# Improving Product Life-Cycle Cost Management by the Application of Recommender Systems

Simon Völker[1], Matthias Walter[2], and Torsten Munkelt[1]

[1] Hochschule für Technik und Wirtschaft Dresden,
Faculty of Informatics / Mathematics, 01069 Dresden, Germany
{simon.voelker, torsten.munkelt}@htw-dresden.com
[2] Technische Universität Dresden, Chair of Information Systems, esp. IS in Manufacturing and Commerce, 01069 Dresden, Germany
{matthias.walter3}@tu-dresden.de

**Abstract.** Shorter market cycles and growing competition require professional product life-cycle cost management for manufacturing companies. During a co-innovation workshop at SAP SE, we analyzed product-cost optimization applied by discrete manufacturers to identify corresponding deficits and requirements of product costing. We identified a significant lack of software support regarding cost optimization, especially in the early phases of the product life cycle. To improve information system support during early phases of product life cycle, this paper points out a new field of application for recommender systems. In detail, we compile the concept of a ready-to-use recommender system that aims at the improvement of product life-cycle cost management. This concept complements the expertise of experts by recommending how to further optimize product costs.

**Keywords:** Product life-cycle cost management, recommender systems, product-cost optimization, enterprise information systems

## 1 Motivation

Efficient cost management has become an indispensable success factor for the discrete manufacturing industry. One reason for the increased relevance of cost management is high competition in global markets in combination with shorter market cycles [1]. Companies need to sell their products at reasonable prices to ensure long-lasting economic success. Reasonable pricing is accomplished by optimal product costs (including such costs as development efforts, procurement costs and production costs) [2]. Therefore, the search for optimal product costs requires concepts that focus on the complete life cycle of a product from ideation to degeneration (Figure 1). Product life-cycle cost management (PLCM) is such a product-related, modern cost management concept, which addresses the challenges and weak points of classical product costing [3]. PLCM considers costs along the life cycle of a product by considering incurred costs in combination with the estimation of committed costs for the upcoming product life cycle [4]. With PLCM, cost analyses are already performed before start of

Multikonferenz Wirtschaftsinformatik 2018,
March 06-09, 2018, Lüneburg, Germany

production (SOP). This is important since almost 90% of the total product costs are committed before SOP (Figure 1) [3].

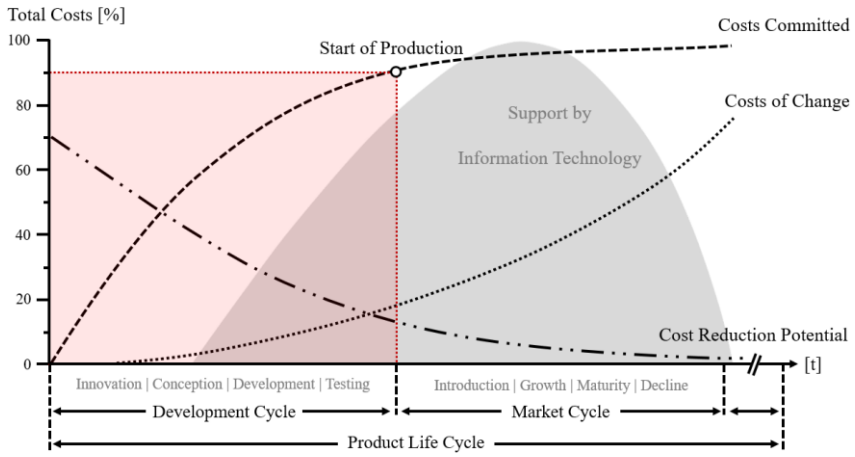


**Figure 1.** Cost reduction potential along product life cycle ([5], adapted from [6])

Although the product development phase is the most important phase to influence and, therefore, optimize product costs, early product-cost optimization seems to lack software support (Figure 1). While ERP systems support product costing for the market cycle, there seems to be potential for further software support during the product development cycle. In 2008, Schicker et al. [7] examined the status quo of PLCM in industry. In total, 91% of the participants requested improved information system support for product costing within product development [7].

