Otto, Kevin; Hölttä-Otto, Katja; Simpson, Timothy W.; Krause, Dieter; Ripperda, Sebastian; Moon, Seung Ki

Global Views on Modular Design Research

Published in:
Journal of Mechanical Design

DOI:
10.1115/1.4033654

Published: 01/07/2016

Document Version
Early version, also known as pre-print

Published under the following license:
Unspecified

Please cite the original version:
Otto, K., Hölttä-Otto, K., Simpson, T. W., Krause, D., Ripperda, S., & Moon, S. K. (2016). Global Views on Modular Design Research: Linking Alternative Methods to Support Modular Product Family Concept Development. *Journal of Mechanical Design*, *138*(7), [071101]. https://doi.org/10.1115/1.4033654

- Submitted to *Journal of Mechanical Design* (MD-15-1628) -

PAPER TITLE: **Global Views on Modular Design Research: Linking Alternative Methods to Support Modular Product Family Concept Development**

AUTHORS:
Kevin Otto*
Department of Mechanical Engineering
Aalto University, Finland
Kevin.otto@Aalto.fi
Fellow of the ASME

Katja Hölttä-Otto
Department of Mechanical Engineering
Aalto University, Finland
Katja.Holtta-Otto@Aalto.fi

Timothy W. Simpson
Department of Mechanical Engineering
The Pennsylvania State University,
University Park, PA 16802
tws8@psu.edu
Fellow of the ASME

Dieter Krause
Institute of Product Development and Mechanical Engineering,
Hamburg University of Technology, Germany
krause@tuhh.de

Sebastian Ripperda
Institute of Product Development and Mechanical Engineering,
Hamburg University of Technology, Germany
sebastian.ripperda@tuhh.de

Seung Ki Moon
School of Mechanical and Aerospace Engineering,
Nanyang Technological University
Singapore 639798
skmoon@ntu.edu.sg

(*CORRESPONDING AUTHOR)

ABSTRACT

Modular product platforms have been shown to provide substantial cost and time savings while still allowing companies to offer a variety of products. As a result, a multitude of product platform methods have been developed over the last decade within the design research community. However, comparison and integration of suitable methods is difficult since the methods have, for the most part, been developed in isolation from one another. In reviewing the literature in modularity and product platforms, we create a generic set of thirteen platform design steps for developing a platform concept. We then examine a set of product platform concept development processes used at several different companies, and from this form a generic sequence of the steps. We then associate the various developed methods to the sequence, thereby enabling the chaining together of the various modular and platform design methods developed by the community.

Keywords: Design methods, Modular and platform design, Platform architecture, Product development process, Product family design

1 INTRODUCTION

Single standalone products are rare. Most products are part of a family of products, a set of product variants that are based on a common platform. Product platforms enable the development of product families and generations of products, using shared assets to enable cost-effectiveness. These shared assets bring cost savings and many operational benefits in parts sourcing, manufacturing, and quality control, for example.

Over the years there has been active work in developing methods to design product platforms: methods to map requirements of a family to a set of products; methods to define the common and unique platform modules; methods to optimize variety, cost, commonality, or other parameters; methods to evaluate platforms; etc. [1, 2]. The transfer of these methods, algorithms, and techniques to industrial practice is now inhibited by the seemingly broad array of material without a coherent organizing structure to compare development process tasks and the associated available methods and tools. In the design research community, each method has typically been developed independently of other methods, and it is not clear if and how various methods could be used jointly. Therefore, a review of the research literature and alignment in design processes is needed. We review the research in the literature to illustrate how the different areas of focus can be linked into one of many design flows for platform concept development. The goal is to show how the various alternative approaches and methods to platforming can contribute to the overall goal of developing a successful product family.

The focus in this paper is on the early phases of design, such as system concepts, component selection, and component geometric layout. Detailed design steps such as product and component testing and validation are outside the scope of this paper. We also do not discuss production system or supply chain development. There remains in the research of modularity and platform that should be completed and explored in these domains.

The remainder of this paper is thus organized as follows. In the next section, we present our approach to defining alternative sequences and chains of the platforming methods available from the research community. In Section 3, we describe the thirteen steps using the various methods to implement the proposed design steps along with a case study

involving a family of unmanned ground vehicles. And then, we provide a guideline to utilize the matrix linking alternatives by selecting several particular flows in Session 4. In Section 5, we discuss the limitations of the proposed matrix and corresponding methods and address challenges for developing platforming strategies and practices.

2 APPROACH AND MODULAR PLATFORM DEFINITION PROCESS

To enable linking of the methods for platform concept development and enable a selection of methods for industrial application, three activities were completed. First, we reviewed and compared several product development processes from large industrial companies to establish an effective sequencing of platform development activities. Next the individual design methods were analyzed, grouped, and mapped into these activities. Finally, the methods themselves were re-associated to each step, now sequenced. This thereby defined a set of alternative methods to each step, and a permutative set of design flows to possibly use in a modular platform design process.

In studying the literature, we grouped methods into thirteen generic steps that are accomplished with the various methods or tools for platform concept development. This is shown in Table 1, listed in alphabetical order. We noticed, however, disagreements on which tasks are done before others. For example, some optimization methods can be set up to search for optimal module boundaries in some problem formulations, or one can predefine modular architectures as modularity permutations and then establish optimization formulations for each permutation. Similarly, one can complete voice-of-the-customer analysis for each product variant, or perhaps one can complete voice-of-the-customer analysis across several product variants. In general, there is not a unique best overall answer to these questions, and it can be highly problem dependent. We therefore do not argue for a single answer, but instead we seek to understand and provide starting points as recommendations.

(Put Table 1 Here)

For example, one can compare sequences of modularity and platforming development processes found in industry against one another to search for a potential common order of development steps. While it is unclear that industrial firms have any more authority in determining a correct sequence of steps, their use in practice does suggest a practical and effective nature. Furthermore, industrial design processes are not generated spuriously; many person-years of thought and effort have gone into their definition.

Therefore, to make a nominal recommendation, we collected a list of industrial design processes (see Table 1) from a sample of ten firms that regularly develop modular product platforms. Industries were selected to provide a variety of applications from automotive, industrial, electronics, and software. These included ABB [3-5], Airbus [6], Carrier [7-9], Cummins [10, 11], Danfoss [12-14], Ford [15, 16], IBM [17, 18], ITT [19, 20], LG [21], and Motorola [22].

Our data for analysis centered on standard-work technical review requirements – the information each design project must present at different points in the development process. We did not directly study standard-work descriptions of process steps, since we found these to be variable in level of definition across the companies, and the degree to which they are followed internal to the companies varies. Specifically, we did not look for the modular or

platform steps listed in Table 1 directly in standard work documentation. Some descriptions were insufficiently detailed, and others far more.

Rather, we focused on descriptions related to each technical review and compared this with the steps listed in Table 1. Given this set of data from several firms, the gate review in each corporate development process for each step in Table 1 was determined and recorded based on our analysis. The results are shown in Table 2, where the steps of Table 1 are now ordered in such a way to eliminate ordering conflicts of gate review checkpoints across these industrial firms.

(Put Table 2 Here)

When interpreting Table 2, each entry vertically would be monotonically increasing down the column, for the column's process to be consistent with the defined sequence. When several rows all have the same number, this indicates that this company does not particularly state which order these tasks are completed for review, as long as they are completed before moving on to the next set of tasks. Therefore one can see there is typically freedom allowed within these product development processes, for particular project leads to determine which tasks are appropriately done before or in parallel with other tasks. All that is required is the resultant documentation of results for review. All tasks so allowed have the same gate review number per column in Table 2.

The resulting sequence in Table 2 represents a nominal design flow. It is difficult to assign step results to corporate standard work descriptions due to a variety of concerns. First, standard work evolves, and what each corporation is doing at the present time is an evolved state from the descriptions available in the literature. Second, there are differences between when the results must be available for reviews and when they must be completed. Therefore, the sequence in Table 2 is our interpretation of the best available information. Finally, the data of Table 2 are that documented in the literature, rather than from observational studies of projects within each company. However, having worked with many of these firms, our experiences are generally consistent with what is described in the literature. Table 2 thus provides evidence of an effective sequence of the modular platform design steps found in the research literature.

Against Table 2, one can see the proposed generic sequence of modularity methods does fit a useful sequence to all the companies. There are no vertical sequences out of order. Given this result, we propose the sequence as a useful starting point for persons studying the design research literature on modularity tools and method. We also offer it is as useful starting point for practical purposes in placing the different methods and tools into useful design flows.

The next section discusses past research in the context of the platform development process shown in Table 2. This is a prescription for a logical sequence of the steps to apply in order, with alternatives to each step as indicated in Figure 1 and Table 3. We also list recommended modularity deliverables for review from each step, generalized and independent of the specific methods. This is not the only sequence one can take through these steps or the only methods that can be applied in a specific step; in general, one can find design problems where alternative orders are appropriate. However, Table 3 depicts a logical and useful starting point for any firm or stakeholder new to the

platform-based product development. Each step consists of alternative methods; one basic method, such as a manual process, as well as one or more alternative, more advanced, approaches.

Several of the mid-sequence steps are dependent on one another, and there is generally a choice on whether to take a functional modeling appraoch or not. This is shown in Table 3 using the expanded 'function' and 'component' columns in the 'alternatives' columns. This will be discussed in detail in the Section 3.5.

We also list "skip" as an alternative, indicating that one available option to consider is to simply skip that particular process step, with an associated loss of detail. As an example, one could choose to skip the architecture roadmap after defining module boundaries and move directly to commonality assignment, but skipping this step will thereby not ensure that the platform is flexible to potential future uncertain changes to the market and technology.

(Put Figure 1 Here)

(Put Table 3 Here)

In general there is always a trade-off between time allocated and used for each step and the expected benefits. Sometimes a choice is made to use a simpler approach with lower expected benefits to expedite the process. The baseline flow depicted in Table 3 represents a minimal set of steps necessary to generate a modular platform concept. It provides a simplified approach that starts with a set of existing products and makes them share common modules. This is the manually listing out of market segments and the product offers for each segment as an attack plan. Against this baseline, all additional methods are discussed in the next section for the added benefits provided over this minimal approach. An illustrative example is also provided in the next section to tie the various methods and approaches together.

3 ALTERNATIVE METHODS AND APPROACHES

3.1 Step 1: Market Segment Definition

The first step in the sequence is to define the market segments. Modular product platforms exist to more easily enable multiple products for multiple markets. Thus, the platform development effort ought to entail defining the population, or populations, of customers and product applications. Typically this starts with an observation of a perceived need in an application domain and extension into exploring the range of different geographies and demographics that may have similar needs. Market segmentation methods are well established [23] and will not be discussed here. The methods help in identifying potential clusters of customers with similar needs. This, in turn, enables designing a product variant on a per-market-segment basis, rather than a product that is trying to meet all the vastly different needs.

