Association for Information Systems
AIS Electronic Library (AISeL)

Research Papers ECIS 2018 Proceedings

11-28-2018

A Product-Service System Configurator for Repurposing used Electric Vehicle Batteries

Johannes Voscort
University of Muenster, johannes.voscort@uni-muenster.de

Markus Monhof
University of Muenster, markus.monhof@ercis.uni-muenster.de

Jörg Becker
University of Muenster, becker@ercis.uni-muenster.de

Follow this and additional works at: https://aisel.aisnet.org/ecis2018_rp

Recommended Citation
Voscort, Johannes; Monhof, Markus; and Becker, Jörg, "A Product-Service System Configurator for Repurposing used Electric Vehicle Batteries" (2018). *Research Papers*. 67.
https://aisel.aisnet.org/ecis2018_rp/67

This material is brought to you by the ECIS 2018 Proceedings at AIS Electronic Library (AISeL). It has been accepted for inclusion in Research Papers by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

A PRODUCT-SERVICE SYSTEM CONFIGURATOR FOR REPURPOSING USED ELECTRIC VEHICLE BATTERIES

Research paper

Voscort, Johannes, University of Münster, Münster, Germany, johannes.voscort@uni-muenster.de

Monhof, Markus, University of Münster, Münster, Germany, markus.monhof@ercis.uni-muenster.de

Becker, Jörg, University of Münster, Münster, Germany, becker@ercis.uni-muenster.de

Abstract

High initial costs are one of the major obstacles for the diffusion of electric vehicles (i.e., electric cars). The repurposing of used electric vehicle batteries (EVBs) is seen as a possibility to lower the total costs of ownership of the vehicles and therefore to foster electric car sales. However, the decision how to repurpose used EVBs requires severe information system (IS) support. In particular, in this article, the configuration task of providing used EVBs as product-service systems (PSSs), which would hardly be possible without IS support, is addressed.

Configurators are a type of information systems that is suited to provide the required support. However, current instantiations of configurators for PSSs are scarce. An existing prototype is available but has certain limitations. Notably, it is only suitable for product-centric PSSs. Therefore, we set out to extend the configurator and present the design of a configurator that is capable of assisting the configuration of other types of PSSs (i.e., use-oriented and result-oriented). The configurator is instantiated and evaluated based on requirements derived from the literature and domain knowledge. While currently the configurator is specific for the domain of repurposing used EVBs it can be adapted to other domains.

Keywords: product-service system, decision support, second-life, design science research.

1 Introduction

The diffusion of electric vehicles (EVs) promises to reduce carbon dioxide emissions (Hagman et al., 2016). However, although the sales figures of EVs are increasing, the demand is still limited (Klör, Beverungen, et al., 2015). One of the reasons are the high initial costs of EVs, which are highly influenced by the production of the electric vehicle battery (EVB) that accounts for up to 40 % of the total manufacturing costs (Casals et al., 2017). Moreover, because of degradation effects, an EVB is no longer usable in an automotive application after about eight years and therefore should be replaced (Heymans et al., 2014). At this point, the EVB still has around 80 % of its initial capacity left. The reduced capacity can be recognized by the driver of the EV through the reduced range that can be driven with a fully charged battery in comparison to a new battery. Considering that the automotive use is very demanding for the battery, it can be further used as a power source in less demanding second-life applications (Ahmadi, Fowler, et al., 2014), even though it is no longer usable in the vehicle,. Examples for such second-life applications are the use as buffer storage for electrical energy generated from photovoltaic panels to increase self-consumption (Bräuer et al., 2016) or for grid-stabilization (Knowles and Morris, 2014). By repurposing and further using an EVB, its residual value increases, which can lead to a decreased total cost of ownership (TCO) of an EV (Conti et al., 2015) and hence to rising sales figures of EVs. In general, used EVBs can be utilized in the same applications as comparable new batteries. Therefore, there are some challenges for the effective marketing because potential customers have to be convinced to choose a used battery instead of a new one. Likely, the main argument in favor has to be the lower costs for the customer. However, there might be other factors like environmental benefits that are valued by the customers. Another challenge is the degradation of a used EVB that is hard to estimate. Thus, potential customers are not able to assess the quality of the offered battery (Bräuer et al., 2016) . This poses the risk for the customer of getting a low-quality battery, which following the Lemon Market Theory might lead to market failure (Beverungen, Klör, et al., 2015). To address this risk, the seller has to signal the quality (Spence, 1973), e.g., by providing certificates or warranties (Klör, Beverungen, et al., 2015). Another possibility is to offer the used EVBs as product-service system (PSS) consisting of the battery and additional services (Bräuer, 2016), since for some types of PSSs the risk remains at the seller (Stoppel and Roth, 2015; Tukker, 2004). Example for such types are leasing or usage-based PSSs. Additionally, PSSs can provide additional value to the customer (Stoppel and Roth, 2015) and therefore provide further incentives. Because of the expected high cost-pressure and the need to distinguish from competitors by providing individual solutions, information system (IS) support in the form of a PSS configurator is needed to minimize the effort and costs for the seller (Bräuer, 2016). This is necessary to enable a repurposing business. However, configurators for PSSs are scarce in general, and currently, there is no configurator that enables other types of PSSs' except the selling of product-service bundles (Monhof, 2018). Therefore, the research question examined in this paper is: *How to design an information system assisting a human decision maker in configuring PSSs for used EVBs?*

The paper is structured as follows: In the next section, the research background on repurposing used EVBs, knowledge-based configuration, and the constraint satisfaction problem are presented. Afterwards, the design of the configurator is provided and discussed. The article ends with a conclusion and research outlook.

2 Research Background

2.1 Product-Service Systems for Repurposing used EVBs

EVBs suffer from a continuous degradation (Conti et al., 2015). Hence, they should be replaced when the capacity drops below 80 % of its initial value, because at this point the vehicle's maximum range is significantly reduced and likely no longer accepted by customers (Beverungen, Klör, et al., 2015). While the speed of the degradation depends on several factors (e.g., battery type, temperatures, load profiles (Monhof et al., 2015)), the replacement is generally supposed to happen after about 150,000 km (Hawkins et al., 2013) or about eight years of usage (Cready et al., 2003). Furthermore, there are technological reasons why a battery should be replaced. Current lithium-ion batteries suffer from highly increased degradation rates after a certain capacity loss. Depending on the cell chemistry and type, this point lies between 70 % and 80 % of the initial capacity (e.g., Bach et al., 2016). Therefore, to maximize its lifetime, the battery should be removed from the electric vehicle before this threshold is reached. Because EVBs contain valuable materials like lithium, nickel, and cobalt, recycling could be an option to handle EVBs at their end of life (Fischhaber et al., 2016) and retrieve at least some of the materials. However, currently the recycling of EVBs is uneconomical (Ahmadi, Yip, et al., 2014) and not ecological (Ahmadi, Fowler, et al., 2014). Therefore, postponing the recycling by repurposing the EVBs in less demanding second-life applications is promising as the batteries still perform reasonably (Ahmadi, Fowler, et al., 2014).

