{E_{S_l}}.\phi(S_{v_l})) \qquad \phi(S_{v_l}) \sqsubseteq \neg(\exists r^-_{E_{S_l}}.\phi(S_{vp_k}))$$

We now show that the encoding for the variability types in a $\Sigma$ is correct. For this, we establish a correspondence between models in $\Sigma^{FOL}$ KB and models of the corresponding $\Sigma^{\mathcal{ALCI}}$ KB.

**Theorem 1.** *Let $\Sigma$ be a SeVaTax diagram, $\Sigma^{FOL}$ the corresponding FOL formalisation and $\Sigma^{\mathcal{ALCI}}$ the $\mathcal{ALCI}$ KB constructed as specified above. Then a service $S$ is consistent in $\Sigma$ iff the corresponding concept $\phi(S)$ is satisfiable w.r.t. $\Sigma^{\mathcal{ALCI}}$.*

*Proof.* [$\Rightarrow$] *Let $S$ be consistent service in $\Sigma$, then there exists a model of $\Sigma^{FOL}$ in which $S$ has a nonempty extension. Let $\mathcal{O} = (\Delta^{\mathcal{O}}, .^{\mathcal{O}})$ be a model of $\Sigma^{FOL}$ such that $S^{\mathcal{O}} \neq \emptyset$. We build a model $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ of $\Sigma^{\mathcal{ALCI}}$ such that $\phi^{\mathcal{I}}(S) \neq \emptyset$, as follows:*

- $\Delta^{\mathcal{I}} = \Delta^{\mathcal{O}}$
- $\phi^{\mathcal{I}}(S_{vp_i}) = S^{\mathcal{O}}_{vp_i}$, *for each* $\phi(S_{vp_i})$ *corresponding to services* $S_{vp_i} \in \Sigma$
- $\phi^{\mathcal{I}}(S_{v_j}) = S^{\mathcal{O}}_{v_j}$, *for each* $\phi(S_{v_j})$ *corresponding to services* $S_{v_j} \in \Sigma$
- $R^{\mathcal{I}}$ *are all the $\mathcal{ALCI}$-roles corresponding to variability types and dependencies. Each role $r^{\mathcal{I}}_{S_j}$ modelling the relation between the first and the $j$th component of any $n+1$-ary variability type (for each arbitrary n), is defined as follows.*
  - $r^{\mathcal{I}}_{S_j} = \{(s_0, s_j)\}$ *such that* $(s_0, s_1, ..., s_n) \in$ *MAND*$^{n+1(\mathcal{I})}$, $1 \leq j \leq n$
  - $r^{\mathcal{I}}_{S'_j} = \{(s'_0, s'_j)\}$ *such that* $(s'_0, s'_1, ..., s'_n) \in$ *OPT*$^{n+1(\mathcal{I})}$, $1 \leq j \leq n$
  - $r^{\mathcal{I}}_{S''_j} = \{(s''_0, s''_j)\}$ *such that* $(s''_0, s''_1, ..., s''_n) \in$ *ALT*$^{n+1(\mathcal{I})}$, $1 \leq j \leq n$
  - $r^{\mathcal{I}}_{S'''_j} = \{(s'''_0, s'''_j)\}$ *such that* $(s'''_0, s'''_1, ..., s'''_n) \in$ *VAR*$^{n+1(\mathcal{I})}$, $1 \leq j \leq n$

  $R^{\mathcal{I}}$ *also includes $r_{R_{S_j}}$ and $r_{E_{S_j}}$ for the variability dependencies* $REQ^{\mathcal{I}}$ *and* $EXC^{\mathcal{I}}$.

*It is immediate to check that $\mathcal{I}$ satisfies all the assertions in $\Sigma^{\mathcal{ALCI}}$. We detail the proof for the Mandatory variability type associating a variant point to only variant services, since the remaining possible types and dependencies are similarly proved.*

*Let $S$ be a variation point. As it is consistent in $\Sigma$, $S^{\mathcal{O}} \neq \emptyset$. Then, there exists a variability type or dependency in which $S$ participates. Let us suppose that this variability type is mandatory. So, there must exist variants $s_1, ..., s_n \in \Delta^{\mathcal{O}}$,and $S_{v_i}(s_i)$ and* MAND$(s_0, s_1, ..., s_n)$ *hold. Moreover:*

