# Reuse Considerations in Evolving Software Products: The Software Product Line Perspective

Iris Reinhartz-Berger[1], Amir Tomer[2] and Malki Grossman[2]

[1] University of Haifa, Haifa, Israel
[2] Kinneret Academic College on the Sea of Galilee, Jordan Valley, Israel
iris@is.haifa.ac.il, tomera@kinneret.ac.il, malkigr@gmail.com

**Abstract.** The evolution of software products is obtained by continuously expanding the variety of features, qualities, and functions of existing products. When similar software products are developed, two important reuse decisions are: (1) whether to develop a software product or adapt organizational software assets, commonly termed core assets; and (2) whether to develop a core asset or extract it from existing product artifacts, e.g., using mining techniques. While many works study how to reuse effectively and efficiently, the considerations taken when reaching such decisions are somehow overlooked or taken as intuitive or self-understood. To this end, we present the results of an exploratory study that investigates the engineering, organizational, and business considerations taking a software product line perspective.

**Keywords:** Reuse, Software Product Lines, Decision Making, Software Assets

## 1 Introduction

### 1.1 Background and Related Work

Software development is a demanding task, which deals with tradeoffs between requirements and resources. The competitive and rapidly changing environments require developing evolving, high quality software while minimizing resource. Noting that most software systems are not new but rather variants of systems that have been developed, software product line engineering [4], [5] is a key idea in software reuse.

In this context, studies have been conducted on development of core assets – reusable software artifacts or resources that are used in the production of more than one product in a product line. The development of core assets can be done up front (forward engineering, see for example the core asset development cycle in [4]), or through extracting existing artifacts (mining or reengineering, see [3] for a recent systematic mapping on methods for reengineering existing products into software product lines). Other studies tackle reuse of core assets to particular products, e.g., through variability mechanisms, which are implementation techniques to delay design decisions to the point in the development cycle where overall business goals can be optimized [8].

While the development of core assets is extensively studied, we found much less research on reuse decisions in creating and utilizing core assets. A decision support tool for assessing the maturity of the software product line process is introduced in [1]. The tool implements a fuzzy logic approach, whose inputs are in the form of questions that refer to the three software product line engineering cycles: core assets development, product development, and management. In [2], four interdependent software development concerns are identified: business, architecture, process, and organization. In [7], aspects that impact product line engineering feasibility decision and transition strategy selection are studied. For example, business motivation, expected Return-On-Investment, and connection with customers are mentioned in the business dimension.

The studies reviewed above deal with companies that adopt or in the process of adopting a software product line engineering approach. The model presented in [6] tackles a broader scope of reuse, referring to two levels: repository assets (which are similar by intention to core assets) and private assets (which are the adaptations to particular products, namely, the product artifacts). The model aims to assist in weighing and evaluating different reuse scenarios, based on accumulated organizational data. To this end, four transformation operations (within levels) and five transition operations (between levels) are introduced. The developers are expected to decide which reuse scenarios (i.e., sequences of elementary operations) to follow in a given situation. The decision considerations, however, are not explored.

### 1.2 Objectives and Structure

We aim to expand the scope of existing studies and analyze reuse decisions in companies that develop variants of similar products, but have not necessarily adopted software product line engineering. The nature of developing multiple products, with a significant degree of commonality among them, requires taking into consideration engineering, organizational, and business aspects. We concentrate on two important reuse decisions: (1) whether to develop a software product or adapt organizational software assets (namely, core assets); and (2) whether to develop a core asset or extract it from existing product artifacts, e.g., using mining techniques. We interviewed senior stakeholders in three companies and identified considerations relevant to these reuse decisions. It is important to note that we examined how reuse *is done* in these companies and not how it *should be done* or what *its contribution to the company* is.

The rest of the paper is structured as follows. In Section 2 we introduce our underlying conceptual framework. In Section 3 we elaborate on data collection and processing, while in Section 4 we present our results regarding decision considerations relevant to the different reuse decisions. In Section 5 we discuss benefits and implications, and, finally, Section 6 concludes and refers to future research plans.

## 2 The Underlying Conceptual Framework

The terms core assets and product artifacts are well established in software product line engineering [4]: *core assets* are software-related artifacts, e.g., architecture, software components, or requirements statements, which are managed with the intention of being reused (in different products in the line). *Product artifacts* are the specific software-related artifacts which are tailored to the needs of specific products or customers. Both core assets and product artifacts can be developed either from scratch or by reusing artifacts internal or external (e.g., from open source) to the developing company. We follow the idea presented in [6] that transition between these types of artifacts are allowed in both directions: core assets can become product artifacts through adaptation and product artifacts can become core assets through extraction, e.g., through mining techniques or manual changes. Figure 1 summarizes these observations.