Having identified this potential to improve information system support for early product-cost optimization in general, we joined a co-innovation development session at SAP SE. During this session, we gained insights into the current state of applied PLCM and its software support. As a result, we identified the need in industry to find software-supported solutions that assist costing experts in cost optimization to exploit optimization potentials of the product development cycle (Figure 1).

As part of a long-term research project, which was introduced by Walter and Leyh in 2017 [5], we initiated the research on the application of recommender systems (RS) to improve information system support for PLCM and moreover, to enhance early product-cost optimization. We formulated the following research questions to structure our research into the potential of RS as a feature for product-costing software:

1. What are essential domain-specific aspects and requirements for recommender systems to support product-cost optimization?
2. What would a recommender system conception look like and which components as well as functionalities would be required?

To answer these research questions, the paper is structured as follows: The next section provides a research background on recommender systems. Subsequently, we introduce our research methodology. Based on our findings, we derive domain-specific

requirements for RS in the context of PLCM. Furthermore, we elaborate core components of a potential RS in PLCM to draft a high-level architectural system concept to overcome hurdles of early product-cost optimization. The paper concludes with a summary of research results.

## 2 Background on Recommender Systems

Recommender systems are nowadays a ubiquitous part of the internet, well known to users of online services like Amazon or eBay. RS basically provide recommendations about items that the user may like to see [8]. Everybody is familiar with messages such as "Customers who bought this item also bought…" [9]. RS were created to help users orient themselves in the ever-increasing diversity of information, products and services available online. Recommendations for movies and videos (Netflix, YouTube), jobs (LinkedIn), or texts (GroupLens) are only minor subsets of existing scenarios [8, 10].

Methods for generating recommendations differ on the basis of the data analyzed [9]. One widespread approach of RS methodology is content-based filtering (CBF) [11, 12]. CBF identifies items to recommend based on their attributes. It assumes that if somebody likes many items with similar attribute values, then he or she will also like other items with similar attributes [8]. Take, for example, the case of a user who bought several books from the genre "fantasy". Based on this preference, a CBF system recommends other books from the same genre to this user. However, this approach can raise problems. A slightly different example could be that a user buys a printer. As people usually do not need multiple printers, recommending similar exemplars does not make any sense. Collaborative filtering (CF) is a method that overcomes this hurdle. CF assumes that if many people are interested in different items and some of those people are also interested in other items, then the rest of the people will also be interested in these other items [11, 12]. Recommendations in CF are mainly retrieved by analyzing users' behavior and the comparison of user profiles [10]. Now, consider again the printer example. The CF system knows that other customers who bought a printer usually also buy ink cartridges and printer paper. With that community knowledge, the system can recommend such complementary items.

The success of the two RS approaches mentioned above strongly depends on the amount of ratings. In both RS approaches, new users explicitly or implicitly need to rate at least a few items to identify the user's preferences (new user problem). In CF, ratings for new items are also required as the system cannot sense their appropriateness among the user base otherwise (new item problem). Beyond that, deep knowledge about users' preferences and constraints are not taken into consideration [13]. Hence, another approach was developed called knowledge-based (KB) RS that achieves better accuracy of predictions integrating specific domain knowledge about certain item features and a user's specific requirements [8]. KB RS emphasizes the user's situation and how recommendations can meet the particular need in that situation. Assessing the same knowledge sources as CBF and CF, KB RS takes further information into account (e.g., specific requirements of the user). Such additional information is typically retrieved by interacting with the user (e.g., interactive dialogs) [13]. For example, CB and CBF can hardly recommend items that are not bought frequently, whereas a

KB RS can do so. Consider the following use case. A user wants to buy an apartment. Through interaction with the user, a KB RS can request personal attributes (e.g., income, family status) and can thus generate a suitable recommendation based on these, whereas neither CBF nor CF would be able to provide reasonable recommendations due to lack of prior knowledge [12]. The knowledge needed to interact with the user must be engineered and then explicitly encoded into a formal and executable representation by domain experts. This initial process is called knowledge acquisition and represents a bottleneck when developing KB RS, as it demands a lot of effort [14].