For a successful marketing, it has been suggested that the used EVBs should be offered as PSSs (Bräuer,

2016) instead of selling them as pure products. PSSs are sets of physical products and services that fulfill the particular requirements of specific customers (Baines et al., 2007; Mont, 2002). In the literature, different types of PSSs are distinguished. Tukker (2004) identified the three categories *product-oriented*, *use-oriented*, and *result-oriented* of PSSs. While product-oriented PSSs are focusing product sales with some additional services, in use-oriented PSSs the product remains the property of the provider and is only made available to the customer. Examples for use-oriented PSSs are leasing or renting of products (Tukker, 2004). In result-oriented PSSs, the customer only buys a previously agreed on output, and it is the responsibility of the seller how to provide it. Examples are pay per service unit or outsourcing (Tukker, 2004).

For repurposing used EVBs, result- and use-oriented PSSs are especially suited because the batteries remain the property of the providers and therefore the risks for the customers are reduced (Bräuer, 2016). One reason why used EVBs should be offered as PSSs is the asymmetric information between the seller and potential customers that can hinder the transaction (Bräuer, 2016). For example, warranties should be added that signal a certain degree of quality and therefore, reduce the risk for the customer. Besides the services that signal the quality of a repurposed EVB or deal with the risk of failure, other services such as transportation or even product extensions that might be necessary for a safe operation or that provide additional value to the customer can be added (Cready et al., 2003; Gohla-Neudecker et al., 2015). Offering configurable PSSs, allows to customize the product to fit the individual requirements of the second-life application and customer. Likely for the repurposing and marketing of used EVBs specialized companies will emerge (Klör, Beverungen, et al., 2015)–the so-called *second-life manufacturers*. They will collect the EVBs from the automotive and battery manufacturers, which in the EU are legally required to take the old batteries back from the customers, and configure PSSs for the specific customers' needs. The services that are contained in the PSSs will be provided by different services providers that have to be selected during the configuration (Klör, Beverungen, et al., 2015). Because of the individual degradation of each EVB (Klör, Bräuer, et al., 2015) and the different requirements of each second-life application, the matching of used EVBs to a second-life application is a complex task that should be supported by a decision support system (DSS) (Beverungen, Klör, et al., 2015). Therefore, a model-driven DSS was designed that supports the matching of used EVBs and second-life applications (Klör, Monhof, et al., 2017a; Klör, 2017). The system supplier can use such a system to find a used EVB that fits the requirements of the customer's second-life application. Furthermore, the second-life manufacturer is responsible for the configuration of the PSS, which–among others–includes the enrichment with value-added services to reduce information asymmetries (Bräuer, 2016; Klör, Beverungen, et al., 2015). A first prototype of a configurator for PSSs in the domain of repurposing EVBs was presented in Klör, Monhof, et al. (2017b) as seven design principles and a method based on the generic decision process from Simon (1977). However, as stated in Monhof (2018), it has the limitation that only product-centric PSSs (i.e., sales) are supported, and other types of PSSs are neglected.

2.2 Knowledge-based Configuration

The shift of the manufacturing paradigm, from mass production to mass customization, in the late 1980s led to rising research interest in configuration problems (Hadzic and Andersen, 2004). The goal of mass customization is to produce individually customized products while keeping costs on the level of standardized, mass-produced goods (Pine et al., 1993). This can be achieved by using a generic architecture and standardized interfaces at the component level to allow the individual combination of components (Sabin and Weigel, 1998). However, having hundreds or thousands of configurable components increases the possibility of errors because of invalid combinations and leads to delays as well as further costly iterations (Sabin and Weigel, 1998). Therefore, information systems that support the configuration process are crucial to the success of mass customization. Configurators are information systems that are used to generate complete configurations automatically or to validate configurations (Stumptner, 1997) and thus to support a decision maker with during a configuration task. Such a configuration task has two essential characteristics: First, a solution (e.g., product or PSSs) is configured from a fixed and well-defined set of components (Mittal and Frayman, 1989; Sabin and Weigel, 1998). Second, the way in which the components can be combined is predetermined (Sabin and Weigel, 1998).

In the literature, different paradigms of configurators are presented (Sabin and Weigel, 1998). The most common ones are *rule-based*, *model-based*, and *case-based*. Rule-based configurators use production rules in the form *if condition then consequence* (Felfernig et al., 2014; Sabin and Weigel, 1998), which contain both the domain knowledge and the control strategy (Sabin and Weigel, 1998). However, rule-based configurators have two main drawbacks: First, the lack of separation of control and domain knowledge that leads to an enormous maintenance effort (Sabin and Freuder, 1996; Stumptner, 1997). Second, the order of the production rules can have an influence on the result of the configuration (Felfernig et al., 2014).

Model-based configurators can be used to overcome this drawbacks (Felfernig et al., 2014). Several approaches for model-based configurators exist (Sabin and Weigel, 1998), such as resource-based, constraint-based, and logic-based. Resource-based configurators use a producer-consumer model (Wang et al., 2009) in which, in contrast to other model-based approaches, no explicit representation of relations between the domain objects exists (Hotz and Krebs, 2003). Instead, the exchange of resources between objects can be seen as implicit relations (Hotz and Krebs, 2003). For each object, the amount of resources it consumes or provides is stated (Heinrich and Jüngst, 1991). The configuration starts with the consumption of the first resource (Hotz and Krebs, 2003) and the final solution is found when all demands are fulfilled and every consumed resource can be balanced with an appropriate resource provision (Hotz and Krebs, 2003; Sabin and Weigel, 1998). The constraint-based approach has been proven to be a promising solution due to its flexibility and generality (Felfernig et al., 2014; Wang et al., 2009). The idea of using a constraint satisfaction problem to create a generic and domain independent model was proposed by Mittal and Frayman (1989). The proposed approach consists of a fixed and pre-defined set of components and constraints (Wang et al., 2009). By only using components, domains, and constraints it is possible to formulate a constraint satisfaction problem of the configuration task. In contrast, logic-based models are based on description logic (Sabin and Weigel, 1998) that has three fundamental elements (Sabin and Weigel, 1998): *individuals* representing objects, *concepts* representing sets of individuals, and *roles* representing binary relations between individuals. Further, *constructors* like *and*, *or*, *at-least* and *all* enable the formulation of complex, composite descriptions. The clear semantics and the use of simple logic operations are advantages of the logic-based approach (Sabin and Weigel, 1998).

The case-based configurator paradigm differs fundamentally from the rule- and model-based (Sabin and Weigel, 1998) because the knowledge is incorporated in records (cases) that have been configured in the past. The approach relies on the assumption that similar problems have similar solutions (Sabin and Weigel, 1998). However, the main advantages of this approach are the low maintenance and creation efforts since no rules or constraints need to be acquired (Sabin and Weigel, 1998).

2.3 Constraint Satisfaction Problem

It has been shown that the constraint satisfaction problem is a powerful approach to model and solve configuration problems (Hadzic and Andersen, 2004; Mittal and Falkenhainer, 1990). A constraint satisfaction problem is defined as sets of variables, domain sets, and constraints (Hadzic and Andersen, 2004; Mittal and Falkenhainer, 1990). Each variable has an associated and non-empty domain set that identifies the set of feasible values. Furthermore, each constraint is defined over some finite and non-empty subset of variables and specifies the valid combinations of the variable's domains. To get a valid configuration each variable needs to be assigned to a domain, and all constraints have to be satisfied. All assignments that satisfy this definition are referred to as *solution tuples*. The set of all solution tuples is referred to as *solution space* while the set of all possible assignments is called *search space*.