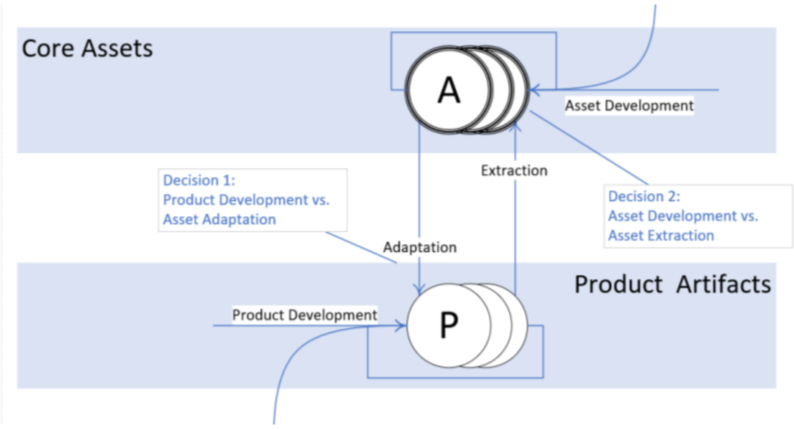

Figure 1. The underlying conceptual framework for reuse operations

This framework reflects two major reuse decisions:

1. Product Development vs. Asset Adaptation: Under what circumstances will asset adaptation be preferred over developing the product from scratch or through ad-hoc reuse of other product artifacts? Under what circumstances will product development be preferred over adapting core assets?
2. Asset Development vs. Asset Extraction: Under what circumstances will asset development be preferred over extracting existing product artifacts in order to create the core asset? Under what circumstances will core asset extraction be preferred over developing the core asset from scratch or through ad-hoc reuse of other core assets?

While the diversity in the considerations may be large and depend on different characteristics of the companies, the involved stakeholders, and the products to be developed, we claim that investigation and understanding of the variety of considerations and the decision-making processes are important for developing suitable methods and supporting tools. Thus, we conducted an exploratory research whose settings and results are described next.

## 3 Settings and Methods

The following research questions drive our work:

RQ1. What are the engineering, organizational, and business considerations relevant to decide whether to develop a product or adapt existing core assets?

RQ2. What are the engineering, organizational, and business considerations relevant to decide whether to extract a core asset from existing product artifacts or develop it?

To answer these questions, we interviewed six senior stakeholders in three companies which develop similar products to different customers or markets. Although the three selected companies are Israeli, they all operate in the global market. Therefore, we believe that the elicited considerations are not significantly biased by cultural attributes but are rather compliant with globally accepted development processes.

Each interviewee was separately interviewed by a subset of authors and all interviews were recorded. The companies' characteristics and details on the interviewee's roles are listed in Table 1. At this initial stage of the research, we did not want to lead the interviewees too much by our research questions, and therefore decided conducting unstructured interviews. Yet, all interviewees were asked to introduce themselves and the roles they had and describe the reuse processes they are familiar with.

Table 1. Companies and Stakeholders involved in the study

| **Company** | **Domain** | **Size** | **Reuse Processes** | **Interviewee's Roles** |
|---|---|---|---|---|
| A | Defense systems | Large (6 divisions) | Well established and followed | A1 – Software section R&D deputy<br>A2 – Project manager (previously software department manager) |
| B | Manufacturing support systems | Large (2 divisions) | Partially established and followed | B1 – Corporate chief systems engineer<br>B2 – Multidisciplinary development team leader<br>B3 – Product manager |
| C | Data integration & big data management | Medium | Partially established and followed | C1 – Executive vice president, R&D and global technical operations |

After the interviews, we followed the following procedure: (1) Each author transcribed the interviews of a specific company; (2) Each author extracted key sentences from the interviews (s)he transcribed. The key sentences referred to the decision points described above; (3) Each author extracted considerations from the key sentences (s)he extracted in the previous step; (4) Each author reviewed the outcomes of steps 1-3 of another author, such that each outcome was created by one author and approved by another; (5) Each author independently categorized *all* considerations according to the two reuse decisions (Product Development/Asset Adaptation, Asset Development/Asset Extraction) and their scope – engineering, organizational, and business; (6) The initial set of considerations was consolidated and refined throughout discussions till reaching the set of considerations proposed below.

