Towards Privacy-aware Software Reuse

Iris Reinhartz-Berger[1], Anna Zamansky[1] and Agnes Koschmider[2]
[1]*Department of Information Systems, University of Haifa, Haifa, Israel*
[2]*Institute AIFB, Karlsruhe Institute of Technology, Karlsruhe, Germany*

Keywords: Reuse, Privacy, Variability Analysis, Compliance.

Abstract: As software becomes more complex, reusing and integrating artifacts from existing projects that may be taken from open or organization-proprietary repositories is becoming an increasingly important practice. This practice requires an in-depth understanding of the projects to be reused and particularly their common and variable features and their non-functional requirements. Different approaches have been suggested to analyze similarity and variability of different kinds of artifacts (mainly, requirements and code), e.g., clone detection and feature mining. These approaches, however, mainly address functional aspects of the software artifacts, while mostly neglecting aspects dictated by non-functional requirements. The recent progress with the General Data Protection Regulation (GDPR) highlights the importance of handling privacy concerns in software development. However, existing approaches do not directly refer to privacy challenges in software reuse. In this paper we propose integrating these two lines of research and introduce a privacy-aware software reuse approach. Particularly, we suggest to extend VarMeR – Variability Mechanisms Recommender – which analyzes software similarity based on exhibited behaviors and recommends on polymorphism-inspired reuse mechanisms, with privacy awareness considerations. These considerations are reflected in "privacy levels" of the reused artifacts.

1 INTRODUCTION

Software reuse has the potential to increase productivity, reduce costs and time-to-market and improve software quality (Lim, 1994). Applying this practice requires an in-depth understanding of the projects to be reused and particularly their common and variable features – functions and qualities. Different approaches have been suggested to analyze similarity and variability of different kinds of artifacts. Clone detection approaches, for example, propose textual, lexical, metric, tree/graph, and other comparisons, for detecting segments (primarily of code) that are similar according to some definition of similarity (Rattan et al. 2013). Feature mining includes *detection* – extraction of relevant information from the input artifacts and *analysis* – use of the information to infer, design and organize partitions that cluster the functional features of the input artifacts (Assunção et al., 2017). The artifacts are mainly requirements and code.

In previous work a high-level approach was suggested for analyzing reuse opportunities based on behaviors rather than specific implementations (Zamansky and Reinhartz-Berger, 2017). The approach, named VarMeR – Variability Mechanisms Recommender – has been developed for analyzing and visualizing similarity relationships across software products, which potentially may form product lines (Reinhartz-Berger and Zamansky, 2018). It is based on an ontological framework of software behavior (Reinhartz-Berger et al., 2015) and introduces three polymorphism-inspired mechanisms (parametric, subtyping, and overloading). The analysis outcomes are visualized as graphs whose nodes are the products (or parts of them) and the edges represent the potential appropriateness of applying the different polymorphism-inspired mechanisms.

The aforementioned approaches, including VarMeR, mainly address functionality, while mostly neglecting aspects dictated by non-functional requirements. Particularly, *privacy concerns* are not explicitly considered when recommending on reuse opportunities. The General Data Protection Regulation (GDPR), in application since May 25th 2018, imposes organizations to consider privacy throughout the complete development process. Due to

Reinhartz-Berger, I., Zamansky, A. and Koschmider, A.
Towards Privacy-aware Software Reuse.
DOI: 10.5220/0007566204480453
In *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2019)*, pages 448-453
ISBN: 978-989-758-358-2
Copyright © 2019 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

the increasing attention that privacy compliance has been receiving, through GDPR, this position paper suggests making software reuse explicitly aware of privacy compliance. More concretely, we suggest to extend VarMeR with analyses that are based on privacy considerations that reflect the "privacy levels" of the components recommended for reuse.

The rest of this paper is organized as follows. Section 2 reviews the relevant literature on privacy metadata (patterns and metamodels), while Section 3 briefly reviews VarMeR. Section 4 elaborates on the main contribution – a suggestion for privacy-aware software reuse. Finally, Section 5 summarizes and refers to the future work.