Each of these RS approaches has its individual disadvantages (e.g., new user problems for CB and CBF, knowledge acquisition bottleneck for KB RS). Synthesizing single technologies to a hybrid recommender system is a widespread approach to achieve some synergy between them and, thus, reduce individual disadvantages [9].

## 3 Research Approach

The purpose of this paper is to elaborate a RS concept for product-cost optimization during product development. It is necessary to consider industry's state of the art to ensure problem relevance, as argued in Rosemann et al. [15] and Österle et al. [16]. At the same time, we wanted to confirm earlier research results from Schicker et al. [7] concerning a lack of domain-specific functionality within PLCM software. Such access to practical knowledge is important in the context of PLCM since product costing and its methods rely on expert knowledge (e.g., analogous cost-estimating techniques [17]) and, moreover, current research [18] has identified industrial practice as the most important source to learn about cost-optimization projects.

We chose the discrete manufacturing industry as reference industry due to product complexity and, therefore, the need for extended cycles to develop products such as automobiles, airplanes, or special machinery [19]. To acquire knowledge concerning the reference industry, we were able to join a co-innovation session at SAP SE, where potential and current SAP customers discussed business concepts and software requirements for the on-going development of SAP Product Lifecycle Costing [20].

In order to gain insights into the status quo and identify problems with product-cost optimization across the discrete manufacturing industry, we hosted a workshop with 19 experts (including 8 product controllers, 5 project controllers, and 4 information technology experts) from international companies from the automotive industry and the mechanical engineering industry [5]. The aim of this workshop was to capture tacit knowledge regarding the process of early product-cost optimization and to elaborate objectives to overcome drawbacks of today's information systems [7].
In a first run, all participants with costing-related competences (i.e., product controllers and project controllers) answered the following questions among others [5]:

1. How, when and why do you optimize product costs?
2. What problems do you face when optimizing product costs (especially related to software support)?
3. Where and to what end could the provision of dedicated recommendations improve the optimization of product costs?

The individual answers were presented to the other costing experts, before the main results were condensed into a joint summary. In the second run of the workshop, this summary was introduced to the whole group of experts, whereby every expert had a chance to challenge and discuss the group results. Industry insights and process deficits collected during this workshop were published in Walter and Leyh [5]. To further strengthen our understanding of the cost-optimization process and to elaborate use cases for optimization measures, we conducted a second study. This study contains an interview series followed by an evaluation with additional domain experts [21].

Based on this research results, we identified cost optimization use cases that offer potential to be supported by software-based approaches. These use cases have formed a foundation for our subsequent construction-oriented research approach. Taking elaborated use cases and requirements into consideration, we followed a requirement driven design process [22] to draft a conceptual approach of a RS in context of PLCM as initial design stage, and therefore to answer the second research question.

## 4 Findings

Initially, we performed a market sounding to gain an overview about information system support aiming at the support of PLCM in general. Examples of such software are SAP Product Lifecycle Costing [20], aPriori Product Cost Management [23], Siemens Teamcenter Product Costing [24], and FACTON EPC Suite [25]. While there is domain-specific software available on the market in general, industry experts are demanding further functionality to improve software-supported cost optimization. Due to a lack of specific functionality, participants confirmed that spreadsheet software is still being used for a variety of tasks regarding early cost optimization [5]. Therewith, we can confirm earlier research results from Schicker et al. in 2008 [7].

But before thinking about new approaches on how to improve software support, we wanted to understand the circumstances of the optimization environment. On the one hand, there exists a huge product complexity. Through our workshop we learned that product cost estimations for one product consist of up to 35,000 single items that require to be costed and optimized during product development. On the other hand, optimization processes are highly dynamic with a variety of coherent as well as contrary optimization measures each being manually evaluated and applied under increasing time pressures (e.g., to prepare customer quotations). Participants stated that quotations in discrete manufacturing industry – containing product cost estimates for upcoming decades – in some cases must be prepared in only 2 business days.