# 4 Results: Reuse Considerations

Table 2 summarizes the engineering, organizational, and business reuse considerations we have found, while the rest of the section discusses them with respect to the two research questions presented in Section 3[1].

Table 2. Engineering, Organizational, and Business Reuse Considerations

| Decision | Aspect | Consideration |
|---|---|---|
| Product Development vs. Asset Adaptation | Engineering (Section 4.1.1) | Quality Requirements (of the Product) |
| | | Development Requirements (Time and Resources) |
| | | Extent of Required Adaptation |
| | Organizational (Section 4.1.2) | Developer Preferences |
| | | Extent of Success of previous Reuse Attempts |
| | | Management Decisions |
| | Business (Section 4.1.3) | Customer Characteristics |
| | | Product Characteristics (Functionality and Quality) |
| | | Competitor Characteristics |
| | | Technological Leadership |
| | | Profitability |
| Asset Development vs. Asset Extraction | Engineering (Section 4.2.1) | Technological Forecast |
| | | Maintainability |
| | | Extent of Similarity |
| | | Complexity of Variability |
| | Organizational (Section 4.2.2) | Knowledge Sharing Support |
| | | Resource Utilization |
| | | (Re)Use Forecast |
| | | Management Decisions |
| | Business (Section 4.2.3) | Customer characteristics |
| | | Product Characteristics |
| | | Technology Characteristics |
| | | Market Needs |

## 4.1 Product Development vs. Asset Adaptation (RQ1)

### 4.1.1 Engineering Considerations

The engineering considerations with respect to this decision point include quality requirements (of the product), development requirements (time and resources), and extent of required adaptation.

***Quality requirements.*** If the product has specific quality requirements, such as performance, asset adaptation may result in violation of these requirements. In such cases,

[1] Due to space limitations, citations taken from the interviews to support the different considerations can be found at: http://is.haifa.ac.il/~iris/research/Assets/InterviewsOutcomes.xlsx.

the decision whether to compromise the quality requirements or comply with them through development is highly relevant.

On the other hand, the way core assets are created, in some cases after being evaluated in several projects, may result in inherited high quality of the asset or other compensating characteristics that make asset adaptation worthy.

***Development Requirements.*** While adaptation of core assets aims to save resources and improve productivity, it may also raise challenges that negatively impact time and resources. Company A, for example, supports two working strategies: either you use the core asset as ***it is*** or you create a local variant. Interviewee B2 further warns against perceiving asset adaptation as time saving, perception that may turn to be misleading.

***Extent of Required Adaptation.*** The extent of adaptation may influence the decision whether to adapt a core asset or to develop the product without reusing core assets. As noted, company A does not consider adaptation, but rather adoption (i.e., 'black-box' reuse) of core assets. However, if there is adaptation that requires a lot of changes for fitting the asset into the specific product, asset adaptation may be less worthy.

### 4.1.2 Organizational Considerations

The organizational considerations with respect to product development vs. asset adaptation include developer preferences, extent of success of previous reuse attempts and management decisions.

***Developer Preferences***. The decision whether to develop a product or adapt existing core assets falls many times on the preferences and understanding of the developers. Developers who have developed the core assets have a commitment to use them rather than to develop new product artifacts.

***Extent of Success of Previous Reuse Attempts.*** Successful reuse of an asset ensures that it will be reused in the future. According to interviewee A1, her company follows a more structured two-step process: first, potentially reusable artifacts are locally handled and available to all developers. The moment these artifacts are used as they are in three projects, they are entered into the asset repository and get the whole shell that enables smooth adoption in future products.

***Management Decisions.*** Management may envision the benefits of asset adaptation and favor a clear preference to base the development of a new product on existing assets and to develop only in cases where there are no assets that meet the requirements.

### 4.1.3 Business Considerations

The business considerations with respect to product development vs. asset adaptation include customer characteristics, product characteristics, competitor characteristics, technological leadership and profitability.

***Customer characteristics.*** Customers differ from each other in many aspects, such as technical sophistication and understanding, financial dominancy, and seniority in the market. Such characteristics may lead the developers to prefer specific strategy: product development or asset adaption. For example, in a case of a powerful customer, who leads the market, new development with small amount of reuse may be considered. On the other hand, the decision might be negotiable, either with the customer or internally, in view of other benefits obtained from asset adaptation.