2 PRIVACY - PATTERNS AND METAMODELS

Privacy-by-design (Cavoukian, 2011) refers to the simple idea of embedding privacy within the design of a new technological system, rather than trying to fix problems afterwards, when often it is too late. To support privacy-by-design, Hoepman (2014) introduced eight privacy design patterns: minimize, hide, separate, aggregate, inform, control, enforce, and demonstrate. These patterns provide an implementation of the legal notion of data protection, bridging the gap between data protection requirements set out in law, and system development practice (Danezis et al., 2015). For instance, the most basic privacy design strategy is data minimization, which states that the amount of personal information that is processed should be minimal (Gürses et al., 2011). The realization of each privacy design pattern, however, requires using different privacy techniques, which challenges the fulfilment of the legal notion of data protection. For instance, privacy impact assessment approaches support realizing the minimize design pattern. Anonymization techniques are recommended to realize the aggregate design pattern, which states that personal information should be processed at the highest level of aggregation and with the least possible detail in which it remains useful. These eight privacy design patterns are compliant with the GDPR and can be considered as requirements for the design of privacy-by-design systems.

Privacy Level Agreements (PLAs) aim to standardize the way cloud providers describe their data protection practices. They are considered as a way to implement GDPR. D'Errico and Pearson (2015) suggest an ontology-based model for representing the information disclosed in PLAs in order to support different automatic analyses, such as service offering discovery and comparison. Diamantopoulou et al. (2017) presented a GDPR-based metamodel for PLAs to support privacy management, based on analysis of privacy threats, vulnerabilities and trust relationships in information systems, whilst complying with laws and regulations.

While most of the literature consider privacy on a technological level, some works refer to the organizational level. For instance, Feltus et al. (2017) introduced a privacy metamodel and discuss and demonstrate how the metamodel may support management of the privacy in enterprises involved in interconnected societies, by integrating the privacy metamodel with the systemic business ecosystem. Further related work on the organizational level can be found in Tom et al. (2018) where a GDPR model is introduced aiming to provide a simple, visual overview to aid process implementers in understanding the associations between different entities in the GDPR. The model is positioned as part of a larger approach for developing organizational privacy policies and extracting compliance rules.

All these approaches generally refer to "privacy awareness" while developing software. We suggest specifically percolating privacy concerns when making reuse decisions. Although such an approach can be applied on different reuse methods, we decided to apply it first to VarMeR, which goes beyond clone detection and feature mining, and actually recommends on reuse opportunities through three polymorphism-inspired reuse mechanisms.

3 VarMeR

VarMer (Reinhartz-Berger et al., 2015; Zamansky and Reinhartz-Berger, 2017; Reinhartz-Berger and Zamansky, 2018) follows a three step process, depicted in Figure 1. In the first step, the exhibited behaviors are extracted from the input artifacts (e.g., object-oriented code), each of which may belong to a different software product ($P_1...P_n$). A behavior is represented via two descriptors: shallow – which refers to the interface of the behavior, including its name, the parameters passed, and the returned type, and deep – which refers to the transformation done by the behavior to the state variables, manifested by the attributes used and the attribute modified.

In the second step, the extracted behaviors are compared using a similarity measure (e.g., based on semantic nets or statistical techniques (Mihalcea et al., 2006)). This way a similarity mapping between the behavior constituents (namely, parameters and returned type for the shallow descriptor and attribute used and attribute modified for the deep descriptor) is applied. Three cases are of interest:

1. USE – the similarity mapping is bijection (each constituent of behavior 1 has exactly one counterpart in behavior 2 and vice versa).
2. REF (abbreviation for refinement) – at least one constituent in behavior 1 has more than one counterpart in behavior 2.
3. EXT (abbreviation for extension) – at least one constituent in behavior 1 has no counterpart in behavior 2.

Based on the comparison results, the following polymorphism-inspired mechanisms can be recommended in the third step (see Table 1):

1. Parametric polymorphism for similar behaviors in terms of both shallow and deep descriptors.
2. Subtyping (inclusion) polymorphism for similar behaviors in terms of shallow descriptors and refined or extended behaviors in terms of deep descriptors.
3. Overloading for similar behaviors in terms of shallow descriptors and different behaviors in terms of deep descriptors.