As our workshop practitioners are fully aware of the potential of product-cost optimization during the development cycle (Figure 1), we were able to jointly elaborate the main objectives to be addressed with future functional approaches to enhance information system support for cost optimization [5]:

1. Reduction of manual effort
2. An integrative adoption (into existing PLCM software) to avoid data inconsistency
3. Reduction of costing complexity by centrally accessing optimization measures

As a next step for our research it was necessary to identify potential use cases which a software-based approach must support. Since experts highlighted the importance of accessing optimization measures, we based our use case elaboration on the evaluation of applied measures among discrete manufacturing within our second study [21]. The result of this study by Walter et al. [21] shows a variety of optimization measures in relation to their relevance (scores from 0 to 10) for different branches (Table 1).

**Table 1.** Optimization measures being used during product development [21]

| *Optimization Measures* | *Average Score* | | | *Std. Dev.* |
|---|---|---|---|---|
| | *Overall* | *Automotive* | *Machinery Construction* | *Overall* |
| Alternative concept and product designs | 7.33 | 7.75 | 7.50 | 1.89 |
| Alternative reference components, assemblies, materials, and recipe ingredients | 7.06 | 7.00 | 7.00 | 2.66 |
| Alternative production plants | 7.29 | 7.82 | 7.75 | 2.37 |
| Alternative production processes and production process optimizations | 6.56 | 7.08 | 4.50 | 2.36 |
| Lot size and cycle-time optimizations | 4.88 | 5.80 | 1.75 | 2.74 |
| Material price optimization | 7.78 | 8.25 | 7.25 | 2.10 |
| Make-or-buy decisions | 7.83 | 7.92 | 8.25 | 1.57 |
| Investments in tools or equipment | 5.88 | 7.73 | 2.50 | 3.01 |
| Optimization of logistics costs | 5.61 | 6.58 | 4.25 | 2.69 |

# 5 Conception of a Recommender System for PLCM

## 5.1 Transform Optimization Measures into Recommendations

With the knowledge about optimization measures (Table 1) we were confronted with a variety of measures addressing different needs during product development projects each requiring a variety of different organizational stakeholders (e.g., purchasers, product engineers, process engineers, and logistic experts). Based on the main objective to centrally access optimization measures (see Section 4), we identified RS as one technique to fit our requirements. In particular, RS provide an easy, intuitive guidance for users (see Section 2) in a diversity of information. In parallel, we identified such a situation with the need to generalize access within a broad bandwidth of information such as 35.000 items to be costed during product development (see Section 4).

To develop an artifact that could serve as basis for iterative evaluation and development, we needed to elaborate an initial design stage for our construction-oriented research approach. How do we bridge the gap between typical RS-use cases and an optimization-use cases as described in Section 4? A simple make-or-buy recommendation could be: “Producing assembly X in plant B could lower production costs by 50 €”. The search for alternative items or assemblies can be turned into a recommendation as well: “In the past, users replaced assembly A by assembly B for product Y. For the current product X, this replacement would reduce total costs by 50 €. Apply recommendation?”. We want to seize common RS functionalities (Section

2) to generate recommendations. However, due to the complexity of cost-estimate structures, iterating all items within a cost estimate and checking for all available optimization measures is not a valid solution. Such an approach would easily lead to performance issues, an information overload for the user and, in the worst case, pure mathematical optimization problems. Therefore, we rely on existing approaches described in Section 2.

We want to offer global recommendations for the whole product cost estimate based on optimizations made in the past (exploit organizational knowledge) by applying CF. Context-sensitive recommendations, on the other hand, are triggered on the level of single-cost items, and are based on CBF. When the user navigates to a specific cost item within the costing estimate, possible optimization measures are validated corresponding to its attributes. If possible, the potential financial savings that can be achieved by accepting a recommendation should be displayed to enhance the user's ability to make faster decisions on evident recommendations. In such optimization recommendations and therefore in the application of RS, we see an opportunity to challenge the high time pressure during product development (Section 4).

### 5.2 Requirements toward Recommender Systems in PLCM

In our requirement analysis, we noticed that the application of a RS in the context of PLCM relates to domain-specific requirements. At first, we collected universal requirements valid for any kind of RS, based on a literature review on evaluation criteria and RS properties [8, 26]. The list of requirements addresses quality characteristics such as a high prediction accuracy, an item coverage, serendipity, or diversity. Furthermore, the knowledge from our findings (Section 4) helped us to identify requirements which are particularly relevant for a RS in PLCM. We value the following requirements as the most critical ones to continue our construction-oriented research approach in the given domain:

- The process of generating recommendations should be transparent for the user in order to increase confidence in the information system support.
- The system should be capable of learning and thus improving the quality of recommendations.