There are many approaches to clustering the market base into these segments. One can, for example, define a wide range of characteristics upon which to subdivide a population of potential customers, such as applications, geographic boundaries, behaviors, and demographics [24]. However, one will also find that many such distinctions are artificial and that there is no real difference in customer needs and there is a risk of over-partitioning the population. For each smallest partition of the population one can conduct surveys of customer demographics, use applications, or

even customer needs analysis to establish clear distinctions amongst the population. Clustering these results into internally homogeneous and externally heterogeneous groups helps form clear *market segments*.

The simplest approach to identifying these clusters of similar customers is to manually cluster on characteristics such as use or business application as market segments. For example, defense-related applications form a different segment than commercial applications, and geographic/regional variations may lead to distinct segmentation. Another option could be use of standard clustering methods such as hierarchical clustering. Within the design research community, clustering methods have been developed to establish common needs among refined segment characteristics including socio-economic attributes [25, 26]. Mathematical optimization models can also be used to determine demands for uncertain markets and estimate the revenue and profit of the defined market segments [27]. For example, Zhang et al. [26] used a fuzzy clustering approach to determine market segments for a product family. One issue that can arise is over-partitioning of the market, i.e., offering too many product variants. Unfortunately this is not always obvious until sales figures become apparent and do not match the sizes of the original clustering. Establishing differences in markets and targeted products remains a fruitful area of study.

We use a family of *unmanned ground vehicles* (UGVs) for explosive ordnance disposal as an example to discuss the various methods in this paper (see Figure 2). In this example, the market segments were manually defined against perceived business categories with lead users already exploring UGVs on their own accord. UGVs are being explored in both civilian and military applications. Smaller UGVs are often used to detonate an explosive remotely. These UGVs may be destroyed along with the ordnance. Larger UGVs, on the other hand, may perform additional functions such as sensing, defusing, or disposing the ordnance, allowing for repeated use and multiple missions.

(Put Figure 2 Here)

UGVs are used primarily to reduce casualty risk of ordnance disposal regardless of whether it is civilian or military; however, they can be used for many other types of functions as well. Partitioning these potential applications can lead to a form of market segmentation. As an example, we segment the UGV "market" into the following applications:

1. Explosive abatement – ordnance detection, disabling, and disposal
2. Hazardous sites – hazard sensing and locating in unhealthy environments
3. Combat – providing attack capability for high-risk missions
4. Reconnaissance – providing site observations and intelligence
5. Target and decoy – providing a target that simulates an enemy vehicle
6. Civil and commercial – UGVs for commercial transport

Meanwhile, the range and autonomy of the UGVs can provide a second axis to categorize potential customer segments:

1. Micro, line of sight
2. Small, remote operated, 2 mile range
3. Tactical, remote operated, 50 mile range

4. Long endurance, autonomous, 100 mile range

3.2 Step 2: Market Attack Plan

Following the product market matrix approach advocated by Kotler and Keller [28] and Meyer and Lehnerd [29], we can create a matrix of these two categories (application versus range) as shown in Figure 3. In their approach the rows of the matrices correspond to market segments and the columns to products segments, which helps designers identify how platform elements may be leveraged across multiple segments. The segments may share specific customer needs with multiple other segments, but this can be determined later during customer needs identification.

This matrix helps in creating a *market attack plan* that determines whether new products are offered simultaneously to all segments in parallel or rolled out sequentially over time to different market segments, Step 2 in the sequence of Figure 1. While the baseline process of market planning involves assessment of many factors include the sales potential, competitive offers, marketing and distribution, we consider here the mutual impact of modularity and the leveraging potential it offers. In uncertain market environments, scenario based Monte Carlo simulation and models could be applied to validate the market planning [30].

Notice that vertical leveraging (solid vertical line in Figure 3) can be associated with *scalable platforms* where different sizes of the same modules and components are used, while horizontal leveraging (dashed horizontal lines in Figure 3) can generally be associated with *swappable modules* where a common subset of core modules are used across different products. Successful platforms often utilize a combination of scaling and modularity to attack the market in a strategic and cost-effective manner [27]. Lei and Moon [31] develop a decision support system to identify new market segments and product positioning for product family design based on market data and product properties.

(Put Figure 3 Here)

To develop the market attack plan for our UGV product family example using the *product market matrix*, we consider the following performance characteristics: weight, speed, range, and lift capacity. Over 50 different potential applications were identified in the matrix based on type of ordnance, functionality (e.g., dig, detonate, diffuse), location of operation, etc. [32]). We then manually clustered the UGV market segmentation matrix into three segments that are, for the sake of this example, consistent with current systems in the market. We identified three "performance tiers" based on weight: (1) small, (2) medium, and (3) large. Mapping this into the matrix, we segmented the columns into small, medium and large UGVs, and the rows into explosion abatement, hazardous sites, and reconnaissance.

Based on this segmentation, a vertical leveraging strategy may be possible as illustrated by the solid vertical arrow in Figure 3. This would enable a new family of UGVs to be developed and scaled up for this initial segment, with platform modules created to satisfy requirements for the three sizes of UGVs. The dashed lines in Figure 3 illustrate the potential to later extend the platform by leveraging these modules horizontally to other market segments.

3.3 Step 3: Customer Needs Gathering

Once isolated as individual market segments, another step is to establish the customer needs for the each targeted product variant, Step 3 of Figure 1. This is a segment by segment step. Therefore, customer needs may be gathered for each individual product separately. In other words, *the intended variety in the product family should be designed prior to defining what the common modules should be in the platform*.

Hauser and Griffin [33] provide seminal work on gathering voice of the customer. Here, interviews are conducted with randomly sampled people from each market segment, and questioned about how they use the product. This elicits qualitative and quantitative need statements which they seek from the product. Zhang et al. [26] use conjoint analysis and discrete choice models to evaluate customer perceived utilities. Yu et al. [34] provide a method to group common modules based on common needs across segments.

The customer needs can be categorized into groups according to the similarity of attributes or the extent of customer satisfaction. For example, Kano [35, 36] introduced a conceptual model to analyze customer preferences. In the Kano's model, customer requirements are classified into three different types of needs: basic, performance, and attractive needs.

The defined customer-needs should be managed and updated continuously to develop a new product for the next generation. Schuh and Tanner [37], Harlou [38], and Eilmus et al. [39] utilize customer-needs variety diagrams to manage customer-needs variety. The link between the variants of a product family and the customer needs can be visualized using the tool *Tree of External Variety* as shown in Figure 4 [40].

(Put Figure 4 Here)

For the UGV example, the customer requirements were gathered from multiple Requirements Working Groups that met over a period of about two years. Each group was comprised of representatives from different branches of the military and included senior personnel, ordnance disposal experts, and technicians involved with logistics, maintenance, and support.

3.4 Step 4: Systems Requirements Definition

Once the customer needs for each market segment and corresponding product variant are clearly defined, the next step is to define quantitative verifiable system requirements for each product variant (see Figure 1). Again, there are well-established methods for this step, including the *House of Quality* [41]. Another option is to define the worst case operating conditions and define the system specifications for those cases. Perhaps the simplest option for this step is to convert the customer needs into system requirements similar to target value setting as described by Ulrich and Eppinger [42] or following requirement definition using INCOSE guidelines [43]. Regardless of what method is used, at the end of this step quantified system requirements have been identified which provide design targets to ensure each variant is capable of satisfying the customer needs in its corresponding market segment. Quality Function Deployment (QFD) can be combined with mathematical models to determine optimal system requirements and resources through sensitivity analysis [44].

In the early design stages, it is hard to define and propagate the system requirements accurately because customers' preference may vary based on specific functional requirements, design, manufacturing process, and new technology. To minimize design and requirement variations during product development, change propagation analysis (CPA) can be employed to identify product variants and predict the changing effects of the requirement. Clarkson et al. [45] introduced a change prediction method to determine the risky area and understand complex dependencies between components based on design changes. Change propagation behaviors [46] and a change propagation index [30] were proposed to quantify the impact of changing the requirements and behaviors.

For the UGV example, requirements were defined (e.g., weight, range, speed, manipulator length) for each segment in the market segmentation grid in Figure 3. Threshold and objective values were identified for each requirement. The threshold value is the minimum value that must be met in order to satisfy a requirement while the objective value provides a target that users would like to achieve. System requirements that were defined for the UGV example following the customer needs analysis. Threshold and objective values for each requirement were defined for each weight class. Depending on the applications, UGVs are needed with different system requirements for unique targets found in commercial or combat environments. For example, the cameras of military grade systems provide high resolution and zooming function to detect specific objectives and obstacles over a long distance. The systems and specifications of the unique targets can be determined by the customers and considered as the unique requirements of a UGV family. Additional examples of system requirements can be found in Donaldson [47].

This information can be used to help identify common, variant, and unique requirements, i.e., those that are the same (identical) among all three UGVs, those that are similar but vary slightly from one UGV to the next and those that are unique to a specific UGV. For example, some of the maneuvering, sensing, and communication requirements are the same for each UGV; however, the range and payload requirements vary for each weight class. Finally, the manipulator, reach, drag/roll/push, and large object requirements are unique to each UGV, with no requirements defined for the small UGV as that capability (e.g., large object pickup) is not needed on the small system. Again, specific examples can be found in Donaldson [47].

3.5 Step 5: Functional Requirements Definition

Requirements are often categorized into *functional requirements* and *constraints* [48, 49]. A functional requirement typically exhibits a certain functional behavior, i.e., to do something. A constraint, on the other hand, is a requirement that cannot be achieved by a single function. Weight and size are typical examples of constraints. Otto and Wood [49] provide a tutorial on defining functions versus constraints. We follow their suggested process, which starts with customer needs that are separated into functional requirements. We then continue to list the customer activities of the desired system and build a functional model using those activities and functional requirements. We then check that functional model against the original customer needs. This process results in helping designers understand *what*, as opposed to *how*, the system should do to achieve the customer needs.

Once the system requirements are set, per Figure 1 one can take one of two approaches to develop a platform architecture: a function-based approach or a component-based approach. A sequential approach going from function-based methods in the early conceptual phases to more refined component-based methods is also possible, but we

describe here a more common alternate approach. The function-based approach takes a more general view to define common functions independent of any particular embodiment decisions. On the contrary, the component-based approach makes use of *a priori* knowledge, however acquired or assumed, of the most relevant components needed. If the function-based approach is used, then that approach is followed by mapping function to form, i.e., defining the components as part of embodiment design. Thus in essence, both approaches lead to the same result though the function-based approach perhaps considers a broader set of alternatives. In this paper we briefly demonstrate both approaches.

Function-Based Approach

Other functional requirements-related research includes a functional requirement cluster analysis. Jiao et al. [50] developed an approach based on a fuzzy clustering and an associate rule mining for mapping customer needs onto functional requirements. Although the functional requirements matched with customer needs are defined, excessive functional variety for customer satisfaction may result in a dramatic increase of costs [50]. To manage and update the functional requirements' variety for the next generation in a product family, Schuh et al. [51] developed variety diagrams such as a hierarchical function structure can be also utilized.