Because of its fixed nature, the classical constraint satisfaction problem has some drawbacks. For example, if it should be possible that none of the given domains have to be selected, an additional arbitrary *None* domain needs to be added (Felfernig et al., 2014). Also, as it is not possible to add variables dynamically, all variables need to be instantiated and are part of the search space. To overcome this drawback, the dynamic constraint satisfaction problem was introduced, which allows "including or removing of variables from a potential solution" (Mittal and Falkenhainer, 1990, p. 25) based on previously identified variables (Mittal and Falkenhainer, 1990). This is achieved by adding activity constraints. Thus, the constraint satisfaction problem terminology is extended by the concept of *activity* of variables, which for each variable can be either active or inactive. Inactive variables are not considered for the final solution and *activity constraints* state the conditions under which variables become active. Furthermore, a valid solution needs to satisfy the following two criteria: The variables in the solution tuple satisfy the activity constraints and no proper subset of a solution tuple is a valid solution.

To identify valid solutions, two general paradigms of search strategies exist (Tsang, 1999): *systematic search strategies* and *repair strategies* (also called heuristic strategies). Systematic search strategies are characterized by the fact that they search step by step in the complete search space to find a valid solution. This guarantees to find a solution or prove that the constraint satisfaction problem is not solvable. However, a drawback of this kind of approaches is that they can be very time-consuming for large search spaces (Barták, 2003; Tsang, 1999). A common example of systematic search is the *backtracking* algorithm (Barták, 2003). In contrast, repair strategies start with a (possibly random) start solution regardless whether any constraints are violated or not. Afterwards, the assignments that violate any constraints are resolved. In the literature, several heuristics and meta-heuristics were presented that are suitable to find valid solutions (Tsang, 1999) in acceptable time. But it is possible that a non-optimal or even no solution might be found.

3 Research Method

To achieve the research objective a design science research (DSR) (Hevner et al., 2004; March and Smith, 1995; Nunamaker Jr. et al., 1990) approach based on the process from Peffers et al. (2007) is used. Because the presented configurator is based on an existing one, the performed research can be seen as the second iteration, which in line with Hevner (2007), forms a relevance cycle. The design of the artifact was informed by knowledge from the literature on existing configurators that was used to derive requirements. Further requirements were elicited using user stories developed as part of a consortium-research project (Österle and Otto, 2010) with experts from battery research, electrical engineering, and logistics. Based on the requirements the configurator was designed and implemented as a prototype. A thorough demonstration and evaluation of the designed artifacts is a major requirement for scientific rigor in every DSR project (Peffers et al., 2007). Therefore, the utility of the configurator can, for example, be demonstrated through case studies. However, this is not presented in this paper in detail due to space limitations. Instead, a criteria-based evaluation was performed (Venable et al., 2016). Even though, an empirical evaluation would be preferred, it is hitherto not possible because of the emerging nature of the field the application lies way into the future and the necessary realistic data and users are not available. In the DSR research contributions framework from Gregor and Hevner (2013) the presented solution can be positioned as exaptation research.

DSR projects can be classified regarding the maturity of the domain and of the solutions (Gregor and Hevner, 2013). The solution maturity of this paper is rather high as the domain of configurators and configuration problems is well researched (Felfernig et al., 2014). In contrast, the repurposing business of EVBs is still an innovative research field (Klör, Beverungen, et al., 2015). Furthermore, different types of PSSs or pricing schemes are currently not considered in the scientific literature on configurators. Therefore, the performed research classifies as *exaptation* research, in which known solutions are adopted for a new, complex domain. Here, constructs and methods from the field of knowledge-based configuration are adopted for the domain of repurposing used EVBs.

4 Design of the Configurator

In line with the literature on repurposing used EVBs, the presented configurator is supposed to be used by an employee of a company that is specialized on the repurposing of used EVBs. In the following, the decision process implemented in the configurator, the requirements towards, and components of the configurator are presented.

4.1 Price models

To integrate different types of PSSs into the configurator, price models are introduced and can be specified and used in the configurator. At this point, price models use a cost-based pricing strategy (Guilding et al., 2005; Peattie and Charter, 2003) and therefore incorporate the different cost objects (i.e., costs for additional services or product extensions). Furthermore, different rules can be defined how those costs are distributed over different revenue types (e.g., one-time payment, recurring payment, pay-per-unit). Additionally, margins can be defined to provide rough estimates of revenues that can be expected. Thus, during the configuration process, a price model is selected to define the type of PSS and later to compare different variants of PSSs.

4.2 Configuration Process

The core functionality of the presented configurator is the configuration of PSSs for specific customer needs that result from the second-life applications. The necessary configuration process implemented in the configurator is based on the process presented in (Klör, Monhof, et al., 2017b) and consists of seven process steps (Figure 1). The main difference lies in the introduction of an additional step for selecting the price model.

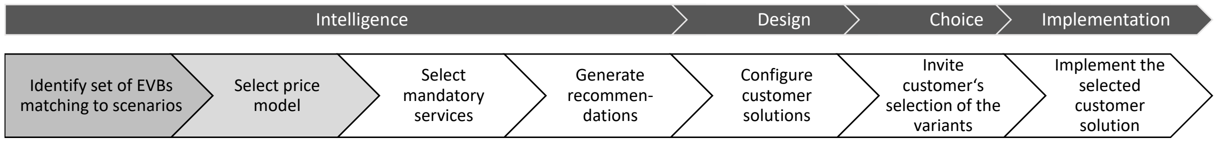

Intelligence
Design
Choice
Implementation
Identify set of EVBs matching to scenarios
Select price model
Select mandatory services
Generate recommen-dations
Configure customer solutions
Invite customer's selection of the variants
Implement the selected customer solution

Figure 1. Decision process adapted from (Klör, Monhof, et al., 2017b)

The first step is the selection of one of the battery-scenario assignments, which for exmaple have been identified with the help of a DSS. Afterwards, an appropriate price model that fits the customer's needs has to be selected by the user. It is important that this step is done before the PSS is configured as the price model can influence the configuration, especially because it can draw additional constraints that have to be considered. Furthermore, depending on the price model, additional parameters like the leasing terms need to be provided by the user. The remainder of the process is similar to the original process. Different variants can be configured for the assignments of used EVBs to second-life applications. For this, mandatory services are selected, and recommendations for additional services are provided by a recommender system as presented in (Klör, Monhof, et al., 2017b). Mandatory services can be defined in the knowledge-base for different contexts, e.g., for specific battery types or second-life applications. For example, transportation services are required for applications of consumers (B2C) because of legal obligations resulting from the classification of lithium-ion batteries as dangerous goods. At the end of the configuration process, the selected services and product extensions including the corresponding costs are summarized. Based on the selected price model the prices are also shown, which allows an estimation of the profit. Depending on the selected price model a few input parameters are needed, to calculate the costs and revenues over the complete lifespan of the PSS. In the case of a leasing price model, no inputs are needed as the leasing term has to be stated in the price model selection step. In contrast, a selling price model requires a parameter that contains the expected usage time to consider the number of repetitions of recurring services. A usage-based price model additionally requires a parameter that contains the amount of, e.g., kWh that are expected to be used, to be able to calculate the generated revenue. Independently of the selected price model, it must be considered that only the number of repetitions of recurring services that will be performed during the stated calculation period should be included in the calculation. For example, assuming a usage period of 23 months, yearly conducted services would be executed only once and therefore should be considered only once in the profit calculation. Finally, it must be considered that the calculated profit does not represent reality as general expenses, e.g, for administration are missing. Additionally, the calculation is based on some assumptions, and hence, the result serves in the first place as a benchmark for the salesperson.