During the innovation sessions, the experts stated their concerns about the feasibility of recommendations made by a system, as they are responsible for providing fully traceable and reasonable cost estimates – which can be ruined by calculations based on erroneous assumptions [5]. Therefore, it is necessary to enhance the recommendations by utilizing detailed information about its accomplishment. Comprehensible explanations are a success factor to increase the transparency of the purpose of the system as well as users' satisfaction and their confidence in the system [27].

In Section 2, we stated that KB RS can gain specific domain knowledge through user interaction to increase recommendation quality. By enabling users to reject recommendations, we want to follow this approach. Users should provide additional information about the reasons for their decisions. Consider the following example. The RS recommends production of an item in *plant A* instead of *plant B*. The user knows

through his or her expert knowledge that it not possible to produce the item in *plant A* (e.g. because of strategic plant utilization). To retrieve the expert's knowledge, the system proposes a set of possible answers. A conceivable set could be: *"It is not possible to produce item X in plant A. Please specify the reason for the rejection: [ ] strategic plant utilization, [ ] local content restrictions], [ ] other: …"*. The system transforms the answers into knowledge content so that the newly acquired knowledge can steer further recommendation generation processes.

### 5.3 Essential Aspects of a Recommender System in PLCM

Based on the theoretical foundations of RS (Section 2) and the knowledge gained during the requirements analysis, we derived a concept of an RS for PLCM. We worked out a set of main aspects that needed to be included to provide a basic functionality. Besides basic functionality, further aspects were considered to improve the quality of the RS. When we had elaborated these aspects, we transformed them into abstract software components from which we deducted a high-level architectural concept.

As declared in previous sections, recommendations made by our system are based on relevant product cost-optimization measures (Table 1). These measures are collected within a dynamically extended catalogue of optimization measures. We distinguish between global recommendations based on CF and context-sensitive recommendations based on CBF (Section 4). The CBF recommendations are generated in real time for particular items or assemblies that the user is currently working on. Focusing on item characteristics and attributes (e.g., production plant, supplier, or product characteristics), our system validates possible optimization measures and, in the case of success, transforms them into recommendations. Consider the following example. While using PLCM software, a product costing expert selects *assembly X*, which is planned to be produced in *plant A*. By analyzing corresponding data, our system validates that *assembly X* can also be produced in *plant B* or *plant C*. As a result, our system recommends alternative production plants (if the impact is advantageous). Besides those attribute-related CBF approaches, CF is considered for retrieving our global optimization recommendations. Due to logging and analyzing the product costing expert's optimization activities, additional recommendations can be derived for other experts. For example, when a set of screws is replaced in one product's cost estimate, those screws may also be replaced in another product. To support CF use cases, a logging module should be implemented to analyze historical data. Global recommendations are shown directly when a user opens a cost estimation structure of a product. To do so, the setup of a database for frequently retrieved recommendations is essential for our system.

As companies are, in terms of processes and structures, heterogeneous constructs, the need for configurable software is an essential requirement [28]. The quality of RS can be increased through utilization of interactive KB RS (Section 2) [13]. Related to this interactivity, the systems' ability to learn is another important aspect. From the perspective of a product-costing expert (as of any other user), it is unsatisfactory to get a similar set of non-applicable recommendations repeatedly. Hence, a system needs to learn by user interaction which recommendations are value-adding and which are

not [29]. The demand for such behavior leads to a configurable and teachable system based on knowledge-based recommender technologies [8]. Implementing this functionality satisfies the need for a system with ongoing learning capability (Section 4). The use of KB RS in combination with CBF and CF makes our system a hybrid RS (Section 2).