A UGV's typical mission consists of moving to a predetermined location, observing and recording the surroundings and collecting objects or samples, or acting on objects as instructed and communicated from a remote location. The supporting functions for these basic activities are moving the camera and manipulators up, down, and horizontally, among other things. Since this is an existing family of robots, our choice of functions was influenced by the existing configuration of the robot family. This will result in a similarity between the embodiment of this functional model (see Figure 5) and the design created using the component-based approach next.

(Put Figure 5 Here)

Component-Based Approach

Components can be defined directly from the requirements without going through the function-based approach first. Alternatively the components can be defined based on the functions as part of the system embodiment phase [49, 52]. Component alternatives research includes Martin and Ishii's [53] *Design for Variety* method; they extend the House of Quality to create a function-component matrix to calculate the *Generational Variety Index* (GVI) for each subsystem. These methods have incorporated the mapping of requirements to components. The GVI matrix helps map engineering requirements into components in order to identify how much effort is needed to redesign a component if there was a change in a requirement. Subsystems with a high GVI value will undergo significant redesign in order to satisfy the range of customer needs while low GVI values indicate components that will remain relatively stable across the family. Figure 6 shows the GVI values for the UGV example [32]. As seen, the manipulator and chassis will vary substantially in the family (i.e., high GVI values) while cameras and OCU (Operator Control Unit) will have little variation. The GVI is a composite systematic. Krause et al. [40] developed a multi-criteria approach using the

Variety Allocation Model (VAM) that can be used to redesign the components as shown in Figure 7. The tool aims at increasing number of standard parts by linking the differentiating customer needs with the variant components.

(Put Figure 6 Here)
(Put Figure 7 Here)

In addition to these two methods, one could use *Modular Function Deployment* (MFD) [54] to create a similar *Product Property Matrix* and identify platform modules. Luo, et al. [55] also offer a QFD-based approach for platform optimization.

Yet another possible component-based approach is to use a DSM, or *Design Structure Matrix* [56-58] to help identify modules. DSM is most appropriate when redesigning existing products. Then a DSM is formed by decomposing those products and mapping their components in those the matrix. A component-based DSM for the UGV family is shown in Figure 8. For the UGV example, the DSM is constructed using a teardown disassembly approach to system decomposition [59]. In this case, four UGVs were disassembled; their subsystems were identified and recorded to create the component-based DSM.

After defining components, introducing methods for managing component variety is known to be an important step for the next generation in a product family [37]. Schuh and Tanner [37], Schuh et al. [51], and Harlou [38] employed component variety diagrams for the variety management.

(Put Figure 8 Here)

3.6 Step 6: Generic System Platform Architecture Definition

A modular platform forms the base for a set of product variants, possibly variants over multiple product generations. It is thus important that a modular architecture should encompass all the variant architectures that it must support, the next step in Figure 1. A simple way to create such a "generic" architecture for a modular platform is to create an architecture for each variant and merge these results together. The variant architectures can be created using either the functional or component-based approach described above.

A modular platform architecture is typically not generated as a single event transforming all product variants. Even when a modularity plan is developed, products themselves may be designed and developed in a sequence, one variant released at a time into the market. The legacy variants remain while the new platform variants are released. In this environment of constrained development resources, one approach is to consider the legacy products in the platform architecture. That is, to consider modular architectures that only modify portions of the entire architecture. Such approaches are entirely amenable to the methods outlined here. Typically several alternatives are considered, from radically new architectures to simple derivative architectures of existing systems. To capture this spectrum of possible redesigns, alternative schematic architectures can be used to keep track of the alternative platform concepts.

The functional modeling approach is sometimes preferred over matrix-based approaches due to its visually intuitive nature and higher abstraction level. In the functional approach [49, 52], the recommendation is to build each

individual functional model and then merge these into a generic platform architectural model. In the UGV example, the functional model in Figure 5 provides a generic functional model as the individual UGVs differ in performance levels, not functionality, in the vertically leveraged scalable platform. While each model may have a different sized payload bay, for example, all of the UGVs have these components to perform common tasks of reaching/grasping objects and observing their surroundings, i.e., the functions are the same.

Harlou [38] provides a system block diagram using the concept of organ from the theory of domains [60], and Bruun et al. [61] extended the Harlou's work to visualize a product system including a set of component variants, interfaces, and modules in a product family.

Matrix-based methods are another option, using the system components as might be described with a system block diagram. The GVI and MFD methods discussed previously can both be used to map functions to components and thereby define modules. The collection of components considered defines the generic platform architecture. Notice that a DSM also represents a generic architecture. Similar to the functional models, the DSM in Figure 8 is already a generic DSM since the UGV family differs in size, etc. of the components but not in generic component types.

Other generic system architecture research includes methods by Blanchard and Fabrycky [62] and heuristic methods by Maier and Rechtin [63]. Crawley et al. [64] make note of system level interactions and architecture. Brière-Côté et al. [65] propose a structure-based approach to the management of product variety in product families.

3.7 Step 7: Component Alternatives

Following the definition of the generic system platform architecture, the variety of components within the product family must be specified. The sizes for the family as well as the functions and its attributes have to be defined. Alternatives can be developed manually or using tools such as the function component matrix [53], variety allocation model (VAM) [40], component variety management [37, 38, 51], or modular function deployment (MFD) [54]. The VAM (see Figure 7) supports the development of alternatives by separating the components in standard and variant shares. This results in platform concepts with more components and a higher level of commonality within the product family or fewer components and a lower level of commonality.

In the UGV example, the main circuit board variant depends on all application-relevant requirements and resulting functions and principles (see Figure 7). An alternative concept could be to strive for more components and split the circuit board into a standard component for the standard functions and additional modular circuit boards to provide optional functions. The standard circuit board is built in every UGV variant, and additional boards can be added depending on the specific application for a given variant. An alternative concept could be to reduce components and design a circuit board that fulfills all functions, but this would likely result in costly unused functionality and limited upgradeability.

3.8 Step 8: Module Boundary Definition

A critical step of either the functional or component-based approach is to partition the architecture into modules of various sizes or capabilities. There are many reviews of modularity methods [66, 67]. Sample methods include various modularity heuristics such as those for module identification from a functional model [68, 69], and heuristic identification of commonalities using a modularity matrix [70]. Ericsson and Erixon [54] developed MFD to identify modules according to strategic reasons to isolate components into modules. Krause et al. combine these product strategic reasons with a functional component-based approach for modularization using the *Module Process Chart* (MPC) [40] (Figure 9). It supports the module definition by showing the preferred modules of each life phase. Further, various clustering algorithms that operate on a DSM have been created with the goal of module definition in mind [71, 72]. Most clustering algorithms include a measure to decide when to conclude the clustering, i.e., when the desired degree of modularity is achieved. Thebeau [73] developed a measure to minimize connections outside modules. This was improved by Borjesson and Hölttä-Otto [74]. Yu, et al. [72] aim to minimize the information needed to describe the connectivity between modules. Hölttä-Otto and de Weck [75], on the other hand, use singular value decomposition to evaluate module independence and Borjesson and Hölttä-Otto introduce means to cluster simultaneously based on module independence and strategic similarity drivers [76] Reviews of both coupling and similarity modularity can be found elsewhere [66, 77, 78]. In addition to the use of metrics, one can also use hierarchical and supervising clustering to define modules [79]. This can be done either based on the functional requirements, component specifications [80], or in conjunction with another method that defines a metric, such as MFD. Moon and McAdams [81] proposed a strategic decision-making method for module sharing in platform design using a game theoretic approach. Mathematical models can be developed to evaluate the interface complexity of the defined modules [70] and utilized to identify the number of the modules and the values of design variables in the modules [82, 83].

(Put Figure 9 Here)

For the module boundary definition step, we continue both the functional and component-based approaches with the UGV example. First, for the functional approach, we apply the module heuristics [68] to identify potential modules. As seen in Figure 5, we identify multiple potential modules using the branching flow heuristic (blue dashed lines) as well as a transmission type module and a dominant flow module (both in red). The functional model is at a relatively high abstraction level, which results in modules that may include only one abstracted function. We note that if the system were to be decomposed further, then we would identify additional module candidates – such as all the separate drives (within the "Allow DoF" functions) as conversion-transmission modules.

For the component based approach we chose to cluster the DSM. While there are many clustering algorithms available [84, 85], we choose to use manual clustering for the UGV example because it is relatively straightforward. Figure 8(b) shows the clustered DSM for the original UGV architecture in Figure 6. As can be seen in the figure, the partitioning suggests two bus modules (for the chassis and electronics), two 3x3 modules (for communications and observation), and six single component modules (for specific functionality, e.g., manipulator). Similar to the function-

based approach, if the system was decomposed further, a larger DSM with larger modules would be created to identify modules within modules. For the UGV example, comparing Figure 5 to Figure 8(b) we see that essentially the same result is obtained. Both methods resulted in the mast with the camera head, for example, to be isolated as a module (functions allow orientation DOF, allow location DOF and sense environment in Figure 5). Similarly, both methods left the chassis (support loads function) and electronics (control UGV function in Figure 5) as buses. The functional approach is more intuitive and visual; DSMs can be programmed and automated.

3.9 Step 9: Architecture Roadmap and Future Uncertainty Management

After defining a platform architecture and its modularity, one may need to consider its evolution over time as technology changes. Architecture roadmap research includes *technology roadmapping* [86, 87] to plan successive module level upgrades and associated product upgrades. In general, a *technology roadmap* is a plan to develop products in the future using technology that is not yet fully developed. Despite the incompleteness, the past trends allow for future prediction of expected performance. A common technology forecast uses Moore's law [88, 89] or similar performance improvement curves in, for example, telecommunication [90], printers [42], and power transmission [91].

Technology roadmaps should be created and updated with product family roadmaps to reflect future technology trends. This is because a *product family roadmap* [92-94] outlines the evolution of the product family, platform, and derivatives based on the technology forecast. Lee and Park [94] introduced different kinds of technology and product family roadmaps and their application.

In product platform design, such roadmaps are often done at the module level, where each module is scrutinized for future evolution and upgrades strategically planned over time. While individual modules change over time, the system architecture and interfaces of those modules remains fixed into the future, until the modules change so radically that the interfaces must change and an entirely new platform is required [30]. The module update cycle is typically faster than the platform redesign cycle; in some cases, the platform may survive 15-20 years while the modules refresh annually. Schuh et al. [51, 95] introduced a product development approach based on module and product roadmaps. To develop modules and products under the roadmaps, a platform-oriented approach for a long-range creation of commonality is preferred to a product-oriented approach in their research.

Other methods such as MFD [54] incorporate the potential for technology evolution during module boundary definition. For example, in MFD a modularity rule states that if a component or sub-assembly is expected to evolve over time, it should be separated into a module.