4.3 Requirements of the Configurator

In the literature the term “product configurator” is mostly used without explicitly stating that products are the only valid application area (e.g., Colace et al., 2009; Stegmann et al., 2003; Tiihonen and Soininen, 1997; Trentin et al., 2012). In contrast, some authors explicit mention that a configurator can be used to configure products and services (Felfernig et al., 2014; Haug, 2007; Trentin et al., 2011). Shen et al. (2012) stresses that bundling services into one customer-specific service is comparable to a product configuration task. Hence, in the following general insights are derived from existing literature on configurators, summarized, and stated as requirements that are used for the design and implementation of the configurator. The derived requirements (section 4.3) are grouped in three categories: configuration requirements (CRs), constraint based requirements (CBRs) and usability and design requirements (UDRs). Five CRs define the basis of the configuration process. Since constraints, which define how components can be combined (Mittal and Frayman, 1989), are the foundation of every configuration problem as they two additional requirements are derived. Finally, nine UDRs regarding the design and usability are stated.

The hitherto derived requirements are of generic nature and could apply to most configurators. Therefore, additional requirements resulting from the domain and envisioned application are identified. With the help of user stories, more specific requirements can be gathered as they allow to express software requirements in a lightweight way (Cohn, 2005). A *user story* is a short description of functionality or feature from the perspective of an individual user role. Thus, to create user stories, first user roles need to be defined. A role is modeled by defining the particular role attributes (Cohn, 2004). A role attribute is a fact about a user that defines the role or distinguishes it from another user role. Some standard role attributes are the expertise in the domain, the frequency of use of the software, general computer and software skills, and the goal of using the software. In the following, the three identified user roles are introduced:

Marketing employee: The marketing employee is an expert in the pricing domain. He/She knows which pricing schemes fit best to the business strategy and how to set the pricing of the products. The marketing employee only uses the software to adapt or check the price models that are contained in the system.
Salesperson: The salesperson perfectly knows the customers and which price model fit best to them. He/She uses the configurator frequently to configure PSSs that satisfy the customer needs.
Knowledge engineer: The knowledge engineer has in-depth knowledge of the configurator application and configuration technologies in general. Further, he/she already gathered in-depth knowledge of the product and service domain from domain experts. The knowledge engineer uses the software to enter or update the configuration rules.

Based on the user roles, nine user stories are defined that especially address the pricing and revenue streams.

Requirement	Description
CR1 - Predefined Sets	The configuration is based on a predefined set of components (Mittal and Frayman, 1989).
CR2 - Constraints	Valid combinations of components are stated in the form of constraints (Mittal and Frayman, 1989).
CR3 - Inference	During the configuration process only components that lead to a valid configuration can be selected (Hadzic and Andersen, 2004).
CR4 - Completeness	During the configuration process all components that lead to a valid configuration should be displayed (Hadzic and Andersen, 2004).
CR5 - Restoration	It is possible to deselect selected components. After a component is deselected, the valid components are redetermined (Hadzic and Andersen, 2004).
CBR1 - Check constraints	Each new constraint is validated whether there are one or more conflicting constraints as it is not possible to get valid configurations in the case of a conflict (Felfernig et al., 2014).
CBR2 - Exclude/Require	A constraint can either exclude components or require them (Blecker et al., 2004).
UDR1 - Overview	At the end of the configuration process an overview of the selected components is presented to the user (Leitner et al., 2014).
UDR2 - Progress bar	A progress bar that indicates the current configuration step to the user should be contained. The progress bar is displayed on a horizontal plane and is clearly visible (Leitner et al., 2014).
UDR3 - Logical Clustering	The presentation of components should follow a logical clustering (Leitner et al., 2014).
UDR4 - Costs	The cost of each component should be clearly visible during the configuration (Leitner et al., 2014).
UDR5 - Shopping Cart	A list of selected components is clearly visible during the configuration process (Leitner et al., 2014).
UDR6 - Total Costs	he total costs of all selected components are clearly visible (Leitner et al., 2014).
UDR7 - Response Time	Having an interactive configuration, the response time of selecting a component and getting an answer should not take more than 250 milliseconds (Hadzic and Andersen, 2004).
UDR8 - No order	The user should not be forced to select the components in a predefined order (Hadzic and Andersen, 2004).
UDR9 - Starting Point	Users should have the possibility to start the configuration with default configurations (Leitner et al., 2014).

Table 1. Generic requirements on configurators derived from the literature.

4.4 Problem Formulation as Constraint Satisfaction Problem

For the design of the configurator, the constraint satisfaction problem approach is chosen because of its flexibility and generality (Felfernig et al., 2014; Wang et al., 2009). These characteristics are of particular importance as some conditions of the emerging repurposing business of used EVBs are still unknown. The case-based paradigm is discarded because no historical data to validate configurations exists. The rule-based paradigm would be a viable option but is dismissed because of the high maintenance effort caused by the lack of separation of domain and control knowledge (Stumptner, 1997).

As stated before, a constraint satisfaction problem is defined as a set of variables, domains, and constraints. Components that can be selected during the configuration process are either *services* or *product extensions*.

User Story	Description
US1:	As a salesperson, I want to see the actual costs of product extensions and services during the configuration so that I know which and when costs occur.
US2	As a salesperson, I want to have a parametrizable profit calculator so that I can estimate the profits of configured PSS for different scenarios.
US3:	As a salesperson, I want to be able to compare different price models applied to the same configured PSS so that I can make the best offer to a customer.
US4:	As a marketing employee, I want the configurator application to contain a price model editor so that price models can be entered directly without any programming effort.
US5:	As a marketing employee, I want the price model editor to validate modeled price models so that no invalid models can be added to the system.
US6:	As a marketing employee, I want to be able to restrict in the price model editor the EVB types and scenario categories a model can be applied to so that the price models can be fitted to specific EVB types and scenario categories.
US7:	As a knowledge engineer, I want the configurator application to contain a configuration rule editor so that configuration rules can be entered or updated in the graphical user interface without the need for any programming effort.
US8:	As a knowledge engineer, I want the price models to be categorized according to the revenue stream type so that configuration rules can easily be applied to all price models of the same type.
US9:	As a knowledge engineer, I want that a configuration rule can have multiple triggers and targets so that it is, for example, possible that a configuration rule is only active if two components are selected during the configuration.

Table 2. User stories for configuring PSSs in the domain of repurposing used EVBs.

Services can be assigned to categories (Klör, Monhof, et al., 2017b), like transportation, maintenance, or warranty. An actual service of the service category transportation could, for example, be *standard delivery* or *express delivery*. By using this differentiation, the services easily fit into the constraint satisfaction problem terminology. Scenario categories like transportation and maintenance are represented as variables, whereas services like express delivery or annually maintenance are represented as domains of the related variable. The same approach can be used to add product extensions. However, because not every category has to be selected, either a dynamic constraint satisfaction problem approach or adding a none domain to each variable is needed.