***Product characteristics (Functionality and Quality).*** Complying with customer requirements is a major business objective, since it directly relates to customer satisfaction. Trying to get as close as possible to the specified product requirements may lead to prefer development over asset adaptation. When specific features have, by nature, to be built for every specific customer, asset adaptation is not an option. Moreover, such features may be sub-contracted locally, as part of the entire deal.

***Competitor characteristics.*** Tough competitors, with technological and business seniority in the market, sometimes dictate the way according to which other companies act. Competitors may introduce new features, which may lead to preferring product development over asset adaptation. Moreover, influential competitors may convince the customer to include features on which they have advantages. This has an effect on the product characteristics. Competitor analysis may also point on inferiority in certain aspects, motivating asset adaptation – as a business strength.

***Technological leadership.*** On top of the last described case, where the company could find an advantage over its competitors, lies the case where the developing company is perceived in the market as a technological leader. This increases the motivation to adapt existing assets and to convince the customer that they are even better that what is needed.

***Profitability.*** The profitability of sales is based directly on product price-tag and quantity. The differences between product development and asset adaptation costs may become redundant, resulting in preference to develop products that best fit customer requirements. Such considerations are relevant to company B which develops manufacturing support systems. Company C on the other hand considers profitability a little bit differently – with respect to other customers. Thus, specific requests of individual customers will most likely not be addressed.

Sometimes profitability is achieved through technological leadership, by providing better value to the customer. This conception will increase the motivation to prefer asset adaptation over product development.

## 4.2 Asset Development vs. Asset Extraction (RQ2)

### 4.2.1 Engineering Considerations

The engineering considerations with respect to this decision point include technological forecast, maintainability, extent of similarity and complexity of variability.

***Technological forecast***. Technology obsolescence or anticipation of new technology may encourage asset development rather than relying on existing technology as reflected in existing products.

***Maintainability.*** Maintaining customizable core assets may increase cost. In these cases, extracting an existing product artifact is considered rather than developing core assets that fit different products but their maintenance is costly. However, development of core assets may result in high quality assets which underwent careful testing and debugging and their future maintenance is low.

***Extent of similarity.*** Products that exhibit similar functional features and have a common underlying architecture (satisfying similar non-functional requirements) are more likely to be used for extracting core assets. This consideration was mainly highlighted by the interviewees of company A, which follows well-structured reuse processes that promote direct adoption of assets.

***Complexity of Variability***. If the variability is large, it may be more difficult to extract the assets from the existing products and it may call for developing core assets with configurable parameters. Interviewee A1, for example, reported on the creation of a cross-division repository which holds high-level artifacts. Interviewee B1 also reported on a similar experience when aiming to reuse artifacts of similar products that are implemented in different operating systems.

### 4.2.2 Organizational Considerations

The organizational considerations with respect to asset development vs. asset extraction include knowledge sharing support, resources utilization, (re)use forecast and management decisions.

***Knowledge Sharing Support.*** Knowledge sharing in organizations is very important particularly when deciding on reuse: developers need to have some ideas regarding the artifacts and the assets that their company owns. Knowledge sharing within development teams that deal with the same project is quite common and there are formal and informal ways to achieve this, e.g., common repositories or team meetings. Knowledge sharing across departments, projects, or divisions is much more challenging and requires some interventions or policies. Such a support may encourage asset extraction.

***Resources Utilization***. Efficiency and the way resources are utilized are important factors of any organization. Asset development requires R&D activities, which are nonrecurring engineering operations. The decision whether and how much to invest in such activities is demanding and hence if such an opportunity for asset development pops up, companies (such as Company B) may be happy to utilize it.

***(Re)Use Forecast.*** Asset development, or even investment in asset extraction, are time and resource consuming and hence are worthwhile only if the core assets are used in different projects. We found evidence to this observation in all companies.

***Management Decisions.*** We already mentioned that the management can influence product-related reuse decisions. Its involvement is also important in asset-related decisions. Particularly, the management can encourage asset development in order to create a common infrastructure for future projects.

### 4.2.3 Business Considerations

The business considerations with respect to asset development vs. asset extraction include customer, product, and technology characteristics, as well as market needs.

***Customer characteristics.*** Although customers are not directly involved in decisions on core assets, their requirements or characteristics may indirectly influence decisions. Moreover, influential customers may request features that will be relevant to additional customers in the future, making asset development more beneficial. Customers with future vision may require all features at once ('one time investment'), encouraging the development of sustainable assets to be later adapted into new products.