The variability of products affects customers' preference by increasing flexibility in choosing a product in competitive market environments [96]. In uncertain market environments, the valuation of a product increases flexibility in decision-making for developing new products or redesigning existing products [97]. Design can be been adapted to changing environments, such as customers' preferences, technologies, economic situations, company's strategies, competitive moves, and government regulations. Strategic adaptability and flexibility are essential in capitalizing on future investment opportunities and responding properly to market trends in the uncertain environments [98]. Market-based design approaches provide the capability to investigate additional flexibility and strategic value

during design. For example, Jiao et al. [99] applied real options to product family design and developed a valuation framework to evaluate the options of configuration inherent in design. Meanwhile, Gamba and Fusari [100] proposed a stochastic dynamic framework for valuing the contribution of modularization process and modular operations in the design of systems using real options.

For the UGV example, we consider the generic product architecture and the current modules sizes, and we can create a plan for evolving (or not evolving) the modules over time based on this. For example, we can expect mobility technology (i.e., Provide Propulsion module, Support Weight module) to remain relatively constant in the near future, and thus there is no plan to evolve these modules on a roadmap. The battery technology (i.e., Store Power module) and computing platform and embedded controls software (e.g., Controls module), on the other hand, are more likely to evolve over time. Thus we would need to redesign or upgrade each UGV as the technology evolves. For example, battery energy per unit cost or per unit weight continually improves, and one can project increased stored energy expectations every two years. Comparing this against stored energy levels needed to achieve increased power loads of improving other modules (e.g., larger powertrain motors, increased imaging cameras, longer range communication, etc.) defines the point in time where it is useful to upgrade the battery module to higher power. Technology roadmapping can help prevent the platform for hindering evolution of new technology infusion.

3. 10 Step 10: Commonality Assignment

As we discussed earlier, the UGV generic architecture and any individual product architecture is identical, since the modules differ in size but not function. In order to define how many different modules are needed to meet the customer requirements for the UGV, or all products in general, the next step in Figure 1 is a commonality analysis. The maximum number of instances for any module is the number of product variants if every instance needs to be unique to every product variant (e.g., the chassis may be unique to the small, medium, and large UGVs). Conversely, the minimum number of instances for any module is one, i.e., one module that is common on all of the variants (e.g., the same camera on every UGV). Reality usually lies in between those two extremes where modules instances may vary slightly across different subsets of products (e.g., the manipulator on the medium UGV may have two articulating segments while the large UGV may have three).

There has been extensive work in determining commonality. Thevenau and Simpson [101] provide a review of multiple commonality indices. GVI provides a useful approach for commonality assignment. A low GVI value indicates a low chance of redesign within the product family; therefore, a single module may suffice for this component. Meanwhile, a high GVI value indicates a significant chance for redesign. In this latter case, multiple modules will be needed to achieve the range of requirements for the family. For the UGV example, a subsequent analysis of each pair of UGVs (e.g., small and medium, medium and large, and large and small) was conducted to translate the GVI values in Figure 6 to parametric variation needed in each subsystem. Through this analysis we sought to identify potential opportunities for scaling, for example, the chassis in one or more dimensions based on the threshold and objective values for each UGV pair even though the chassis will vary across each weight class. Table 4 summarizes the final recommendations for commonality in key subsystems. Here a grey box indicates common settings across two or more UGVs, e.g., chassis height can be common to all three UGVS, but only the small and

medium have common chassis length and width based on the requirements. Performance models of the UGV could be combined with optimization algorithms to determine the best settings for each module and each parameter in the family, using these recommendations as a starting point.

(Put Table 4 Here)

3.11 Step 11: Architecture Module Sizing

Sizing of the module alternatives from the previous step is a well-researched area. It includes a host of approaches using numerical methods, clustering, trade-off analysis and optimization techniques to determine module sizes. Simpson, et al. [32], Fujita and Yoshisa [102], Khire et al. [103], Yu, et al. [34], Kumar, et al. [104], Dai and Scott [105], Liu. et al. [106], Hernandez, et al. [107], Chowdhury et al. [108, 109], and Moon, et al. [83] provide example formulations. Han and Papalambros [110] and Kokkolaras, et al. [111] discuss *target cascading*, an approach for sizing modules based on higher level system requirements. Given targets on each system response for each product variant, the shared modules can be sized to identify optimal settings using such numerical methods.

Additionally the functional interface has to be specified for each module variant. The functional module boundaries are compared against the range and sizes of the modules resulting in the performance requirements at the interfaces. These requirements are documented in the functional interface specifications; so, all concerned stakeholders know how the modules must match functionally. This can reduce the design effort and enable the rapid configuration of product variants.

This is repeated for each platform architecture alternative, thereby quantifying their performance (on all variants) and allowing a well determined platform down-selection. In some cases also multiple platforms might be needed to best benefit from platform commonality without sacrificing performance or other aspects. Research on platforming extent [112, 113] aims to answer how many platforms there should be within a given family or products.

3.12 Step 12: Architecture Concept Layout

Based on the developed modules and defined variants, the first three-dimensional platform concepts can be designed. In this step, the design of the size and type of the module interfaces is important. It has to be assured that each interface with its dimensions, forces, and media flows is suitable for the range of developed modules.

Krause et al. [114, 115] introduced the Module Interface Graph (MIG) (Figure 10) to visualize and analyze the variety of components and connecting flows (i.e., interfaces) in a product family. Particularly, the MIG also provides the information on the simplified spatial view of the real product. It supports the designing of the first three-dimensional platform concepts and interfaces. The modular product structure of the UGV product family is shown in Figure 10 based on the results of the clustered DSM in Figure 8b. For instance, an electrical, information and structural connection are needed for the interface between the main body and the front arm for tools. Between the main body and the battery, an electrical and a structural connection are necessary.

(Put Figure 10 Here)

3. 13 Step 13: Architecture Downselection

Assuming these works result in more than one optimal solution, trade-offs amongst the results must be considered. Khire, et al. [103] present Pareto frontier solution visualization methods. Gonzalez-Zugasti, et al. [116] extend Pugh's process to compare platforms that support multiple products, each evaluated separately. Saari and Sieberg [117] discuss pairwise comparison methods to improve decision-making with alternatives. Saari [118] discusses hierarchical decisions to improve optimal solutions by introducing a generalized inconsistency theorem that could also be extended for platform selection. Frey et al. [119] show that the concept selection process with multiple stakeholders is forced to be inconsistent with a set of assumptions that each appear reasonable at face value. While all of these concerns raised by these authors are true, this means one or more of the group decision-making assumptions must be violated. In industrial practice, this typically means a senior person (e.g., platform manager, program manager, project manager) assumes responsibility for the decision, and it no longer is a group decision.

For the UGV example, we can create equations of the requirements in terms of module sizing variables. For example, we can describe batteries with energy storage capacity, size, weight, etc. Using such variables, we can derive UGV system-level equations of the UGV top speed, overall mass, climbing angle, etc. Table 5 shows the module sizes for the platform, which UGV variant each is used in, and the overall performance of each UGV variant on several criteria. Note that Table 5 is different for each platform architecture, and a concept selection matrix can be used to rank these platform architectures.

(Put Table 5 Here)

4 DISCUSSION

Each of the various tasks of platform concept development has been discussed along with the various methods available, as outlined in Figure 1. Given this selection of methods for each of the tasks, it is apparent there is a wide array of permutations, each representing an alternative "design flow" through the matrix linking each set of alternatives. We offer thoughts on selection of several particular platform design flows.

The first design flow to discuss is the baseline, the straight flow-down in the first column of the matrix in Table 3. This approach is the most basic and fastest approach to achieving a platform, though it is also limited in assurance of correctness in terms of necessary future platform redesign iterations to fix any likely product or performance gaps on individual products. Nonetheless, the approach is typical for simple two or three variant product platforms, such as a small medical device product family where two or three distinct products serving distinct procedures and market segments are sought to be made from a common platform. The market segmentation and attack planning (Steps 1 and 2) are often obvious and can be done manually in this simple case. Requirement differences (Step 4) are considered, and these are mapped to a basic systems block diagram (Step 6). The variety and unique requirements help partition the diagram into shared and unique blocks (Step 8). With this, specific modules can be sized (Step 11), and then the

platform concept is laid out (Step 12). All other steps, such as the customer needs analysis and consideration of future products, while recommended, is not strictly necessary for a platform redesign. For cases where there is a small portfolio of non-changing products sought to be brought into one platform for cost reduction requirements, this approach may be enough.

A second design flow through the available methods is the Modular Function Deployment (MFD) approach used extensively by Erixon et al. [53], who have documented many industrial applications. For companies new to shared modular platform design, this approach may be the most useful starting point, offering modularity design guidance without significant tool complexity. While the approach incorporates all thirteen steps as deemed needed by any application, the basic approach makes use of five basic steps, each with a specific tool. The basic design flow consists of (1) mapping customer needs to requirements with a Quality Function Deployment (QFD) matrix, (2) generating a system block diagram of components, (3) establishing which components are the most critical and partitioning them using the Modular Function Deployment (MFD) matrix, (4) grouping the components into modules either manually or by using clustering algorithms on the MFD matrix, and (5) generating concept layouts using the suggested clustering.

Several other design flows are possible making use of the Design Structure Matrix (DSM) methods. The range of project complexity and tool chain automation can vary substantially with DSM methods. For example, as a simplified method, in some projects it might be valuable to do a fast rough estimate for potential modules, jumping straight to the final steps early. As we can see in Figure 1 and Table 3, one of the shorter paths to a platform from the baseline approach is to simply build a DSM reflecting a system block diagram directly from off-the-shelf components and link these to customer requirements using QFD, for example, similar to the MFD approach. One could then simply cluster the component DSM to form modules using the available surrogate cost minimization clustering algorithms, and then proceed with the module sizing and layout. This is again most likely suitable for a redesign of an existing family for more efficient variety and commonality ratio, where modules likely vary in type rather than size.

The choice of using a DSM clustering algorithm approach versus an MFD clustering approach is one of choice on available algorithms. The basic MFD approach uses manual clustering, Stake [120] applied multi-objective K-means clustering to align common specifications, and Stake and Blackenfeldt [121], Borjeson and Hölttä-Otto [76] and others combined the multi-objectives of MFD with the DSM clustering algorithms. The span from MFD to DSMs is well researched and marked by the degree of manual versus automated clustering desired.