Multiple recommendations can influence each other when they exist at the same time [30]. Imagine a certain material is required to produce an assembly and two recommendations are made: the first (*recommendation A*) recommends purchasing the item from an external supplier, while the second (*recommendation B*) recommends the production in a different plant. What is the system supposed to do with *recommendation B*, when the user accepts *recommendation A*, or vice versa? Accepting *recommendation A* does not automatically imply the invalidity or change of relevance of *recommendation B*. Rejecting all other recommendations for the selected item may lead to a loss of optimization potential. Therefore, interdependencies between recommendations have to be validated and resolved.

### 5.4 High-level Architecture of a Recommender System in PLCM

To visualize the concept of an RS that improves PLCM, we deliberated over a component-based high-level architecture (Figure 2). Driven by the idea of transforming optimization measures into recommendations we elaborated our architecture model by translating former deliberated aspects (Section 5.1) into software components. We enriched these domain-specific components by RS modules as suggested by Imran et al. [31] and synthesized all components into a holistic model. Furthermore, the architecture is expanded by user roles including their interaction correlations.

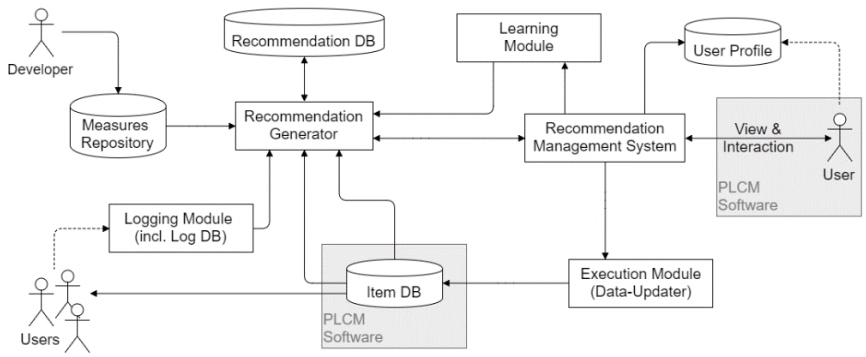


**Figure 2.** Component-based high-level architecture of a recommender system in PLCM

The directional arrows in Figure 2 indicate the direction of data access, whereby dashed lines indicate not explicit access, but implicit access gained by interpreting the user's behavior. Our system is primary managed by the Recommendation Management System (RMS). It manages the interaction with the users. The user interface must be adaptable for available PLCM software as elaborated in Section 4. We suggest

separating the presentation logic and the control logic of the RS to ensure the interchangeability of the user interface depending on the underlying PLCM software. The independence of the presentation layer can be achieved by following the architectural pattern Model-View-Controller (MVC) [32]. The RMS forwards feedback retrieved from the user (explicit or implicit) to the Learning Module. This module includes a knowledge database and is used to improve recommendation quality through KB RS functionalities. One conceivable approach is reinforcement learning, where the learning effect is achieved by the repetitive interpretation of positive (applied recommendations) or negative (rejected recommendations) feedback [33]. When a recommendation, which includes proposed data changes, is accepted, the RMS also initiates the application of optimization measures at the Execution Module. The database of PLCM software serves as the Item Database, where all cost-estimate items are stored. It includes all relevant cost items of any product-cost estimate. The data layer should also be separated from the presentation logic and control logic by MVC to enable integration into different PLCM tools. Corresponding data updates are initiated by the Execution Module over provided interfaces. The RMS requests the recommendations, which are shown to the user from the Recommendation Generator (RG). The RG generates recommendations based on information from several different data sources. Digital representations of available optimization measures are stored in the Measures Repository, which can be extended dynamically. The RG transforms these measures into recommendations integrating CBF and CF algorithms. Furthermore, the RG takes information from the Learning Module in account to enhance the recommendation quality. Historical data about optimization activities are stored in and retrieved from the Logging Module, which is used to generate global recommendations. Corresponding recommendations are continuously updated in the Recommendation DB to be available to the user at any time.

## 6 Conclusion and Future Work

Product development is the most important phase to leverage optimization potentials and to ensure overall economic success. Therefore, the lack of information system support in early product-cost optimization is astonishing, but has been confirmed by a variety of business experts from the discrete manufacturing industry. Product-costing activities during the market cycle are covered by ERP systems. Main product-costing activities within the development cycle are nowadays supported by costing software. However, optimization of product costs is still impeded by enormous amounts of data, which need to be evaluated under increasing time pressures. Optimization operations are indeed applied manually by experts. In this article, we suggest the application of a recommender system to supplant manual efforts and improve product-cost optimization. This RS generates recommendations relating to optimization measures that are relevant for the industry.