Previously it has been described how services and product extensions can be represented in the knowledge base. Apart from that, additional data might be needed to create meaningful configuration rules for the domain. For example, a configuration rule that requires a specific product extension based on a particular battery type could be useful. Therefore, all battery types are added to the domain knowledge. For this, each battery type is represented as a domain of one battery variable. The same is done for the price models and the scenario categories. A mobile second-life application, for example, could require a water-resistant cover and a usage-based price model an extended battery management system (BMS) to monitor and log the usage. Adding price models to the constraint satisfaction problem model could be done in the same way. However, it is reasonable to categorize price models, as stated in US7, to enable the creation of configuration rules on revenue stream type level. Hence, for each revenue stream type, a variable is created. Afterwards, a price model can be assigned as a domain to the corresponding revenue stream type. Eventually, a none domain needs to be added to each variable as otherwise, a price model of each revenue stream type has to be selected to get a valid configuration.

Finally, the scenario categories have to be added to the knowledge base. In contrast to the other data, scenario categories are hierarchically structured and can have multiple levels, which makes it more challenging to fit the structure in the constraint satisfaction problem definition. However, as a starting point, a variable that represents a virtual parent scenario category is created. Afterwards, the highest scenario categories are added as domains. These are stationary, semi-stationary, and mobile applications (Beverungen, Bräuer, et al., 2017). Next, a new variable, which represents the stationary applications, is created and the corresponding child categories are added as domains. This step is repeated for all scenario categories that have child categories. The virtual parent variable is needed as it is possible to have second-life applications that are only categorized on the highest level. Until now, all scenario categories can be added to the knowledge base, but the hierarchical structure is missing. For example, adding a configuration rule that has the scenario category mobile application as a trigger, should also be executed if any child scenario category of mobile applications is selected. To achieve the intended behavior, structural configuration rules (Wang et al., 2009) are introduced. With the help of constraints, it is possible to require or exclude specific variables (CBR2). The intended result can be achieved by adding constraints that link each child with its parent category. Hence, the constraints need to force the selection of the parent domain as soon as the child domain is selected. As a result, whenever a child domain is selected, the parent domain also has to be selected and therefore the constraints of the parent category are triggered.

4.5 Components of the Configurator

The presented configurator consists of several components that are described in the following. They are based on and extend the configurator presented in (Klör, Monhof, et al., 2017b), which consists of a configuration rules engine, a service recommender system, a variants configurator, a task list, and a customer interface.

The main component is the *configuration component* that incorporates the configuration process. In this component, the configuration task is represented as a constraint satisfaction problem. The user can select different services and product extensions for the specific customer needs of the second-life application and the selected battery. Invalid configurations are prohibited using the constraints formulated in the knowledge base of the constraint satisfaction problem. For this, the basic configuration rules engine from (Klör, Monhof, et al., 2017b) is replaced by the presented formulation as constraint satisfaction problem that can be solved using standard solvers. This is supposed to achieve better scalability through reduced maintenance effort as well as run-time savings. Furthermore, the recommender system presented in (Klör, Monhof, et al., 2017b) is integrated as is to provide additional decision support by recommending different variants of PSSs that are likely to be bought by the customer.

Because the user might want to configure and compare different alternatives for specific second-life application of a certain customer, the *variant manager* is included. It is needed to depict the whole sales process, from making the configuration, sending it to the customer, and receiving an answer. All created offers of configured PSSs can be found in the variant manager. The salesperson decides which offers will be sent to the customer. After sending them to the customer, the customer gets a notification with a link to the customer interface. The customer interface allows the customers to compare the offers and accept or reject them. The second functionality of the variant manager is to apply different price models to a configured PSS. This functionality can be useful if the salesperson wants to provide multiple offers of the same configured PSS based on different pricing schemes (US 3). For example, this can be used to compare different revenue streams like selling or leasing the PSS with certain terms.

For the configuration process, certain data is needed. Especially, the knowledge base of the constraint satisfaction problem has to be populated. Therefore different administrative components are included. Since it is inefficient to hard-code the constraints into the control strategy, an interface that can be used to maintain the configuration rules is included (US6). In the following, this interface is called *configuration rule editor*. Nevertheless, well-skilled employees like knowledge engineers are still needed to create meaningful configuration rules. In the same fashion, a *price model editor* is included to allow the modeling of different pricing alternatives.

The configurator was implemented as a web solution using .NET Windows Communication Foundation as back-end and Angular[1] as front-end framework.

5 Evaluation

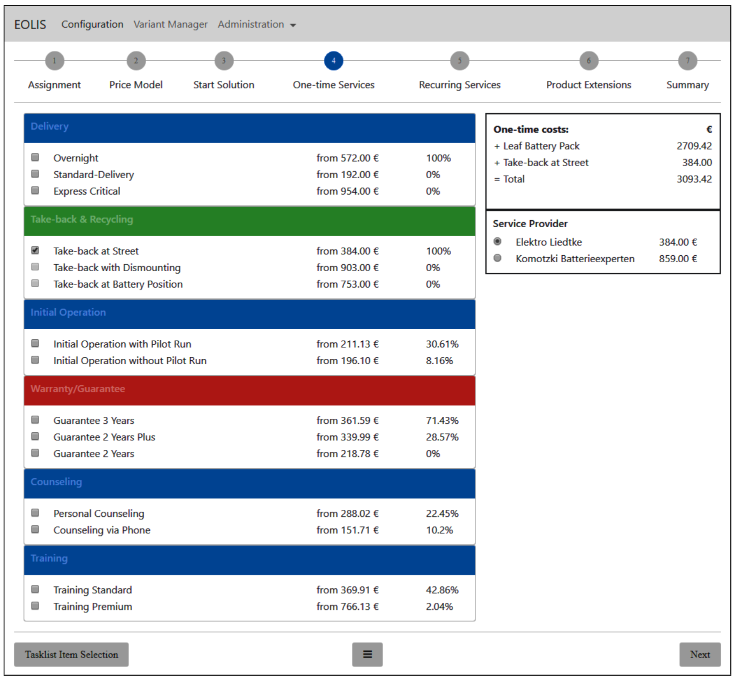
DSR requires to evaluate the designed artifact (Peffers et al., 2007). The evaluation of the presented configurator requires an artificial evaluation (Venable et al., 2016) as the deployment of the prototype in a real-world setting cannot be realized because the market is only about to establish and potential users of the system are not yet available. A criteria-based evaluation (Venable et al., 2016) is selected as it is most appropriate in this context. An artifact can be evaluated regarding different criteria (Jadhav and Sonar, 2009). In the following, we evaluate the developed artifact regarding *completeness*, *usability*, and *utility* based on the presented requirements and user stories.

To evaluate the completeness of the configurator, it is examined whether all requirements are met. The knowledge base of the configuration application is initialized whenever the configuration process is started. Therefore, the set of variables and its domains cannot be changed during the configuration (CR1). After each selection of a domain, the solver recalculates all valid domains. This ensures the inference and completeness characteristics (CR3 and 4). Also, it is possible to deselect previously selected services or product extensions. As this results in a recalculation of the domains, the restoration requirement is met as well (CR5). Besides the configuration process, the configurator provides a configuration rule editor. The editor allows the creation of configuration rules that state how domains can be combined (CR2). Besides, a configuration rule can be of the type require or exclude (CBR2). Last, before a newly created configuration rule can be saved, it is validated (CBR1). In addition to the generic requirements from literature, further requirements result from the user stories. During the configuration process, the costs of each component (domain) are clearly visible (US1). Moreover, at the end of the configuration process, the salesperson can use the profit calculator to check the offer based on the configured PSS (US2). Furthermore, the variant manager allows the user to apply different price models to a configured PSS (US3). The price model editor allows salespersons to add new price models to the system (US4). The created price models can be validated in the editor (US5). Furthermore, using the editor EVB types and scenario categories can be assigned to the models (US6). Created price models are automatically categorized according to their revenue stream type (US8). Also, the configuration rule editor allows the knowledge engineer to create configuration rules that have multiple triggers and/or targets (US7 and 9).