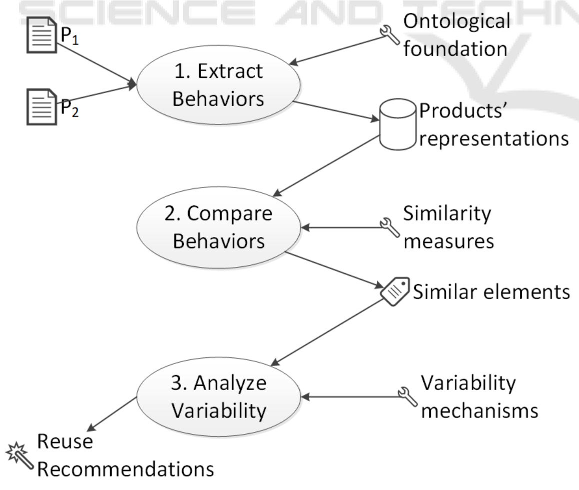

Figure 1: VarMeR process.

Table 1: Characteristics of Polymorphism-Inspired Mechanisms.

Shallow	**Deep**	**Polymorphism-Inspired mechanism**
USE	USE	Parametric
USE	REF	Subtyping
USE	EXT	
USE	REF+EXT	
USE	None	Overloading

The outcomes of VarMeR are visualized as graphs whose nodes are the products (or parts of them) and the edges represent the potential appropriateness of applying the different polymorphism-inspired mechanisms. Each product is visualized in a unique color. An example of VarMeR outcomes is shown in Figure 2. The components of two products (depicted in red and green) are compared.

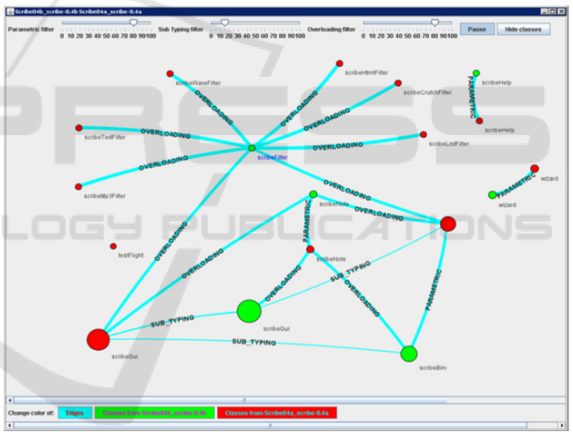

Figure 2: A snapshot of VarMeR comparing two products.

One important application of the VarMeR approach, described in (Reinhartz-Berger and Zamansky, 2018), is for supporting decisions concerning turning a set of potentially similar products into a product line, or in other words measuring product-line ability of a set of products (Berger et al., 2014). The idea is to consider various subgraphs of VarMeR as potential core assets, namely, reusable artifacts that can be used by different products in the line. This is done analyzing different characteristics: the number of products in the potential core assets (i.e., the number of colors in the sub-graph), and the number of parametric, overloading and subtyping relations.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

After setting the minimal number of products in a core asset to m, we define the ***m-color behavioral similarity degree*** of a sub-graph G'=(V', E') that is composed of at least m colors as a triplet (PS, SS, OS), where:

PS = $\frac{2P_{G'}}{k(k-1)}$ is the parametric similarity degree,

SS = $\frac{2S_{G'}}{k(k-1)}$ is the sub-typing similarity degree,

OS = $\frac{2O_{G'}}{k(k-1)}$ is the overloading similarity degree,

$P_{G'}$, $S_{G'}$, $O_{G'}$ are the numbers of parametric, subtyping and overloading edges in G', respectively,

k=|V'| is the number of nodes in the sub-graph.

The 3-color behavioral similarity degree of the sub-graph G_1 in Figure 3 is (1, 0, 0), indicating on a 3-colored "maximally parametric" asset. For G_2 the behavioral similarity degree is lower, (0.33, 0, 0.67), indicating on a "less parametric" and "more overloading" 2-colored asset.