Following a comprehensive analysis of common scientific and domain-specific requirements of an RS within PLCM, we identified essential aspects that need to be considered for future implementations. Our approach follows the concept of hybrid RS. It is a necessarily configurable approach supporting heterogeneous organizations within

discrete manufacturing industries. Furthermore, the integration of learning mechanisms is indispensable to improving recommendation quality gradually. We transformed these aspects into a component-based high-level architecture (Figure 2). Transforming this architecture into a recommendation system is addressing major objectives to overcome hurdles in today's product-cost optimization, especially for large cost estimates that are required to be optimized within a limited period of time. This concept should not replace, but complement the expertise of experts by recommending possibilities to further optimize product costs.

Further research in this area should concentrate on the iterative development of artifacts like proof of concepts to evaluate the proposed system concept [34]. What could be taken into further consideration is the integration of additional optimization dimensions besides costs, such as ecological aspects, risks, and quality. Moreover, attention should be paid to sophisticated information presentation for recommendation details in order to heighten both the transparency and the traceability of the system to the users.

## References


1. Joos, T.: Controlling, Kostenrechnung und Kostenmanagement - Grundlagen - Anwendungen - Instrumente (in German). Springer Gabler, Wiesbaden (2014)
2. Georg, S.: Cut! Rezepte für ein wirkungsvolles Kostenmanagement (in German). Vahlen, München (2016)
3. Hansen, D.R., Mowen, M.M.: Cost Management Accounting and Control. Thomson South-Western, Mason (2006)
4. Schild, U.: Lebenszyklusrechnung und lebenszyklusbezogenes Zielkostenmanagement (in German). Deutscher Universitäts-Verlag/GWV Fachverlage GmbH, Wiesbaden (2005)
5. Walter, M., Leyh, C.: Knocking on Industry's Door: Product Cost Optimization in the Early Stages Requires Better Software Support. In: IEEE 19th Conference on Business Informatics. CBI 2017, Thessaloniki (2017)
6. Eigner, M., Stelzer, R.: Product Lifecycle Management: Ein Leitfaden für Product Development und Life Cycle Management. Springer, Heidelberg (2009)
7. Schicker, G., Mader, F., Bodendorf, F.: Product Lifecycle Cost Management - Status quo, Trends und Entwicklungsperspektiven im PLCM - eine empirische Studie (in German). In: Arbeitspapier Wirtschaftsinf. II (02/2008). Universität Erlangen-Nürnberg, Nürnberg (2008)
8. Ricci, F., Rokach, L., Shapira, B.: Recommender Systems Handbook. Springer, New York (2015)
9. Burke, R.: Hybrid Web Recommender Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds) The Adaptive Web. LNCS, vol. 4321, pp. 377-408. Springer, Berlin, Heidelberg (2007)
10. Klahold, A.: Recommender Systems - Grundlagen, Konzepte und Lösungen (in German). Vieweg+Teubner, Wiesbaden (2009)
11. Mettouris, C., Papadopoulos, G.A.: Ubiquitous recommender systems. In: G.A. Computing 96 (3), 223-257. Springer, New York (2014)
12. Mandl, M., Felfernig, A., Teppan, E., Schubert, M.: Consumer decision making in knowledge-based recommendation. In: J Intell Inf Syst (2011) 37(1), pp 1-22 (2011)
13. Felfernig, A., Burke, R.: Constraint-based Recommender Systems: Technologies and Research Issues. In: ICEC '08 Proceedings of the 10th international conference on Electronic commerce, Innsbruck, Article 3. ACM, New York (2008)