- *by definition of R, there exist $r^{\mathcal{I}}_{S_j}$ roles relating $s_0$ with each $s_j$. As $s_0 \in \phi^{\mathcal{I}}(S_{vp_0})$ and $s_j \in \phi^{\mathcal{I}}(S_{v_j})$ for each $j$, thus both $\exists r_{S_j} \sqsubseteq \phi(S_{vp_0})$ and $\exists r^{-}_{S_j} \sqsubseteq \phi(S_{v_j})$ hold.*
- *$\phi^{\mathcal{I}}(S_{vp_0})$ participates in the mandatory association through roles $r^{\mathcal{I}}_{S_j}$ and as $\phi^{\mathcal{I}}(S_{v_j}) \neq \emptyset$ for all $j$, each second element of each $r^{\mathcal{I}}_{S_j}$ is a element $\phi^{\mathcal{I}}(S_{v_j})$, then $\phi(S_{vp_0}) \sqsubseteq \bigsqcap_{1 \leq j \leq n} \exists r_{S_{v_j}} \phi(S_{v_j})$ holds.*
- *Conversely, each existing $\phi^{\mathcal{I}}(S_{v_j})$ participates in the corresponding $r^{\mathcal{I}}_{S_j}$, whose first element is also $\phi^{\mathcal{I}}(S_{vp_0})$, for each $j$. Then the assertion $\phi(S_{v_j}) \sqsubseteq \exists r^{-}_{S_j}.\phi(S_{vp_0})$ holds.*

*Hence, $\mathcal{I}$ satisfies the corresponding $\mathcal{ALCI}$ assertions for the Mandatory variability type. The proof is similar if we consider variation points instead of variant in the mandatory variability. Moreover, the same proof applies to the remaining variability types and dependencies.*

[$\Leftarrow$] *By the tree-model property, if $\phi(S)$ is satisfiable w.r.t. the $\Sigma^{\mathcal{ALCI}}$ KB then there exists a tree-like model $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ of $\Sigma^{\mathcal{ALCI}}$ such that $\phi^{\mathcal{I}}(S) \neq \emptyset$. From such a tree-like model we can build an instantiation $\mathcal{O} = (\Delta^{\mathcal{O}}, .^{\mathcal{O}})$ of $\Sigma$ such $S^{\mathcal{O}} \neq \emptyset$ as follows.*

- $\Delta^{\mathcal{O}} = \bigcup_{S_{vp} \in \mathcal{VP}} \phi^{\mathcal{I}}(S_{vp}) \bigcup_{S_v \in \mathcal{V}} \phi^{\mathcal{I}}(S_v)$, *where* $\mathcal{VP}$ *and* $\mathcal{V}$ *denote the set of variation points and variants of* $\Sigma$*, respectively.*
- $S^{\mathcal{O}}_{vp_i} = \phi^{\mathcal{I}}(S_{vp_i})$, *for all service* $S_{vp_i}$ *in* $\Sigma$.
- $S^{\mathcal{O}}_{v_j} = \phi^{\mathcal{I}}(S_{v_j})$, *for all service* $S_{v_j}$ *in* $\Sigma$.
- $REQ^{\mathcal{O}} = r_{R_{S_j}},\ r_{R_{S_j}} \in R^{\mathcal{I}}$ *for each Requires dependency in* $\Sigma$
- $EXC^{\mathcal{O}} = r_{E_{S_j}},\ r_{E_{S_j}} \in R^{\mathcal{I}}$ *for each Excludes dependency in* $\Sigma$
- $MAND^{n+1(\mathcal{O})} = \{(s_0, s_1, ..., s_n) \mid \bigwedge_{i=1}^{n}(s_0, s_j) \in r^{\mathcal{I}}_{S_j}\}$
- $OPT^{n+1(\mathcal{O})} = \{(s_0, s_1, ..., s_n) \mid \bigwedge_{i=1}^{n}(s_0, s_j) \in r^{\mathcal{I}}_{S_j}\}$
- $ALT^{n+1(\mathcal{O})} = \{(s_0, s_1, ..., s_n) \mid \bigwedge_{i=1}^{n}(s_0, s_j) \in r^{\mathcal{I}}_{S_j}\}$
- $VAR^{n+1(\mathcal{O})} = \{(s_0, s_1, ..., s_n) \mid \bigwedge_{i=1}^{n}(s_0, s_j) \in r^{\mathcal{I}}_{S_j}\}$