A more fundamental overhaul of a product platform, or a “clean sheet” design of a new product family, would likely require other combinations of methods. For example, the use of function-based methods can help abstract the problem and can lead to better solutions than simple reconfiguration of components. Function-based methods are used in more complex, hierarchical systems such as building systems including complex HVAC systems and elevator systems [122], or even larger systems including ship building [123] and aircraft [124]. In such systems, the system functionality must be allocated across subsystems. For example, the electrical controls of an elevator might be a functional module that is shared across many sizes of elevators. This module consists of all wires and software connecting the pressed buttons and sensors to the drive motor ultimately. Yet, the button switches must be on each floor and in each cab, whereas the motor must be in the hoistway. This functional module has shared components, yet

it also is dispersed across different subsystems that must be grouped for location reasons. In such hierarchical modular platform designs, there must be both dispersed functional modules (e.g., electrical controls) as well as localized component modules (e.g., a cab module or floor button module). The key difference with systems requiring functional modularity is that the system functionality must be allocated to different component modules, and there are degrees of freedom in how this is accomplished. For example, it is perhaps functionally irrelevant where to locate the computational dispatching algorithms in an elevator system, whether on the local controller moving with cab, on a floor level button controller, on a controller next to the motor controller, or even distributed over all of these, or even none of the above and entirely separate. To consider these questions of complex hierarchical systems, functional block diagrams become necessary along with partitioning of the functionality.

Another lens to consider alternative design flows using the various methods is to consider the level of automation desired. This can span from purely manual methodologies to algorithmic search and trade-offs at each step. For example, Becz [124] presents an automated concept design exploration tool to search over several alternative functional modularity schemes with associated alternative components for the conceptual design of F35 aircraft. Trade-space comparisons of thousands of different modular design alternatives are possible, to establish and explore the Pareto frontier.

Finally, while vastly different in complexity of actions, notice the distinction between the functional and component approaches are overall not that different. Only 4 of 13 steps are necessarily different; much of the early work involving customer needs and specification requirements are similar. Also, the work needed to ascertain the pool of available components and design options to consider is the same. Similarly, the design work on concept layout is the same. Overall, we find the sequence of methods in Table 3 to be logically consistent and supported by what we have found and observed in practice.

5 SUMMARY AND CONCLUSIONS

In this work, we attempted to review the literature to link the different strands of platform research into logical sequences that can be practically used for product platform development. We do not assert that this order is the only one or even the best. We do claim, however, that it is a good starting point since it is well aligned with actual platform development processes at companies who regularly develop product platforms as shown in Table 2.

Questions often arise with firms on whether one should go through all the steps presented. We argue that it is not necessarily needed or warranted in all cases; there are dependencies. Static/mature industrial markets need less technology forecasting and higher focus on cost models, for example. The choice of steps to make use of is often problem-specific, and it also depends on the resources and information available.

Interesting future work includes comparisons and analysis of the different industrial processes across the columns in Table 2. For example the order, number, and content of the various gate reviews appears to perhaps have evolved over time, and future work could investigate this evolution with respect to a company's platforming strategy and practice.

REFERENCES

[1] Simpson, T.W., Z. Siddique, and J. Jiao, eds. *Product Platform and Product Family Design: Methods and Applications*. 2005, Springer: New York.

[2] Jiao, J., T.W. Simpson, and Z. Siddique, *Product Family Design and Platform-Based Product Development: A State-of-the-Art Review.* Journal of Intelligent Manufacturing, 2007. **18**(1): p. 5-29.

[3] Swanstrom, L., *Greening the Gate Model.* ABB Review, 2009. **2**: p. 23-24.

[4] Leup, P. and C. Rytoft, eds. *Special Report IEC61850*. 2010, ABB Review. 62 pages.

[5] Anderstig, S., D. Eklund, and M.T. , Mälardalen University, 2012, *Development of Production Concept*. 2012, Mälardalen University: Västerås, Sweden.

[6] Altfeld, H.-H., *Commercial aircraft projects: Managing the development of highly complex products*. 2010: Ashgate Publishing, Ltd.

[7] Vavoski, S., *A Global Sourcing Strategy for Durable Tooling*. 2002, MIT: Cambridge, MA.

[8] Hutton, T., *ACE versus Six Sigma*. 2004, MIT: Cambridge, MA.

[9] LeBlanc, A., *Integrating Value Methodologies into Product Development and Project Management Processes at Pratt & Whitney Canada.* Value World, 2006. **29**(2): p. 2-7.

[10] Gokal, R., *New Product Development: Examining the Art of the Possible Whilst Imagining the Future.* HTI Cummins Turbo Technology, 2010. **14**: p. 7-8.

[11] Hoffman, C. and R. Maher, *How Cognition Cockpit Improves Development Processes at Cummins, Inc.* 2010: Cognition Corporation.

[12] Hauksdottir, D., A. Vermehren, and J. Savolainen. *Requirements Reuse at Danfoss*. in *IEEE Requirements Engineering Conference*. 2012. IEEE.

[13] Kvist, M., *Product Family Assessment*. 2010, Danish Technical University: Copenhagen, Denmark.

[14] Matzen, D., *A Systematic Approach to Service Oriented Product Development*, in *Management*. 2009, Danish Technical University: Copenhagen, Denmark.

[15] Soderborg, N., *Design for Six Sigma at Ford.* Six Sigma Forum Magazine, 2004: p. 15-22.

[16] Šurinová, Y., *Ford's System for Cost Reduction due to Development Time*, in *Institute of Quality Engineering*. 2009, Slovak University of Technology.

[17] Bauer, R., et al., *The Silverlake Project: Transformation at IBM*. 1992, New York, NY: Oxford University Press.

[18] Tang, V. and R. Bauer, *Competitive Dominance: Beyond Strategic Advantage and Quality Management*. 1995, New York, NY: John Wiley & Sons.

[19] ITT Industries, *VBPD: Filling the Product Pipeline with Sure Winners*, in *In Our Hands Extra*. 2002, ITT Industries: http://www.itt.com/iohextra/rel13/text-article1.html.

[20] Reed, T., *Integrating Design for Assembly and Manufacturing Into ITT Industries' New Product Development Process*. 2nd Edition ed. 2004, Newport, RI: DFMA Forum.

[21] Kim, S., *SSPL Strategy in Electronics Industry*, in *Software & System Product Line Seminar* 2012: KOSTA, Seoul, South Korea.

[22] Tegel, D. and R. Kriva, *Motorola – Incorporating Six Sigma into New Product Development.* PDMA Visions Magazine, 2004. **28**(4): p. 14-16.

[23] Churchill, G. and D. Iacobucci, *Marketing Research: Methodological Foundations*. 9th Edition ed. 2004, Cincinnati, OH: South-Western College Pub.

[24] Cleveland, M., N. Papadopoulos, and M. Laroche, *Identity, demographics, and consumer behaviors.* International Marketing Review, 2011. **28**(3): p. 244-266.

[25] Moon, S.K., S.R.T. Kumara, and T.W. Simpson. *Data Mining and Fuzzy Clustering to Support Product Family Design*. in *ASME Design Engineering Technical Conferences - Design Automation Conference*. 2006. Philadelphia, PA: ASME.

[26] Zhang, Y., J. Jiao, and Y. Ma, *Market segmentation for product family positioning based on fuzzy clustering.* Journal of Engineering Design, 2007. **18**(3): p. 227-241.

[27] Kumar, D., W. Chen, and T.W. Simpson, *A market-driven approach to product family design.* International Journal of Production Research, 2009. **47**(1): p. 71-104.

[28] Kotler, P. and K. Keller, *Marketing Management*. 2009, Upper Saddle River, NJ: Prentice Hall.

[29] Meyer, M.H. and A.P. Lehnerd, *The Power of Product Platforms: Building Value and Cost Leadership*. 1997, New York: The Free Press.

[30] Suh, E., O. de Weck, and D. Chang, *Flexible product platforms: framework and case study.* Research in Engineering Design, 2007. **18**(2): p. 67-89.

[31] Lei, N. and S.K. Moon, *A Decision Support System for market-driven product positioning and design.* Decision Support Systems, 2015. **69**: p. 82-91.
[32] Simpson, T.W., et al., *From User Requirements to Commonality Specifications: An Integrated Approach to Product Family Design.* Research in Engineering Design, 2012. **23**(2): p. 141-153.
[33] Hauser, J. and A. Griffin, *The Voice of the Customer.* Marketing Science, 1993. **12**(1): p. 1-27.
[34] Yu, J.S., J.P. Gonzalez-Zugasti, and K.N. Otto, *Product Architecture Definition Based Upon Customer Demand.* ASME Journal of Mechanical Design, 1999. **121**(3): p. 329-335.
[35] Kano, N., et al., *Attractive quality and must-be quality.* The Journal of the Japanese Society for Quality Control, 1984. **14**(2): p. 39-48.
[36] Wang, T. and P. Ji, *Understanding customer needs through quantitative analysis of Kano's model.* International Journal of Quality & Reliability Management, 2010. **27**(2): p. 173-184.
[37] Schuh, G. and H. TANNER. *Mastering Variant Variety using the Variant Mode and Effects Analysis*. in *ASME Design Engineering Technology Conference, Paper No.:DETC98/DFM-5736*. 1998. Atlanta, Georgia.
[38] Harlou, U., *Developing product families based on architectures: Contribution to a theory of product families*, in *Department of Mechanical Engineering*. 2006, Technical University of Denmark.
[39] S., E., et al. in *The 12th International Design Conference, DESIGN2012*. 2012. Dubrovnik, Croatia: Design Societies.
[40] Krause, D., et al., *Integrated Development of Modular Product Families: A Methods Toolkit*, in *Advances in Product Family and Product Platform Design*, T.W. Simpson, et al., Editors. 2014, Springer New York. p. 245-269.
[41] Hauser, J.R. and D. Clausing, *The House of Quality.* Harvard Business Review, 1988. **66**(3): p. 63-73.
[42] Ulrich, K.T. and S.D. Eppinger, *Product Design and Development*. 3rd Edition ed. 2004, New York: McGraw-Hill/Irwin.
[43] INCOSE, *INCOSE Systems Engineering Handbook v3.2*. 2010, www.incose.org: International Council on Systems Engineering.
[44] Simpson, T.W., et al., *Advances in Product Family and Product Platform Design*. 2014: Springer-Verlag New York.
[45] Clarkson, P.J., C. Simons, and C. Eckert, *Predicting Change Propagation in Complex Design.* Journal of Mechanical Design, 2004. **126**(5): p. 788-797.
[46] Eckert, C., P.J. Clarkson, and W. Zanker, *Change and customisation in complex engineering domains.* Research in Engineering Design, 2004. **15**(1): p. 1-21.
[47] Donaldson, B., *Application of Product Family Design Tools to Unmanned Ground Vehicles*, in *Mechanical & Nuclear Engineering*. 2010, The Pennsylvaina State University: University Park, PA.
[48] Suh, N.P., *The Principles of Design*. 1990, New York, NY: Oxford University Press.
[49] Otto, K.N. and K.L. Wood, *Product Design: Techniques in Reverse Engineering and New Product Development*. 2001, Upper Saddle River, NJ: Prentice Hall.
[50] Jiao, J. and Y. Zhang, *Product portfolio identification based on association rule mining.* Computer-Aided Design, 2005. **37**(2): p. 149-172.
[51] Schuh, G., J. Arnoscht, and S. Rudolf. *Integrated development of modular product platforms*. in *Technology Management for Global Economic Growth (PICMET), 2010 Proceedings of PICMET '10,IEEE*. 2010.
[52] Pahl, G. and W. Beitz, *Engineering Design: A Systematic Approach*. 2nd Revised Edition ed, ed. K. Wallace. 1996, New York: Springer-Verlag.
[53] Martin, M.V. and K. Ishii, *Design for variety: Developing standardized and modularized product platform architectures.* Research in Engineering Design 2002. **13**(4): p. 213:235.
[54] Ericsson, A. and G. Erixon, *Controlling Design Variants: Modular Product Platforms*. 1999, New York: ASME.
[55] Luo, X., J. Tang, and C. Kwong, *A QFD-based Optimization Method for a Scalable Product Platform.* Engineering Optimization, 2010. **42**(2): p. 141-156.
[56] Steward, A.D., *The Design Structure System: A Method for Managing the Design of Complex Systems.* IEEE Transaction on Software Engineering, 1981. **28**(3): p. 71-74.
[57] Eppinger, S.D., et al., *A Model-Based Method for Organizing Tasks in Product Development.* Research in Engineering Design, 1994. **6**(1): p. 1-13.
[58] Browning, T.R., *Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions.* IEEE Transactions on Engineering Management, 2001. **48**(3): p. 292-306.
[59] Chiriac, N., et al. *Three Approaches to Complex System Decomposition*. in *13th International Dependency and Structure Modelling Conference*. 2011. Cambridge, MA.