14. Felfernig, A., Friedrich, G., Jannach, D., Zanker, M.: Constraint-Based Recommender Systems. In; Ricci, F., Rokach, L., Shapira, B. (eds.): Recommender Systems Handbook. Chapter 5, pp. 161-190. Springer, New York (2015)
15. Rosemann, M., Vessey, I.: Toward Improving the Relevance of Information Systems Research to Practice: The Role of Applicability Checks. In: MIS Quarterly, 32(1), 1-22 (2008)
16. Österle, H., Otto, B.: Consortium Research. Bus Inf Syst Eng. 2, pp. 283-293 (2010)
17. Asiedu, Y., Gu, P.: Product Life Cycle Cost Analysis: State of the Art Review. International Journal of Production Research. 36(4), 883-908 (1998)
18. Mörtl, M., Schmied, C.: Design for Cost - A Review of Methods, Tools and Research Directions. Journal of the Indian Institute of Science. 95(4), 379-404 (2015)
19. Roda, I., Garetti, M.: TCO Evaluation in Physical Asset Management: Benefits and Limitations for Industrial Adoption. In: Grabot, B., Vallespir, B., Gomes, S., Bouras, A., Kiritsis, D. (eds.) Advances in Production Management Systems 2014. Proceedings, Part III, pp. 216-223. Springer, Berlin (2014)
20. Vosough, Z., Walter, M., Rode, J., Hesse, S., Groh, R.: Having Fun with Customers: Lessons Learned From an Agile Development of a Business Software. In: Proceedings of 9th Nordic Conference on Human-Computer Interaction (NordiCHI'16). ACM, New York (2016).
21. Walter, M., Leyh, C., Strahringer, S.: Knocking on Industry's Door: Needs in Product-Cost Optimization in the Early Product Life Cycle Stages. Complex Systems Informatics and Modeling Quarterly (CSIMQ). 13, 43-66 (2017)
22. Braun, R., Benedict, M., Wendler, H., Esswein, W.: Proposal for Requirements Driven Design Science Research. In: Donnellan, B., Helfert, M., Kenneally, J., VanderMeer, D., Rothenberger, M., Winter, R. (eds) DESRIST 2015: New Horizons in Design Science: Broadening the Research Agenda. Proceedings, pp. 135-151. Springer, Cham (2015)
23. aPriori Product Cost Management, https://www.apriori.com/products (Accessed 06.11.2017)
24. Teamcenter Product Costing, https://www.plm.automation.siemens.com/de/products/teamcenter/product-cost-management/ (Accessed 06.11.2017)
25. FACTON EPC SUITE, https://www.facton.com/software/facton-epc-suite (Accessed 06.11.2017)
26. Wu, W., He., L., Yang, J.: Evaluating recommender systems. In: Seventh International Conference on Digital Information Management ICDIM 2012. IEEE, Macau (2012)
27. Tintarev, N., Masthoff, J.: A Survey of Explanations in Recommender Systems. In: ICDEW '07 Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering Workshop, pp. 801-810. IEEE Computer Society, Washington (2007)
28. Quéva1, M., Männistö, T., Ricci, L., Probst, C.W.: Modelling Configuration Knowledge in Heterogeneous Product Families. In: Proceedings of the IJCAI 2011 Workshop on Configuration, pp. 9-16. CEUR Workshop Proceedings, Aachen (2011)
29. Rubens, N., Elahi, D., Sugiyama, M., Kaplan, M.: Active Learning in Recommender Systems. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 809-846. Springer, New York (2015)
30. Ziegler, C.N.: Social Web Artifacts for Boosting Recommenders. Springer, Cham (2013)
31. Imran, H., Belghis-Zadeh, M., Chang, T.W., Kinshuk, Graf, S.: A Rule-Based Recommender System to Suggest Learning Tasks. In: Trausan-Matu S., Boyer K.E., Crosby M., Panourgia K. (eds) Intelligent Tutoring Systems. ITS 2014. Lecture Notes in Computer Science, 8474, pp. 672-673. Springer, Cham (2014)
32. Sommerville, I.: Software engineering, 9th Edition. Addison-Wesley, Boston (2011)
33. Portugal, I., Alencar, P., Cowan, D.: The Use of Machine Learning in Recommender Systems: A Systematic Review. Expert Systems with Applications. 97, 205-227, in press (2017)
34. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly. 28(1), 75-105, MIS Research Center, Minneapolis (2004)