*Observe that, since* $\mathcal{O}$ *is a tree-like model, it is guaranteed that there is only one object* $s_0 \in S^{\mathcal{O}}_{vp}$ *that represents a variation point. Hence, by definition of mandatory dependency there must also exist the elements* $s_j \in S^{\mathcal{O}}_{v_j}$ *representing tuples for this type so as the FOL assertions for MAND*$(s_0, s_1, ...s_n)$ *hold. Similarly for the other variability types and dependencies. Finally,* $\mathcal{O}$ *correctly instantiates* $\Sigma$. □

**Theorem 2.** *Reasoning over SeVaTax schemes is EXPTIME w.r.t. SeVaTax service consistency.*

*Proof. By theorem 1 and* $\mathcal{ALCI}$ *complexity [1].* □

*Example 2.* Before ending this section, we revisit the example 3 in order to show the DL encoding for the SeVaTax diagram showing the use of the mandatory variability type for the $Connectivity$ VP. The $\mathcal{ALCI}$ encoding for mandatory type is the following:

$$\exists r_{WIFI} \sqsubseteq Connectivity \quad \exists r_{USB} \sqsubseteq Connectivity \quad \exists r_{Bluetooth} \sqsubseteq Connectivity$$

$$\exists r^{-}_{WIFI} \sqsubseteq WIFI \quad \exists r^{-}_{USB} \sqsubseteq USB \quad \exists r^{-}_{Bluetooth} \sqsubseteq Bluetooth$$

$$Connectivity \sqsubseteq \exists r_{WIFI}.WIFI \sqcap \exists r_{USB}.USB \sqcap \exists r_{Bluetooth}.Bluetooth$$

$$Connectivity \sqsubseteq \exists r^{-}_{WIFI}.WIFI \quad Connectivity \sqsubseteq \exists r^{-}_{USB}.USB$$

$$Connectivity \sqsubseteq \exists r^{-}_{Bluetooth}.Bluetooth$$

## 5 Related Works

Table 1 compares main works in the literature proposing variability analysis of FMs due to, at the best of our knowledge, there does not exist any work embedding other modelling approaches in DL. Thus, the second column of the table shows the formal model (F.M.) used by each proposal, the third one indicates the reasoners, and the last columns describe the questions which reasoners should be capable of answer [16, 17, 3]. On the other hand, it does exist proposals integrating OVM and SAT solvers [20, 28], however, they present the same drawbacks explained in section 1. Then, we conclude this section only detailing the related works whose proposals embed FMs in DL.

| Aproach | F.M. | Reasoner | Verify | | | | |
|---|---|---|---|---|---|---|---|
| | | | VM | VI | AP | NP | PD |
| Wang et al. [34] | OWL | FACT++ | x | x | - | - | x |
| Noorian et al. [21] | DL | Pellet | x | x | - | - | x |
| Fan et al. [8] | DL | Racer | x | x | - | - | - |
| Das et al. [7] | OWL-DL | RACER | x | x | - | - | x |
| Ripon et al.[26] | OWL-DL | RACER | x | x | - | - | x |
| Ryssel et al. [29] | OWL-CSP | FACT++ | x | x | - | - | - |
| Rincon et al. [25] | OWL - SQWRL | JESS | x | x | - | - | x |
| Zaid et al. [35] | OWL-DL SWRL | Pellet | x | x | - | - | x |

**Table 1.** Summary of Semantic Analysis Process of Variability Models

In Table 1, we can see the work of Wang et. al [34] in which authors propose a method for formalising FMs into OWL [9], by using a prototype tool (that in future works will be available as a Protégé [15] plugin. Then, a FACT++ reasoner [33] is used for problem detection by providing a debugger module, which identifies the most general conflicts and gives an explanation. Noorian et.al [21] present a translation of FMs into DL, and use Pellet [30] and a Reiter's algorithm [24] for solving two types of inconsistencies (applied to structural and integrity constraints). In Fan & Zhang [8] authors present a set of translation rules for building DL knowledge bases $\mathcal{ALCQI}$ from FMs. Then, the consistency of these models is checked by using the RACER reasoner [10]. In Das et al. [7] and Ripon et al. [26] the consistency and detection of inconsistent classes are evaluated according to the responses of the RACER reasoner. In Rincon et al. [25] authors detect dead and false optional features and explain the causes in natural language. Finally, Zaid et al. [35] detect specific classes generating inconsistencies by the set of rules defined in SWRL [12] over the OWL model.