[60] Andreasen, M.M., *Syntesemetoder på Systemgrundlag - Bidrag Til En Konstruktionsteori*, in *Department of Machine Design*. 1980, Lund Institute of Technology.
[61] Bruun, H.P.L., N.H. Mortensen, and U. Harlou, *Interface diagram: Design tool for supporting the development of modularity in complex product systems.* Concurrent Engineering, 2014. **22**(1): p. 62-76.
[62] Blanchard, B.S. and W.J. Fabrycky, *Systems Engineering and Analysis*. 5th Edition ed. 2010, Englewood Cliffs, NJ: Prentice Hall.
[63] Maier, M.W. and E. Rechtin, *The Art of Systems Architecting*. 2nd Edition ed. 2000, New York: CRC Press.
[64] Crawley, E., et al., *The Influence of Architecture in Engineering Systems*. 2004, Cambridge, MA: MIT.
[65] Brière-Côté, A., L. Rivest, and A. Desrochers, *Adaptive generic product structure modelling for design reuse in engineer-to-order products.* Computers in Industry, 2010. **61**(1): p. 53-65.
[66] Gershenson, J.K., G.J. Prasad, and Y. Zhang, *Product Modularity: Measures and Design Methods.* Journal of Engineering Design, 2003. **15**(1): p. 33-51.
[67] Fixson, S.K., *Modularity and Commonality Research: Past Developments and Future Opportunities.* Concurrent Engineering: Research and Applications, 2007. **15**(2): p. 85-111.
[68] Stone, R.B., K.L. Wood, and R.H. Crawford, *A Heuristic Method to Identify Modules from a Functional Description of a Product.* Design Studies, 2000. **21**(1): p. 5-31.
[69] Zamirowksi, E.J. and K.N. Otto. *Identifying Product Portfolio Architecture Modularity Using Function and Variety Heuristics*. in *ASME Design Engineering Technical Conferences - Design Theory and Methodology*. 1999. Las Vegas, NV: ASME.
[70] Dahmus, J.B., J.P. Gonzalez-Zugasti, and K.N. Otto, *Modular Product Architecture.* Design Studies, 2001. **22**(5): p. 409-424.
[71] Helmer, R., A. Yassine, and C. Meier, *Systematic Module and Interface Definition Using Component Design Structure Matrix.* Journal of Engineering Design, 2010. **21**(6): p. 647-675.
[72] Yu, T.-L., A.A. Yassine, and D.E. Goldberg, *An Information Theoretic Method for Developing Modular Architectures Using Genetic Algorithms.* Research in Engineering Design, 2007. **18**(2): p. 91-109.
[73] Thebeau, R., *Knowledge Management of system Interfaces and Interactions for Product Development Process*, in *System Design & Management Program*. 2001, MIT: Cambridge, MA.
[74] Borjesson, F. and K. Hölttä-Otto. *Improved Clustering Algorithm for Design Structure Matrix*. in *ASME International Design Engineering Technical Conferences*. 2012. Chicago, IL: ASME.
[75] Holtta-Otto, K. and O. de Weck, *Degree of Modularity in Engineering Systems and Products with Technical and Business Constraints.* Concurrent Engineering: Research and Applications, 2007. **15**(2): p. 113-126.
[76] Borjesson, F. and K. Hölttä-Otto, *A module generation algorithm for product architecture based on component interactions and strategic drivers.* Research in Engineering Design, 2014. **25**(1): p. 31-51.
[77] Hölttä-Otto, K., et al., *Comparative Analysis of Coupling Modularity Metrics.* Journal of Engineering Design, 2012. **23**(10-11): p. 790-806.
[78] Guo, F. and J.K. Gershenson. *A Comparison of Modular Product Design Methods based on Improvement and Iteration*. in *ASME Design Engineering Technical Conferences - Design Theory & Methodology*. 2004. Salt Lake City, UT: ASME.
[79] Moon, S., T. Simpson, and S.T. Kumara, *A methodology for knowledge discovery to support product family design.* Annals of Operations Research, 2010. **174**(1): p. 201-218.
[80] Hölttä-Otto, K., V. Tang, and K. Otto, *Analyzing Module Commonality for Platform Design Using Dendrograms.* Research in Engineering Design, 2008. **19**(2-3): p. 127-141.
[81] Moon, S.K. and D.A. McAdams, *A Market-Based Design Strategy for a Universal Product Family.* Journal of Mechanical Design, 2012. **134**(11): p. 111007.
[82] Simpson, T., et al., *From user requirements to commonality specifications: an integrated approach to product family design.* Research in Engineering Design, 2012. **23**(2): p. 141-153.
[83] Moon, S., K. Park, and T. Simpson, *Platform design variable identification for a product family using multi-objective particle swarm optimization.* Research in Engineering Design, 2014. **25**(2): p. 95-108.
[84] Eppinger, S.D. and T.R. Browning, *Design Structure Matrix Methods and Applications*. 2012, Cambridge, MA: MIT Press.
[85] Lei, N. and S.K. Moon. *Decision Support Systems Design for Data-Driven Management*. in *ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Paper No. DETC2014-34871*. 2014. Buffalo, New York: American Society of Mechanical Engineers.
[86] Albright, R., *How to Use Roadmapping for Global Platform Products.* PDMA Visions Magazine, 2002. **XXVI**(4): p. 19-22.

[87] Cosner, R., et al., *Integrating Roadmapping into Technical Planning.* Research Technology Management, 2007. **50**(6): p. 31-48.
[88] Allan, A., et al., *2001 Technology Roadmap for Semiconductors.* Computer, 2002. **35**(1): p. 42-53.
[89] Edenfeld, D., et al., *2003 Technology Roadmap for Semiconductors.* Computer, 2004. **37**(1): p. 47-56.
[90] Willyard, C.H. and C.W. McClees, *Motorola's Technology Roadmap Process.* Research Management, 1987. **30**(5): p. 13-19.
[91] Daima, T.U. and T. Oliverb, *Implementing Technology Roadmap Process in the Energy Services Sector: A Case Study of a Government Agency.* Technological Forecasting and Social Change, 2008. **75**(5): p. 687-720.
[92] Wheelwright, S.C. and W.E. Sasser Jr, *The new product development map.* Harvard Business Review, 1989. **67**(3): p. 112.
[93] Meyer, M.H. and A.P. Lehnerd, *The power of product platforms: building value and cost leadership*. New York, NY. Vol. 10020. 1997, New York, NY.
[94] Lee, S. and Y. Park, *Customization of technology roadmaps according to roadmapping purposes: Overall process and detailed modules.* Technological Forecasting and Social Change, 2005. **72**(5): p. 567-583.
[95] Schuh, G., M. Schiffer, and J. Arnoscht. *Scenario based development of robust product architectures*. in *Technology Management for Emerging Technologies (PICMET), 2012 Proceedings of PICMET '12, IEEE*. 2012.
[96] Chan, S.L. and W.H. Ip, *A dynamic decision support system to predict the value of customer for new product development.* Decision Support Systems, 2011. **52**(1): p. 178-188.
[97] Bollen, N.P.B., *Real Options and Product Life Cycles.* Management Science, 1999. **45**(5): p. 670-684.
[98] Smit, H.T.J. and L. Trigeorgis, *Strategic Investment: Real Options and Games*. 2004, Princeton, New Jersey: Princeton University Press.
[99] Jiao, J., C.M. Lim, and A. Kumar, *Real options identification and valuation for the financial analysis of product family design.* Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2006. **220**(6): p. 929-939.
[100] Gamba, A. and N. Fusari, *Valuing Modularity as a Real Option.* Management Science, 2009. **55**(11): p. 1877-1896.
[101] Thevenot, H.J. and T.W. Simpson, *Commonality Indices for Product Family Design: A Detailed Comparison.* Journal of Engineering Design, 2006. **17**(2): p. 99-119.
[102] Fujita, K. and H. Yoshida, *Product Variety Optimization Simultaneously Designing Module Combination and Module Attributes.* Concurrent Engineering, 2004. **12**(2): p. 105-118.
[103] Khire, R., et al. *Product Family Commonality Selection through Interactive Visualization*. in *ASME Design Engineering Technical Conferences - Design Automation Conference*. 2008. New York, NY: ASME.
[104] Kumar, D., W. Chen, and T.W. Simpson, *A Market-Driven Approach to Product Family Design.* International Journal of Production Research, 2009. **47**(1): p. 71-104.
[105] Dai, Z. and M.J. Scott, *Product Platform Design through Sensitivity Analysis and Cluster Analysis.* Journal of Intelligent Manufacturing, 2007. **18**(1): p. 97-113.
[106] Liu, Y., et al., *A Hierarchical Statistical Sensitivity Analysis Method for Multilevel Systems with Shared Variables.* ASME Journal of Mechanical Design, 2010. **132**(3): p. 031006 (11 pages).
[107] Hernandez, G., J.K. Allen, and F. Mistree, *Platform Design for Customizable Products as a Problem of Access in a Geometric Space.* Engineering Optimization, 2003. **35**(3): p. 229-254.
[108] Chowdhury, S., A. Messac, and R.A. Khire, *Comprehensive Product Platform Planning (CP3) Framework.* Journal of Mechanical Design, 2011. **133**(10): p. 101004-101004.
[109] Chowdhury, S., et al. *Comprehensive product platform planning (cp3) for a modular family of unmanned aerial vehicles*. in *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Paper No.:DETC2013-13181*. 2013. American Society of Mechanical Engineers.
[110] Han, J. and P. Papalambros, *A Sequential Linear Programming Coordination Algorithm for Analytical Target Cascading.* ASME Journal of Mechanical Design, 2010. **132**(2): p. 021033 (8 pages).
[111] Kokkolaras, M., et al., *Analytical Target Cascading in Product Family Design*, in *Product Platform and Product Family Design: Methods and Applications*, T.W. Simpson, Z. Siddique, and J. Jiao, Editors. 2005, Springer: New York. p. 225-240.
[112] Seepersad, C.C., G. Hernandez, and J.K. Allen. *A Quantitative Approach to Determining Product Platform Extent*. in *ASME Design Engineering Technical Conferences - Design Automation Conference*. 2000. Baltimore, MD: ASME.