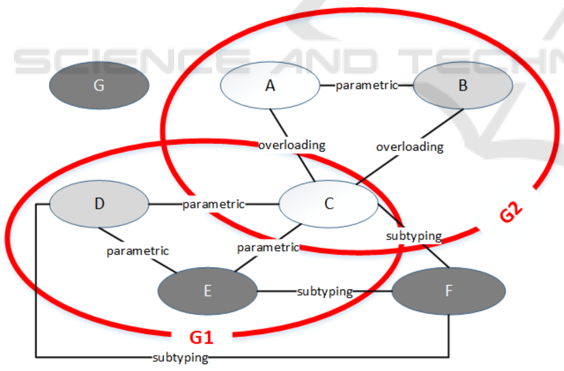

Figure 3: Example of product-line ability metrics.

4 PRIVACY AWARENESS

The main idea behind our suggested approach is preferring reuse of components that are more complaint with privacy regulations. To this end, we suggest refining the process described in Figure 1 as shown in Figure 4.

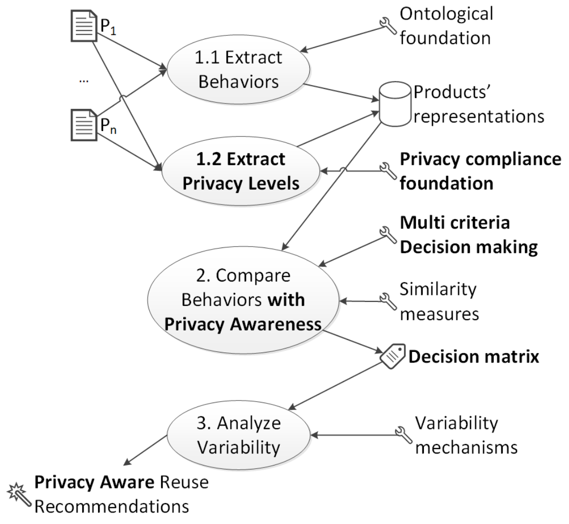
P1
...
Pn
1.1 Extract Behaviors
1.2 Extract Privacy Levels
2. Compare Behaviors with Privacy Awareness
3. Analyze Variability
Ontological foundation
Products' representations
Privacy compliance foundation
Multi criteria Decision making
Similarity measures
Decision matrix
Variability mechanisms
Privacy Aware Reuse Recommendations

Figure 4: Privacy-aware VarMeR process.

We first introduce the notion of *privacy levels* into the context of software reuse. Privacy level of a software component (class, operation, project, etc.) is metadata concerning the privacy compliance of this component. The documentation of privacy levels may be done manually, semi-automatically, or automatically (if appropriate tools are developed), and may follow any privacy compliance foundation (such as the design patterns and metamodels reviewed in Section 2).

We can then extend the VarMeR approach to address the information on privacy levels in the following way (see step 1.2 in Figure 4). Each node in VarMeR will exhibit privacy metadata – a vector of elements of the form p:x, where p is some privacy compliance principle, pattern, meta-element, or requirement, and x is its value for the certain node. x may be Boolean indicating whether the principle holds or not. Alternatively, it may be of some other well-ordered (and so comparable) type, indicating the degree of compliance. In the context of the eight privacy design patterns (Hoepman, 2014), the vector can look as follows: <minimize: true, hide: true, separate: false, aggregate: false, inform: true, control: true, enforce: true, demonstrate: true> (see Figure 5).

The challenge now is combining the information on privacy levels with existing analysis of VarMeR on behavioral similarity, leading to useful metrics that can guide reuse decisions (product-line ability in our case). Intuitively, we would like to be able to identify those subgraphs which represent similar enough elements, but also have high enough privacy levels. When focusing on behavioral considerations of similarity, an intuitive ordering is imposed on the similarity degrees: the more parametric the subgraph is, the more similarity it contains. However, with privacy levels involved, the ordering is not so trivial. This requires more sophisticated methods for weighing the different alternatives and choosing the most appropriate ones.