To evaluate the usability of the configurator, it is checked whether the UDRs are implemented. UDR1 requires an overview of the selected services and product extensions at the end of the configuration process. As the configuration process ends with an overview of the actual costs and the offer including the prices for the customer, this requirement is met. Furthermore, the implementation of the UDR2-6 are shown in the screenshot (Figure 2). A progress bar is displayed at the top of the configuration process (UDR2). Moreover, the configuration process is divided into the three parts: one-time services, recurring services, and product extensions (UDR3). The minimum costs of each domain are clearly visible, and when a component is selected, the exact costs for each service provider are displayed (UDR4). Furthermore, all selected components, as well as their costs and the total costs, are visible at the right top (UDR5 and 6). On our test machine and for our demonstration use cases, the response time of selecting a service/product is about 30-50 ms. During this time the server redetermines valid domains and sends the answer back to the client. According to UDR7, this should take less than 250 ms to "ensure the interactive feeling of working in real-time" (Hadzic and Andersen, 2004, p. 2). UDR8 requires that the user should not be forced to select the domains in a predefined order. Since it is possible to switch through the three

[1] https://angular.io/

groups (one-time services, recurring services, and product extensions) without selecting all mandatory services or product extensions, the requirement is met. Nevertheless, it is only possible to display the summary of the configuration process when the configuration is valid. The last UDR requires some start solutions (UDR9). These can be identified through the integration of the recommender system that enables automatic configuration by selecting components that are most likely to be bought by the particular customer. After all, the configurator should be empirically evaluated in real-world scenarios to gain more insights of the usability, which hitherto was not possible.



Delivery			
▢	Overnight	from 572.00 €	100%
▢	Standard-Delivery	from 192.00 €	0%
▢	Express Critical	from 954.00 €	0%
Take-back & Recycling			
✓	Take-back at Street	from 384.00 €	100%
▢	Take-back with Dismounting	from 903.00 €	0%
▢	Take-back at Battery Position	from 753.00 €	0%
Initial Operation			
▢	Initial Operation with Pilot Run	from 211.13 €	30.61%
▢	Initial Operation without Pilot Run	from 196.10 €	8.16%
Warranty/Guarantee			
▢	Guarantee 3 Years	from 361.59 €	71.43%
▢	Guarantee 2 Years Plus	from 339.99 €	28.57%
▢	Guarantee 2 Years	from 218.78 €	0%
Counseling			
▢	Personal Counseling	from 288.02 €	22.45%
▢	Counseling via Phone	from 151.71 €	10.2%
Training			
▢	Training Standard	from 369.91 €	42.86%
▢	Training Premium	from 766.13 €	2.04%

One-time costs:	**€**
+ Leaf Battery Pack	2709.42
+ Take-back at Street	384.00
= Total	3093.42

Service Provider		
●	Elektro Liedtke	384.00 €
○	Komotzki Batterieexperten	859.00 €

Figure 2.   Screenshot of the configurator prototype

6 Discussion

Through the developed and instantiated artifact–the configurator–we showed how the configuration of PSSs in the domain of repurposing used EVBs could be supported by adequate information systems. Since the configuration task is complex because of a high number of possible alternatives and constraints, IS support, as provided by the presented configurator, is essential for the repurposing of used EVBs. Therefore, our DSR project contributes to descriptive and prescriptive knowledge (Gregor and Hevner, 2013) in this domain as well as the construction of ISs for configuration problems. First, we extended an existing configurator to be able to account for different types of PSSs. For this the concept of price models that allow the conceptual modeling of different revenue streams was introduced. Such models are especially useful for the combination of different PSS types or revenue streams. Furthermore, we conceptualized and implemented the configuration problem as constraint satisfaction problem. This reduces maintenance effort and provides better scalability than the previous rule-based approach. Additionally, it shows how established concepts from the artificial intelligence literature can be used for the configuration of PSSs.

Despite that the configurator was developed specifically for the domain of repurposing used EVBs the findings can be generalized to a certain extend. Basically, the configurator can be used in different domains with similar characteristics by adapting certain parts of the presented components. Main characteristics of the domain are the complex and individual used product and the individual requirements of the scenarios they are intended to be used in. Furthermore, because of the used nature of the product, it has certain disadvantages in comparison to similar new products. In general, underlying data model can easily be adapted for other domains in that a fit between characteristics of a product and requirements of an application scenario have to be matched. The same is true for the knowledge base formulated as constrained satisfaction problem. Furthermore, the recommender system would have been modified and different domain-specific key figures would have to be displayed.

7 Conclusion and Outlook

In the presented DSR project we examined how to design a PSS configurator for used EVBs that integrates different types of PSSs to create offers that are tailored to the individual customer's needs. Furthermore, the configurator was protoypically implemented. A peculiarity of the business is the importance of extending the product with additional services to reduce information asymmetries and gain advantages over competitors by providing additional customer-value. Additionally, the individuality of each EVB and the range of possible second-life applications requires an individual configuration of each PSS. Furthermore, on the backdrop of mass-customization this individualization can Moreover, due to unknown circumstances of the emerging market, the integration of different price models into the configurator application is required. To realize a straightforward process from the creation of a price model to its integration in the configurator application, a consistent depiction of price models is needed. Hence, (cost-based) price models have been introduced to account for different pricing schemes. Those price models can be created using the price model editor component of the presented configurator and be used to compare different pricing alternatives. The design and conceptualization of the configurator were based on requirements derived from literature and user stories. Moreover, the constraint satisfaction problem approach was selected as configurator paradigm due to its generality and flexibility. In total, four core functionalities that the configurator needs to support were derived. The first functionality is a configuration rule editor that allows the adding of constraints to the control strategy. The second one is a price model editor that allows entering price models into the system. The configuration process that guides the configuration of PSSs, based on the added configuration rules and the selected price model, is the third functionality. The last one is a variant manager that allows, on the one hand, to apply different price models to a configured PSS and, on the other hand, to send offers of configured PSSs to the customers. Eventually, a criteria-based evaluation regarding the criteria completeness, usability, and utility was performed.

The main contributions lie in the design of a PSS configurator that is able to cope with different PSS types. For this, pricing models were introduced. Furthermore, the formulation as constraint satisfaction problem enables efficient configuration and flexibility towards changing requirements in an emerging domain. These changes have influence on the presented decision process configurator design.