***Product characteristics.*** The relatively expensive development of assets may be compensated by other product characteristics which will decrease the total cost. However, when the target product is intended for the market, and not specifically tailored for a customer, the considerations for asset extraction are much more complicated (e.g., higher functionality vs. lower cost).

***Technology Characteristics.*** Exploiting a technology, even prior to any specific product, is another drive for asset development. Assets are then in the form of feasibility studies or prototypes, which may later be adapted for specific products.

***Market needs.*** The forecast of market needs may lead to asset development, to tackle future needs in advance, or asset extraction, when capabilities of a specific product are predicted to be needed.

## 5 Benefits and Implications

The variety of considerations brought above shows that there is a need for a deep exploration of reuse-related decisions in order to understand what lies behind decision-making in this context. Decisions at two levels were explored: the project level – where a specific product is provided to a specific customer or market, and the infrastructure level – where core assets are managed for the benefit of all projects. Although these levels may be intertwined in organizations, decision-makers are usually situated in either of these levels.

The separation of concerns between the two levels is not trivial in some cases as it raises ‘the chicken or the egg causality’ dilemma. For example, when a new need emerges, would it be handled first at a specific project level, calling for product development specifically for the customer, and only later-on extraction into an asset? Or would it be handled in the infrastructure level up front, calling for developing an asset with generic and customizable features and then adapting it into the specific product, satisfying the customer requirements? It seems that in real life cases cycles of asset adaptation and asset extraction occur continuously.

We further observed that the engineering, organizational and business considerations may interrelate and influence each other. For example, a decision made on the basis of engineering considerations (e.g., *maintainability*) may be jeopardized by business aspects (e.g., *product characteristics/customer satisfaction*). This calls for developing a method for guiding each decision based on the different considerations and the relations among them. This method needs to be flexible and customizable, as we noticed differences among companies. Company B, for example, whose products are developed for markets rather than for individual customers, considers business aspects more than the other two companies. Company A, which operates in the defense domain, highlights the engineering aspects, since excellence in engineering is a major advantage in this domain. Company C, which operates in the civil industry market, emphasizes profitability. This observation may call for considering the company goals and visions in the supporting reuse decision method.

## 6 Conclusions and Future Research

In this paper, we introduce a set of considerations taken by developers and managers while making reuse decisions in the context of evolving products. We focus on two major types of decisions inspired by software product line engineering: whether to develop a product or adapt existing core assets and whether to extract a core asset from existing product artifacts or develop it. We conducted an exploratory study to investigate engineering, organizational, and business considerations relevant to these levels.

The research has some limitations that raise directions for future research. First, we interviewed only six stakeholders from three Israeli companies. While the variety of responses is quite large, we plan to build a questionnaire based on our observations and insights. This questionnaire will be distributed in different companies among different stakeholders, potentially increasing and refining the found considerations. It will further enable quantifying the relevant considerations. Second, we discussed each consideration separately. We intend to transform the considerations into a guiding, customizable method for reuse decision making, also addressing relations among considerations. A further step will be to develop a decision-support tool. Finally, we documented and categorized the reuse considerations as expresses by the interviewees. We plan to evaluate their quality and contribution to the company and integrate the outcomes into a conceptual reuse-related decision making framework.

## References

1. Ahmed, F., & Capretz, L. F. (2015). A decision support tool for assessing the maturity of software product line process. arXiv preprint arXiv:1507.06941.
2. America, P., Obbink, H., van Ommering, R., & van der Linden, F. (2000). COPAM: A component-oriented platform architecting method family for product family engineering. In Software Product Lines (pp. 167-180). Springer, Boston, MA.
3. 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, 1-45.
4. Clements, P., & Northrop, L. (2002). Software product lines. Addison-Wesley.
5. Pohl, K., Böckle, G., & van Der Linden, F. J. (2005). Software product line engineering: foundations, principles and techniques. Springer Science & Business Media.
6. Tomer, A., Goldin, L., Kuflik, T., Kimchi, E., & Schach, S. R. (2004). Evaluating software reuse alternatives: a model and its application to an industrial case study. IEEE Transactions on Software Engineering, 30(9), 601-612.
7. Tüzün, E., Tekinerdogan, B., Kalender, M. E., & Bilgen, S. (2015). Empirical evaluation of a decision support model for adopting software product line engineering. Information and Software Technology, 60, 77-101.
8. Van Gurp, J., Bosch, J., & Svahnberg, M. (2001). On the notion of variability in software product lines. In Working IEEE/IFIP Conference on Software Architecture, pp. 45-54.