[113] de Weck, O., *Determining Product Platform Extent*, in *Product Platform and Product Family Design: Methods and Applications*, T.W. Simpson, Z. Siddique, and J. Jiao, Editors. 2005, Springer: New York. p. 241-301.
[114] Blees, C. and D. Krause, *On the development of modular product structures: a differentiated approach*, in *10th International Design Conference - Design 2008* 2008: Dubrovnik, Croatia. p. pp. 301-308.
[115] Gebhardt, N., T. Bahns, and D. Krause, *An example of visually supported design of modular product families*, in *24rd CIRP Design Conference*. 2014: Milano, Italy.
[116] Gonzalez-Zugasti, J.P., K.N. Otto, and J.D. Baker, *Assessing Value for Platformed Product Family Design.* Research in Engineering Design, 2001. **13**(1): p. 30-41.
[117] Saari, D.G. and K.K. Sieberg, *Are Partwise Comparisons Reliable?* Research in Engineering Design, 2004. **15**(1): p. 62–71
[118] Saari, D.G., *Aggregation and Multilevel Design for Systems: Finding Guidelines.* Journal of Mechanical Design, 2010. **132**(8): p. 081006-1-9.
[119] Frey, D.D., et al., *The Pugh controlled convergence method: model-based evaluation and implications for design theory.* Research in Engineering Design, 2009. **20**(1): p. 41-45.
[120] Stake, R., *On conceptual development of modular products - Development of supporting tools for the modularisation process*. 2000, KTH: Stockholm, Sweden
[121] Stake, R. and M. Blackenfeldt, *Modularity in Use – Experiences from Five Companies*, in *4th WDK Workshop on Product Structuring*. 1998: Delft.
[122] Otto, K.N. and C. Jacobson. *Using Model Uncertainty to Reduce Verification and Validation in Noise and Vibration Problems*. in *ASME Design Engineering Technical Conferences*. 2012. Chicago, USA.
[123] Wellsandt, S. and D. Cerri, *Manutelligence: Product Service Design and Manufacturing Intelligence Engineering Program*. 2015, Document D2.1, www.manutelligence.eu.
[124] Becz, S., et al. *Design system for managing complexity in aerospace systems*. in *13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. 2010. Fort Worth, TX.

FIGURE CAPTIONS LIST

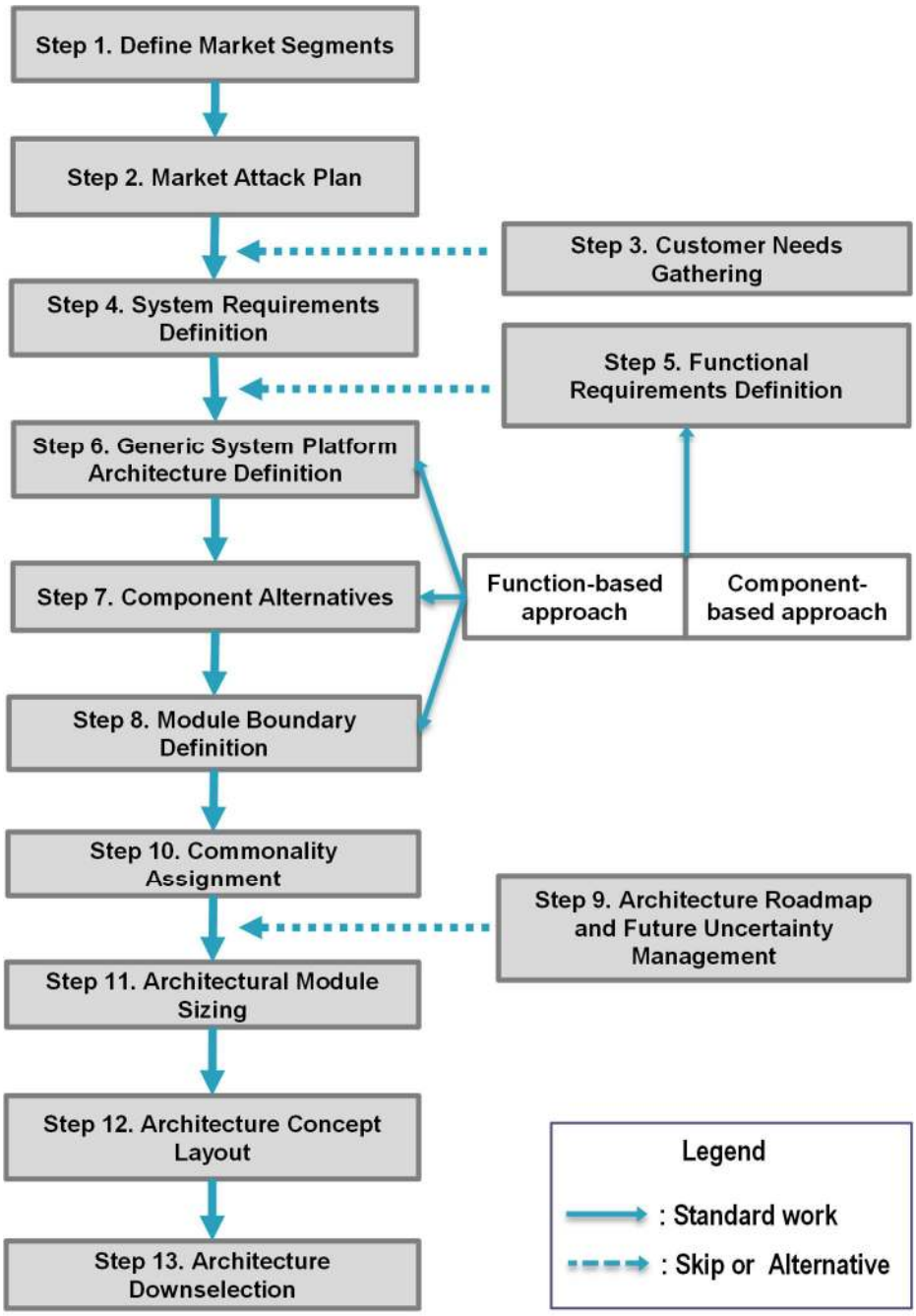

FIGURE 1: ARCHITECTING STEPS AND AVAILABLE METHODS.

(a) Bombot
http://www.defensetech.org

(b) Talon
http://www.foster-miller.com

(c) Packbot
http://www.irobot.com

FIGURE 2: EXAMPLES OF UNMANNED GROUND VEHICLES.

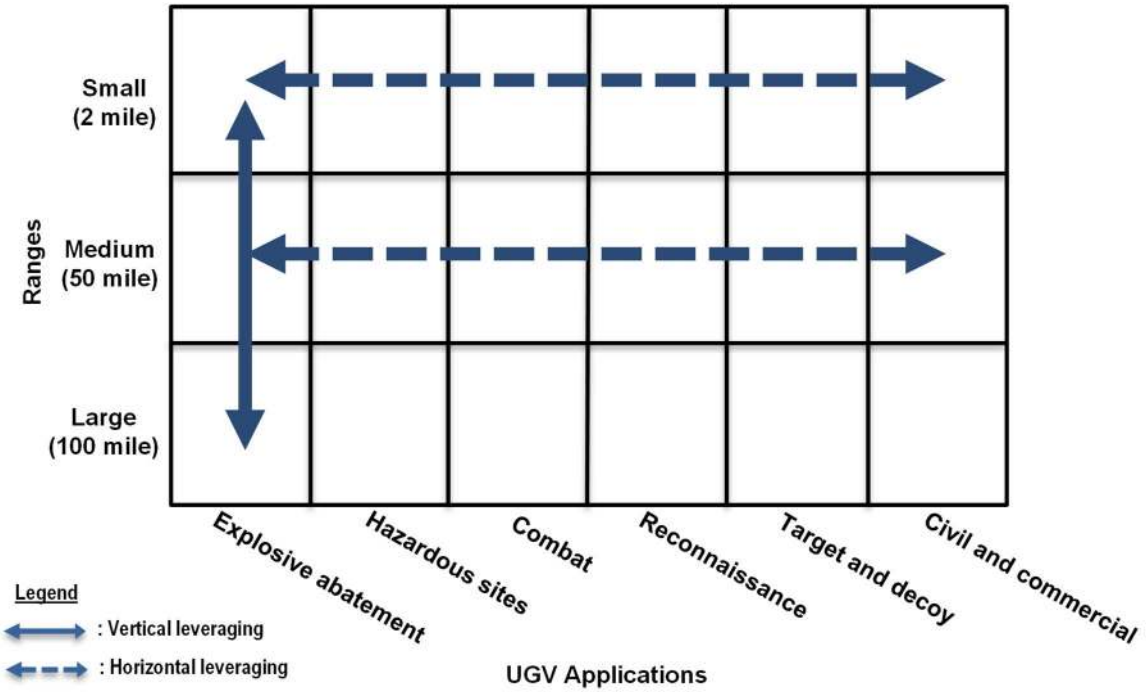

FIGURE 3: PLANNED MARKET STRATEGY FOR THE UGV EXAMPLE.

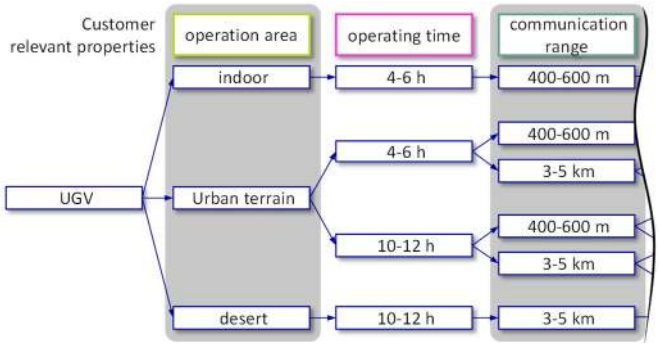

FIGURE 4: TREE OF EXTERNAL VARIETY FOR THE UGV EXAMPLE.