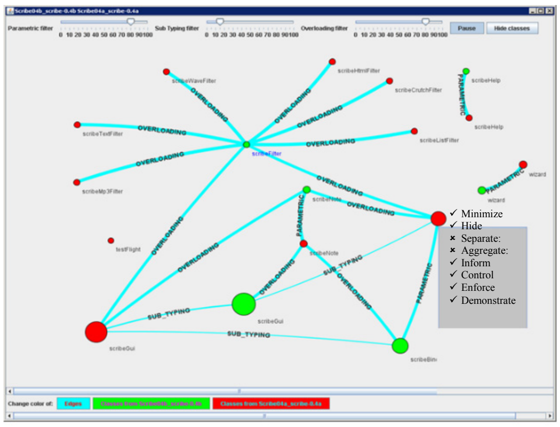

Figure 5: VarMeR equipped with privacy metadata.

One particularly relevant approach here is multi-criteria decision making (MCDM) (Velasquez and Hester, 2013), which allows to explicitly weigh and evaluate multiple non-comparable criteria in decision making. A typical outcome of MCDM is a decision matrix shown in Figure 6. The matrix is constructed after given m different alternatives that should be judged against n chosen criteria. The cell xij in the matrix holds the evaluation given to alternative i with respect to criterion j, and is usually computed according to a weight assigned to each criterion.

$$\begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_m \end{bmatrix} \quad \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

Figure 6: The structure of a multi-criteria decision matrix.

In the context of our problem, the above approach can be applied as follows (see step 2 in Figure 4). The alternatives $A_1,\ldots,A_m$ are the different sub-graphs which can potentially form core assets in a product line. The criteria they are judged against can be related to behavioral parameters (e.g., PS, OS, SS as defined above), as well as criteria reflecting privacy levels. Just to give a simple concrete example, consider Figure 7 which adds privacy levels on top of Figure 3. Assume p1, p2, p3 are three privacy design patterns, e.g., minimize, aggregate, and hide, respectively. + indicates following the pattern and – indicates violating it.

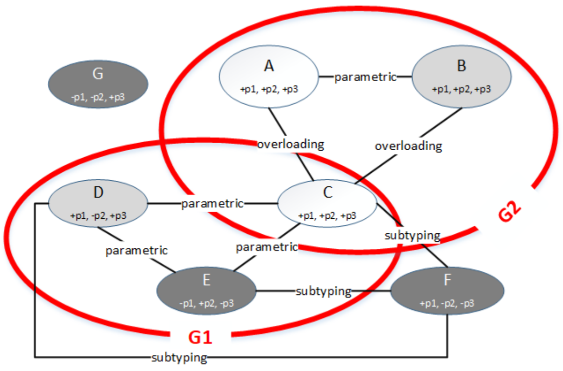

Figure 7: Example of privacy levels.

Suppose that the privacy level of the sub-graph is computed by applying a logical AND operation on all of its nodes. Then the decision matrix in Table 2 is obtained.

Table 2: Example of a multi-criteria decision matrix.

	PS	SS	OS	P1	P2	P3
G1	1	0	0	0	0	0
G2	0	0.33	0.67	1	1	1

We can see that while on the behavioral level G1 is rated higher than G2, taking privacy considerations into account changes the picture. This simple example demonstrates how reuse decisions may be affected by privacy considerations and justify the need for the privacy-aware software reuse suggested and demonstrated in this paper.

5 SUMMARY AND FUTURE WORK

The research fields of software reuse and of privacy have so far been going in orthogonal directions. In this position paper we have proposed a concrete framework for combining the two fields of research. This combination is of increasing importance due to the continuous GDPR efforts, which bring about dramatic changes in the way software will be developed in organizations, while at the same time, software reuse practices are increasing both from organization-proprietary repositories, and from open source repositories, where the tracking of privacy meta-data is even more challenging.

The idea behind the proposed framework is to integrate reuse decisions made on the basis of VarMeR's behavioral analysis of software artifacts with privacy considerations. More concretely, we demonstrated how taking privacy considerations explicitly into account can affect product-line ability decisions. This raises several interesting challenges for future research. First of all, we proposed the notion of privacy level meta-data – what it should include, and how it should be represented is a challenging question, in light of the fact that more and more privacy design patterns for software developers are emerging. We intentionally left the notion of privacy levels in this paper very abstract to open the door for discussions on the nature of this metadata.