As we can see, none of the analysed works makes an analysis of All Products (AP) and Instantiation Number (IN) queries, which are two of the most common queries needed in any variability analysis process. At the same time, more specific queries [16, 17, 3] are not analysed either.

## 6 Conclusions and Future Work

The well-known benefits of using DL to conceptual representation admitting decidable reasoning fit with the requirements for implementing an automated variability analysis process for the SeVaTax framework. The contributions of this work are twofold. Firstly, the formalisation of OVM-based SeVaTax model expressed in first order logic and secondly, a novel $\mathcal{ALCI}$ encoding allowing reasoning over SeVaTax. Moreover, we establish the EXPTIME-membership of reasoning on its diagrams w.r.t. service consistency. Until now such a formalisation of OVM-based diagrams had not been done.

As future work, we plan to continue extending reasoning capabilities of our framework. Additionally, we are currently developing a prototype providing a back-end for the automated support of consistency checking of SeVaTax diagrams along with its encoding into DL knowledge bases.

## References


1. Artale, A., Calvanese, D., Kontchakov, R., Ryzhikov, V., Zakharyaschev, M.: Reasoning over Extended ER Models. In: Conceptual Modeling - ER 2007, 26th International Conference on Conceptual Modeling, Auckland, New Zealand (2007)
2. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, New York, NY, USA (2003)
3. Benavides, D., Segura, S., Ruiz-Cortés, A.: Automated Analysis of Feature Models 20 Years Later: A Literature Review. Inf. Syst. 35(6), 615–636 (Sep 2010)
4. Brisaboa, N.R., Cortinas, A., Luaces, M.R., Pol'la, M.: A Reusable Software Architecture for Geographic Information Systems Based on Software Product Line Engineering. In: Model and Data Engineering - 5th International Conference, MEDI 2015, Rhodes, Greece, September 26-28, 2015, Proceedings. pp. 320–331 (2015)
5. Buccella, A., Cechich, A., Pol'la, M., Arias, M., Doldan, S., Morsan, E.: Marine ecology service reuse through taxonomy-oriented spl development. Computers & Geosciences –(0), In press (2014)
6. Buccella, A., Cechich, A., Arias, M., Pol'la, M., del Socorro Doldan, M., Morsan, E.: Towards systematic software reuse of gis: Insights from a case study. Computers & Geosciences 54(0), 9 – 20 (2013)
7. Das, N.C., Ripon, S., Hossain, O., Uddin, M.S.: Requirement Analysis of Product Line Based Semantic Web Services. Lecture Notes on Software Engineering (2014)
8. Fan, S., Zhang, N.: Feature Model Based on Description Logics, pp. 1144–1151. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
9. Grau, B.C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U.: OWL 2: The Next Step for OWL. Web Semant. 6(4), 309–322 (Nov 2008)
10. Haarslev, V., Möller, R.: RACER System Description. In: Goré, R., Leitsch, A., Nipkow, T. (eds.) International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, Siena, Italy. pp. 701–705. Springer-Verlag (2001)
11. Haugen, O., Moller-Pedersen, B., Oldevik, J., Olsen, G.K., Svendsen, A.: Adding Standardized Variability to Domain Specific Languages. In: 2008 12th International Software Product Line Conference. pp. 139–148 (2008)
12. Horrocks, I., Patel-Schneider, P.F., Bechhofer, S., Tsarkov, D.: OWL Rules: A Proposal and Prototype Implementation. Web Semant. 3(1), 23–40 (Jul 2005)
13. Janota, M.: Do SAT Solvers Make Good Configurators? In: Software Product Lines, 12th International Conference, SPLC 2008, Limerick, Ireland, September 8-12, 2008, Proceedings. Second Volume (Workshops) (2008)
14. Kang, K., Cohen, S., Hess, J., Nowak, W., Peterson, S.: Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University Pittsburgh, PA. (1990)
15. Knublauch, H., Fergerson, R., Noy, N., Musen, M.: The Protégé OWL plugin: An open development environment for semantic web applications (2004)
16. Kowal, M., Ananieva, S., Thüm, T.: Explaining Anomalies in Feature Models. In: Proceedings of the 2016 ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences. GPCE 2016, ACM, New York, NY, USA (2016)
17. von der Massen, T., H. Lichter, H.: Deficiencies in Feature Models. In: Mannisto, T., Bosch, J. (eds.) Workshop on Software Variability Management for Product Derivation - Towards Tool Support (2004)
18. Mendonca, M., Cowan, D.: Decision-making coordination and efficient reasoning techniques for feature-based configuration. Science of Computer Programming (2010), coordination Models, Languages and Applications (SAC'08)