Limitations are caused by the emerging nature of the market for trading used EVBs. Hence, many circumstances are still unknown yet, and assumptions regarding the market setting have been made. For example, it is assumed that external service providers are responsible for providing value-added services and product extensions. A detailed description of the services especially regarding value proposition, processes, and customer integration (co-creation) is neglected. Furthermore, since the configurator is already instantiated, ex-post evaluation methods like case studies or workshops should be performed to extend the criteria-based evaluation. This includes, among others, long-time tests of the PSSs to allow valid estimates of their failure rate and to be able to cover this risk in the pricing. Also, PSSs can be configured with different product extensions. Hence, it is necessary to evaluate if product extensions can be reused after the PSS's end of life and how this can be addressed in the pricing. Additionally, the price models are included on a very basic level and need to be extended in future research. Because the configurator's price model editor enables the integration of designed price models, it allows the creation of offers based on different price margins and pricing schemes. That can be used to evaluate different offers and reveal economically reasonable and accepted price models.

References

Ahmadi, L., M. Fowler, S. B. Young, R. A. Fraser, B. Gaffney, and S. B. Walker (2014). “Energy efficiency of Li-ion battery packs re-used in stationary power applications.” *Sustainable Energy Technologies and Assessments* 8, 9–17.

Ahmadi, L., A. Yip, M. Fowler, S. B. Young, and R. A. Fraser (2014). “Environmental feasibility of re-use of electric vehicle batteries.” *Sustainable Energy Technologies and Assessments* 6, 64–74.

Bach, T. C., S. F. Schuster, E. Fleder, J. Müller, M. J. Brand, H. Lorrmann, A. Jossen, and G. Sextl (2016). “Nonlinear aging of cylindrical lithium-ion cells linked to heterogeneous compression.” *Journal of Energy Storage* 5, 212–223.

Baines, T. S., H. W. Lightfoot, S. Evans, A. Neely, R. Greenough, J. Peppard, R. Roy, E. Shehab, A. Braganza, A. Tiwari, J. R. Alcock, J. P. Angus, M. Bastl, A. Cousens, P. Irving, M. Johnson, J. Kingston, H. Lockett, V. Martinez, P. Michele, D. Tranfield, I. M. Walton, and H. Wilson (2007). “State-of-the-art in product-service systems.” *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 221 (10), 1543–1552.

Barták, R. (2003). *Constraint Propagation and Backtracking-based Search*. Tech. rep. Charles University, pp. 1–43.

Beverungen, D., S. Bräuer, F. Plenter, B. Klör, and M. Monhof (2017). “Ensembles of context and form for repurposing electric vehicle batteries: an exploratory study.” *Computer Science - Research and Development* 32 (1-2), 195–209.

Beverungen, D., B. Klör, S. Bräuer, and M. Monhof (2015). “Will They Die Another Day? A Decision Support Perspective on Reusing Electric Vehicle Batteries.” In: *Proceedings of the Twenty-Third European Conference on Information Systems (ECIS 2015)*.

Blecker, T., N. Abdelkafi, G. Kreutler, and G. Friedrich (2004). “Product Configuration Systems: State of the Art, Conceptualization and Extensions.” In: *Eight Maghrebian Conference on Software Engineering and Artifical Intelligence*. Ed. by A. B. Hamadou, F. Gargouri, and M. Jmaiel. Tunis, Tunisia: Génie logiciel & Intelligence artificielle, pp. 25–37.

Bräuer, S. (2016). “They Not Only Live Once – Towards Product-Service Systems for Repurposed Electric Vehicle Batteries.” In: *Proceedings of the Multikonferenz Wirtschaftsinformatik (MKWI) 2016*. Ed. by V. Nissen, D. Stelzer, S. Straßburger, and D. Fischer. Ilmenau, Germany.

Bräuer, S., M. Monhof, B. Klör, F. Plenter, C. Siemen, and D. Beverungen (2016). “Residential Energy Storage from Repurposed Electric Vehicle Batteries - Market Overview and Development of a Service-Centered Business Model.” In: *Proceedings of the 18th IEEE Conference on Business Informatics (CBI)*. Paris, France.

Casals, L. C., B. A. García, F. Aguesse, and A. Iturrondobeitia (2017). “Second life of electric vehicle batteries: relation between materials degradation and environmental impact.” *The International Journal of Life Cycle Assessment* 22 (1), 82–93.

Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Redwood City, CA, USA: Addison-Wesley.

— (2005). *Agile estimating and planning*. Upper Saddle River, NJ, USA: Prentice Hall.

Colace, F., M. De Santo, and P. Napoletano (2009). “Product Configurator: An Ontological Approach.” In: *Proceedings of the Ninth International Conference on Intelligent Systems Design and Applications*. IEEE, pp. 908–912.

Conti, M., R. Kotter, and G. Putrus (2015). “Energy Efficiency in Electric and Plug-in Hybrid Electric Vehicles and Its Impact on Total Cost of Ownership.” In: *Electric Vehicle Business Models*. Ed. by D. Beeton and G. Meyer. Lecture Notes in Mobility. Berlin, Germany: Springer International Publishing, pp. 147–165.

Cready, E., J. Lippert, J. Pihl, I. Weinstock, P. Symons, and R. G. Jungst (2003). *Final Report - Technical and Economic Feasibility of Applying Used EV Batteries in Stationary Applications A Study for the DOE Energy Storage Systems Program*. Tech. rep. Sandia National Laboratories.

Felfernig, A., L. Hotz, C. Bagley, and J. Tiihonen (2014). *Knowledge-based Configuration: From Research to Business Cases*. Waltham, MA, USA: Morgan Kaufmann, p. 376. ISBN: 9780124158177.

Fischhaber, S., A. Regett, S. F. Schuster, and H. Hesse (2016). *Studie: Second-Life-Konzepte für Lithium-Ionen-Batterien aus Elektrofahrzeugen*. Tech. rep., p. 156.

Gohla-Neudecker, B., M. Bowler, and S. Mohr (2015). "Battery 2nd life: Leveraging the sustainability potential of EVs and renewable energy grid integration." In: pp. 311–318. ISBN: 9781479987047. DOI: 10.1109/ICCEP.2015.7177641.

Gregor, S. and A. R. Hevner (2013). "Positioning and Presenting Design Science Research for Maximum Impact." *MIS Quarterly* 37 (2), 337–355.

Guilding, C., C. Drury, and M. Tayles (2005). "An empirical investigation of the importance of cost-plus pricing." *Managerial Auditing Journal* 20 (2), 125–137.

Hadzic, T. and H. R. Andersen (2004). *An introduction to solving interactive configuration problems*. Tech. rep. August. Copenhagen: IT University of Copenhagen.

Hagman, J., S. Ritzén, J. J. Stier, and Y. Susilo (2016). "Total cost of ownership and its potential implications for battery electric vehicle diffusion." *Research in Transportation Business & Management* 18, 1–7.

Haug, A. (2007). "Representation of Industrial Knowledge - as a Basis for Developing and Maintaining Product Configurators." PhD Thesis. Technical University of Denmark, p. 268.

Hawkins, T. R., B. Singh, G. Majeau-Bettez, and A. H. Strømman (2013). "Comparative Environmental Life Cycle Assessment of Conventional and Electric Vehicles." *Journal of Industrial Ecology* 17 (1), 53–64.

Heinrich, M. and E. Jüngst (1991). "A resource-based paradigm for the configuring of technical systems from modular components." In: pp. 257–264.

Hevner, A. R. (2007). "A Three Cycle View of Design Science Research." *Scandinavian Journal of Information Systems* 19 (2), 87–92.

Hevner, A. R., S. T. March, J. Park, and S. Ram (2004). "Design Science in Information Systems Research." *MIS Quarterly* 28 (1), 75–105.