Secondly, we envision the extension of VarMeR approach to the setting of software search and integration decisions, where again privacy considerations can be an important factor. To this end, a query language is needed to support querying a repository of software components and recommending on the most suitable ones in terms of behavioral similarity and privacy considerations.

To summarize, our goal here was to bring to attention the fact that privacy considerations matter for reuse decisions, and reuse decisions affect privacy compliance. This circle deserves further discussion, which will hopefully be started by this position paper.

REFERENCES

Assunção, W. K., Lopez-Herrejon, R. E., Linsbauer, L., Vergilio, S. R., & Egyed, A. (2017). Reengineering legacy applications into software product lines: a systematic mapping. *Empirical Software Engineering, 22(6), 2972-3016.*

Berger, C., Rendel, H. and Rumpe, B. (2014). Measuring the Ability to Form a Product Line from Existing Products. *Proceedings of the Fourth International Workshop on Variability Modelling of Software-intensive Systems (VaMoS).*

Cavoukian, A. (2011). Privacy by design in law, policy and practice. A white paper for regulators, decision-makers and policy-makers.

Danezis, G., Domingo-Ferrer, J., Hansen, M., Hoepman, J-M., Le Métayer, D., Tirtea, R., Schiffner, S. (2015). Privacy and Data Protection by Design - from policy to engineering. *CoRR, abs/1501.03726, arXiv.org/*

D'Errico, M., & Pearson, S. (2015, March). Towards a formalised representation for the technical enforcement of privacy level agreements. In *2015 IEEE International Conference on Cloud Engineering (IC2E) (pp. 422-427). IEEE.*

Diamantopoulou, V., Angelopoulos, K., Pavlidis, M., & Mouratidis, H. (2017). A Metamodel for GDPR-based Privacy Level Agreements. In *ER Forum/Demos (pp. 285-291).*

Feltus, C., Grandry, E., Kupper, T., & Colin, J. N. (2017). Model-driven Approach for Privacy Management in Business Ecosystem. In *MODELSWARD (pp. 392-400).*

Gürses, S., Troncoso, C. & Diaz, C. (2011). Engineering privacy by design. In *Conference on Computers, Privacy & Data Protection (CPDP 2011).*

Hoepman, J. H. (2014, June). Privacy design strategies. In IFIP International Information Security Conference (pp. 446-459). *Springer, Berlin, Heidelberg.*

Lim, W. C. (1994). Effects of reuse on quality, productivity, and economics. *IEEE software, (5), 23-30.*

Mihalcea, R., Corley, C., and Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. *American Association for Artificial Intelligence (AAAI'06), pp. 775-780.*

Rattan, D., Bhatia, R., and Singh, M. (2013). Software clone detection: A systematic review. *Information and Software Technology, 55(7), 1165-1199.*

Reinhartz-Berger, I., Zamansky, A., & Kemelman, M. (2015). Analyzing variability of cloned artifacts: formal framework and its application to requirements. In *International Conference on Enterprise, Business-Process and Information Systems Modeling (pp. 311-325). Springer, Cham.*

Reinhartz-Berger, I., & Zamansky, A. (2018). A Behavior-Based Framework for Assessing Product Line-Ability. In *International Conference on Advanced Information Systems Engineering (pp. 571-586). Springer, Cham.*

Tom, J., Sing, E., & Matulevičius, R. (2018, September). Conceptual Representation of the GDPR: Model and Application Directions. In International Conference on Business Informatics Research (pp. 18-28*). Springer, Cham.*

Velasquez, M., & Hester, P. T. (2013). An analysis of multi-criteria decision making methods. *International Journal of Operations Research, 10(2), 56-66.*

Zamansky, A., & Reinhartz-Berger, I. (2017). Visualizing Code Variabilities for Supporting Reuse Decisions. In *Symposium on Conceptual Modelling Education (pp. 25-34).*