19. Mendonca, M., Wasowski, A., Czarnecki, K.: SAT-based Analysis of Feature Models is Easy. In: Proceedings of the 13th International Software Product Line Conference. pp. 231–240. SPLC '09, Carnegie Mellon University, Pittsburgh, PA, USA (2009)
20. Metzger, A., Pohl, K., Heymans, P., Schobbens, P.Y., Saval, G.: Disambiguating the Documentation of Variability in Software Product Lines: A Separation of Concerns, Formalization and Automated Analysis. In: 15th IEEE International Requirements Engineering Conference (RE 2007). pp. 243–253 (Oct 2007)
21. Noorian, M., Ensan, A., Bagheri, E., Boley, H., Biletskiy, Y.: Feature Model Debugging based on Description Logic Reasoning. In: Proceedings of the 17th International Conference on Distributed Multimedia Systems, DMS 2011, October 18-20, 2011, Convitto della Calza, Florence, Italy. pp. 158–164 (2011)
22. Pohl, K., Böckle, G., Linden, F.J.v.d.: Software Product Line Engineering: Foundations, Principles and Techniques. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2005)
23. Pol'la, M., Buccella, A., Arias, M., Cechich, A.: SeVaTax: service taxonomy selection validation process for SPL development. In: 2015 34th International Conference of the Chilean Computer Science Society (SCCC). pp. 1–6 (Nov 2015)
24. Reiter, R.: A Theory of Diagnosis from First Principles. Artif. Intell. 32(1), 57–95 (Apr 1987)
25. Rincón, L.F., Giraldo, G.L., Mazo, R., Salinesi, C.: An Ontological Rule-Based Approach for Analyzing Dead and False Optional Features in Feature Models. Electron. Notes Theor. Comput. Sci. (2014)
26. Ripon, S., Piash, M.M., Hossain, S.M.A., Uddin, M.S.: Semantic Web Based Analysis of Product Line Variant Model. JCEE 2014 6(1), 1–6 (2014)
27. Roos-Frantz, F., Galindo, J.A., Benavides, D., Cortés, A.R., Garcıa-Galán, J.: Automated analysis of diverse variability models with tool support. Jornadas de Ingenierıa del Software y de Bases de Datos (JISBD 2014), Cádiz. Spain p. 160 (2014)
28. Roos-Frantz, F., Galindo, J.A., Benavides, D., Ruiz-Cortés, A.: FaMa-OVM: A Tool for the Automated Analysis of OVMs. In: Proceedings of the 16th International Software Product Line Conference - Volume 2. SPLC '12, ACM, New York, NY, USA (2012)
29. Ryssel, U., Ploennigs, J., Kabitzsch, K.: Reasoning of Feature Models from Derived Features. In: Proceedings of the 11th International Conference on Generative Programming and Component Engineering. GPCE '12, ACM, New York, NY, USA (2012)
30. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Web Semant. 5(2), 51–53 (Jun 2007)
31. Sree-Kumar, A., Planas, E., Clarisó, R.: Analysis of Feature Models Using Alloy: A Survey. In: Proceedings 7th International Workshop on Formal Methods and Analysis in Software Product Line Engineering, FMSPLE@ETAPS 2016, Eindhoven, The Netherlands, April 3, 2016. pp. 46–60 (2016)
32. Sun, J., Zhang, H., Li, Y.F., Wang, H.: Formal Semantics and Verification for Feature Modeling. In: ICECCS '05: Proceedings of the 10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS'05). IEEE Computer Society (2005)
33. Tsarkov, D., Horrocks, I.: FaCT++ description logic reasoner: System description. In: In Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006. pp. 292–297. Springer (2006)
34. Wang, H.H., Li, Y.F., Sun, J., Zhang, H., Pan, J.: Verifying Feature Models Using OWL. Web Semant. 5(2), 117–129 (Jun 2007)
35. Zaid, L.A., Kleinermann, F., De Troyer, O.: Applying Semantic Web Technology to Feature Modeling. In: Proceedings of the 2009 ACM Symposium on Applied Computing. SAC '09, ACM, New York, NY, USA (2009)