Heymans, C., S. B. Walker, S. B. Young, and M. Fowler (2014). "Economic analysis of second use electric vehicle batteries for residential energy storage and load-levelling." *Energy Policy* 71, 22–30.

Hotz, L. and T. Krebs (2003). "Configuration - State of the Art and New Challenges." In: *Proceedings of the 17th Workshop Planen,Scheduling und Konfigurieren,Entwerfen (PuK) – KI 2003 Workshop*, pp. 145–157.

Jadhav, A. S. and R. M. Sonar (2009). "Evaluating and selecting software packages: A review." *Information and Software Technology* 51 (3), 555–563.

Klör, B., M. Monhof, D. Beverungen, and S. Bräuer (2017a). "Design and Evaluation of a Model-Driven Decision Support System for Repurposing Electric Vehicle Batteries." *European Journal of Information Systems (EJIS)*.

Klör, B. (2017). "Building a Model-Driven Decision Support System for Repurposing Electric Vehicle Batteries – Design and Evaluation." PhD Thesis. University of Münster.

Klör, B., D. Beverungen, S. Bräuer, F. Plenter, and M. Monhof (2015). "A Market for Trading Used Electric Vehicle Batteries – Theoretical Foundations and Information Systems." In: *Proceedings of the 23rd European Conference on Information Systems (ECIS 2015)*. Münster, Germany.

Klör, B., S. Bräuer, D. Beverungen, and M. Monhof (2015). "A Domain-Specific Modeling Language for Electric Vehicle Batteries." In: *Proceedings of the International Conference on Wirtschaftsinformatik 2015*. Osnabrück, Germany, pp. 1038–1054.

Klör, B., M. Monhof, D. Beverungen, and S. Bräuer (2017b). "Recommendation and Configuration of Value-Added Services for Repurposing Electric Vehicle Batteries: A Vertical Software Prototype." In: *Proceedings of the 19th IEEE Conference on Business Informatics (CBI)*. Thessaloniki, Greece.

Knowles, M. J. and A. Morris (2014). "Impact of Second Life Electric Vehicle Batteries on the Viability of Renewable Energy Sources." *British Journal of Applied Science & Technology* 4 (1), 152–167.

Leitner, G., A. Felfernig, P. Blazek, F. Reinfrank, and G. Ninaus (2014). "User Interfaces for Configuration Environments." In: *Knowledge-based Configuration: From Research to Business Cases*. Ed. by A. Felfernig, L. Hotz, C. Bagley, and J. Tiihonen. Waltham, MA, USA: Morgan Kaufmann, pp. 89–106.

March, S. T. and G. F. Smith (1995). "Design and natural science research on information technology." *Decision Support Systems* 15 (4), 251–266.

Mittal, S. and B. Falkenhainer (1990). "Dynamic constraint satisfaction problems." In: *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI'90)*. Boston, MA, USA, pp. 25–32.

Mittal, S. and F. Frayman (1989). "Towards a Generic Model of Configuraton Tasks." In: *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI)*. Detroit, MI, USA, pp. 1395–1401.

Monhof, M. (2018). "Concepts of Product-Service Configurators for Repurposing used Electric Vehicle Batteries." In: *Proceedings of the Multikonferenz Wirtschaftsinformatik (MKWI)*.

Monhof, M., D. Beverungen, B. Klör, and S. Bräuer (2015). "Extending Battery Management Systems for Making Informed Decisions on Battery Reuse." In: *New Horizons in Design Science: Broadening the Research Agenda*. Ed. by B. Donnellan, M. Helfert, J. Kenneally, D. VanderMeer, M. Rothenberger, and R. Winter. Vol. 9073. Lecture Notes in Computer Science. Dublin, Ireland: Springer International Publishing, pp. 447–454. DOI: 10.1007/978-3-319-18714-3_37.

Mont, O. (2002). "Clarifying the concept of product–service system." *Journal of Cleaner Production* 10 (3), 237–245.

Nunamaker Jr., J. F., M. Chen, and T. D. M. Purdin (1990). "Systems Development in Information Systems Research." *Journal of Management Information Systems* 7 (3), 89–106.

Österle, H. and B. Otto (2010). "Consortium Research." *Business & Information Systems Engineering* 2 (5), 283–293. ISSN: 1867-0202.

Peattie, K. and M. Charter (2003). *The Marketing Book*. 5th Edition. Burlington, MA, USA: Butterworth Heinemann, p. 875.

Peffers, K., T. Tuunanen, M. A. Rothenberger, and S. Chatterjee (2007). "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems* 24 (3), 45–77.

Pine, J. B. I., B. Victor, and A. C. Boynton (1993). "Making Mass Customization Work." *Harvard Business Review* 9, 108–119.

Sabin, D. and E. C. Freuder (1996). "Configuration as Composite Constraint Satisfaction." In: *Proceedings of the AI and Manufacturing Research Planning Workshop*, pp. 153–161.

Sabin, D. and R. Weigel (1998). "Product configuration frameworks - A survey." *IEEE Intelligent Systems and Their Applications* 13 (4), 42–49.

Shen, J., L. Wang, and Y. Sun (2012). "Configuration of product extension services in servitisation using an ontology-based approach." *International Journal of Production Research* 50 (22), 6469–6488.

Simon, H. A. (1977). *The new science of management decision*. Englewood Cliffs, NJ, USA: Prentice Hall.

Spence, M. (1973). "Job Market Signaling." *The Quarterly Journal of Economics* 87 (3), 355–374.

Stegmann, R., M. Koch, M. Lacher, T. Leckner, and V. Renneberg (2003). "Generating Personalized Recommendations in a Model-Based Product Configurator System." In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) - Workshop on Configuration*.

Stoppel, E. and S. Roth (2015). "Consequences of usage-based pricing in industrial markets." *Journal of Revenue and Pricing Management* 14 (3), 140–154.

Stumptner, M. (1997). "An overview of knowledge-based configuration." *AI Communications* 10 (2), 111–125.

Tiihonen, J. and T. Soininen (1997). *Product Configurators-Information Systems Support for Configurable Products*. Tech. rep. Helsinki, Sweden: TAI Research Centre and Laboratory of Information Processing Science.

Trentin, A., E. Perin, and C. Forza (2011). "Overcoming the customization-responsiveness squeeze by using product configurators: Beyond anecdotal evidence." *Computers in Industry* 62 (3), 260–268.

— (2012). "Product configurator impact on product quality." *International Journal of Production Economics* 135 (2), 850–859.

Tsang, E. (1999). "A Glimpse of Constraint Satisfaction." *Artificial Intelligence Review* 13, 215–227.

Tukker, A. (2004). "Eight types of product-service system: Eight ways to sustainability? Experiences from suspronet." *Business Strategy and the Environment* 13 (4), 246–260.

Venable, J., J. Pries-Heje, and R. Baskerville (2016). "FEDS: a Framework for Evaluation in Design Science Research." *European Journal of Information Systems* 25 (1), 7–89.

Wang, L., W. K. Ng, and B. Song (2009). "Constraint Satisfaction Approach on Product Configuration with Cost Estimation." In: *Next-Generation Applied Intelligence*. Ed. by B.-C. Chien, T.-P. Hong, S.-M. Chen, and M. Ali. Vol. 5579. Lecture Notes in Computer Science. Berlin, Germany: Springer Berlin Heidelberg, pp. 731–740.