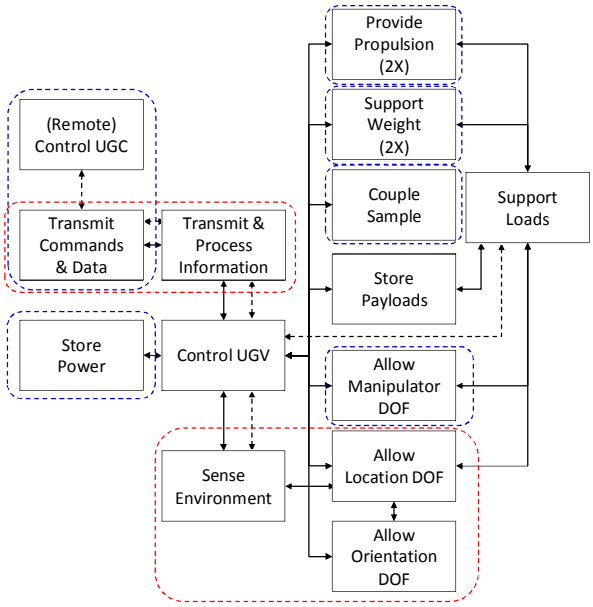

FIGURE 5: FUNCTIONAL MODEL OF A UGV

	Subsystem										
Requirement	Chassis	Battery	Tracks	Communication box	Electronics box	Manipulator	Gripper	Cameras	Payload bay	Antennae	OCU
Range (feet)				6						6	
Slope Climb (deg)	3		3			1					
Maneuver width (in)	3		3								3
On board vol (in^3)	6				3				6		
On board wt (lb)	6				3				6		
Drag/Roll/Push (lb)	6	3	6			6	6				
Horiz reach (in)	6	3				9	1				
Vert high reach (in)	1					9	1				
Sensing type									3		
Video vert high reach (in)	1					9	1				
Large Obj Pickup (length)						3	6				
Large Obj Pickup (width)						3	6				
Large Obj Pickup (height)						3	6				
Lift capac (lb)	6					9	3				
Tool precision				1			6	1			1
Tool size (in^3)						3	6				
Tool wt (lb)						3	6				
Comm range (ft)		3		6						6	
GVI Values	**38**	**9**	**12**	**13**	**6**	**58**	**48**	**1**	**15**	**12**	**4**

GVI Rating

Rating	Description
9	**Requires major redesign of the component (>50% of initial redesign costs)**
6	**Requires partial redesign of component (<50%)**
3	**Requires numerous simple changes (<30%)**
1	**Requires few minor changes (<15%)**
0	**No changes required**

FIGURE 6: GENERATIONAL VARIETY INDEX FOR UGV FAMILY [32].

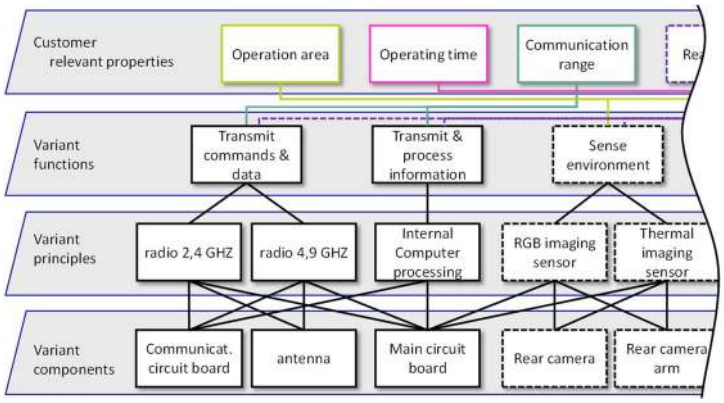

FIGURE 7: VARIETY ALLOCATION MODELL FOR UGV FAMILY.

		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Chassis			x	x			x	x				x		
2	Battery														
3	Flipper	x					x								
4	Main track	x					x								
5	Communication box						x							x	
6	Electronics box		x	x	x	x		x	x	x	x	x	x		
7	Manipulator	x					x								
8	Mast	x					x			x					
9	Head						x		x			x			
10	Gripper						x								
11	Cameras						x			x					
12	Payload Bay	x					x								
13	Antenna					x									x
14	OCU													x	

(a) Unclustered

		1	6	2	13	5	14	3	4	7	8	9	11	10	12
1	Chassis							x	x	x	x				x
6	Electronics box			x		x		x	x	x	x	x	x	x	x
2	Battery														
13	Antenna					x	x								
5	Communication box		x		x										
14	OCU				x										
3	Flipper	x	x												
4	Main track	x	x												
7	Manipulator	x	x												
8	Mast	x	x									x			
9	Head		x								x		x		
11	Cameras		x									x			
10	Gripper		x												
12	Payload Bay	x	x												

(b) Clustered

FIGURE 8: UNCLUSTERED AND CLUSTERED DSM FOR THE UGV FAMILY.

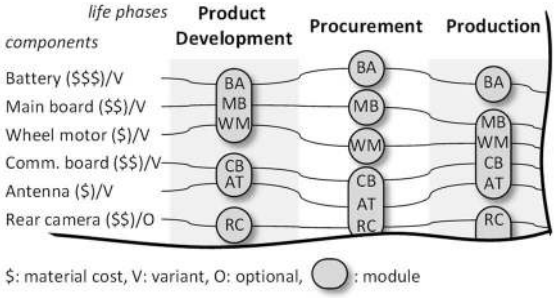

FIGURE 9: MODULE PROCESS CHART FOR UGV FAMILY.

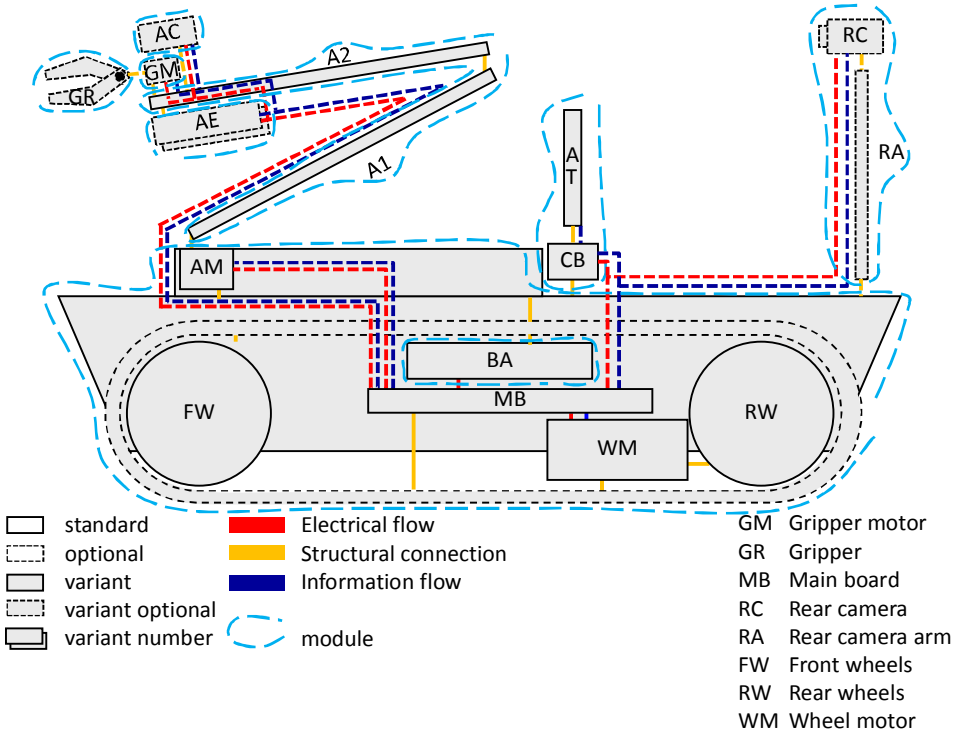

FIGURE 10: MODULE INTERFACE GRAPH FOR UGV FAMILY.

TABLE CAPTION LIST

TABLE 1: STEPS COMMON ACROSS RESEARCH*

Architecture Concept Layout	Architecture Downselection
Architecture Module Sizing	Architecture Roadmapping
Commonality Assignment	Component Alternatives
Define Market Segments	Functional Requirement Definition
Generic System Architecture	Market Attack Plan
Module Boundary Definition	System Requirement Definition
Understand Customer Needs	

*(Alphabetically ordered)

TABLE 2. INTERPRETED REVIEW WHERE STEP RESULT IS SCRUTINIZED.

Steps	ABB	Airbus	Carrier	Cummins	Danfoss	Ford	IBM	ITT	LG	Motorola
Define Market Segments	1	1	1	1	1	1	1	1	1	1
Market Attack Plan	2	1	1	1	1	1	1	1	1	2
Customer Needs Gathering	2	1	2	1	1	1	1	1	1	3
System Requirement Definition	3	2	2	2	1	2	2	1	2	3
Functional Requirement Definition	3	2	2	2	1	2	2	2	2	3
Component Alternatives	3	3	3	2	2	3	2	2	2	4
Generic System Architecture	3	3	3	2	2	3	2	2	2	5
Module Boundary Definition	3	4	3	2	2	3	2	2	2	5
Architecture Roadmap and Future Uncertainty Management	3	5	3	2	2	3	2	2	3	5
Commonality Assignment	3	5	3	2	2	3	2	2	3	6
Architectural Module Sizing	4	6	4	3	3	4	3	3	4	6
Architecture Concept Layout	4	6	4	3	3	4	3	3	4	6
Architecture Downselection	4	6	4	3	3	4	3	3	4	6

TABLE 3. ARCHITECTING STEPS AND AVAILABLE METHODS.

Steps	Baseline	Alternatives		Deliverables
Step 1. Define Market Segments	Manual Categories	Market Data Cluster Analysis, Kawakita Jiro (KJ) Qualitative Cluster Analysis, Customer Behavior Analysis		• Groups of potential customers (people, markets, organizations) that share common characteristics (geography, attributes, purchase propensities) • Potential niche market segments • Size of potential market segments and competitors • Uncertainty related key functional attributes and design variables
Step 2. Market Attack Plan	Manual Planning	Market Segment Grid, Product Positioning		• Decisions on whether/which segments to attack in what sequence • Impact of competition in different market segments • Unique or shared products per segment: Product family positioning • Unique module offers: platform leveraging strategy
Step 3. Customer Needs Gathering	Skip	Single Voice of Customer (VOC) & KJ across product variants, Customer-Needs Variety Analysis, Customer Preference Variety Analysis, Tree of External Variety		• Weights and ranks of customer needs • Customer-preferred attribute values and price
Step 4. System Requirements Definition	Manually Listing Out	Product-Line House of Quality, Worst Case Condition Analysis		• System requirement targets and allowance on all product variants • Mapping between system requirements and customer preferred attribute values on all product variants (Portfolio Quality Functional Deployment (QFD)) • Identification of product variant with least margin on each requirement
Step 5. Functional Requirements	Skip	FUNCTION	COMPONENT	• List of functionality needed in each product variant • Mapping of functional importance • Modular drivers
		Requirements to Functions, Function Variety Management, Functional Requirement Cluster Analysis	Modular Function Deployment (MFD) drivers	
Step 6. Generic System Architecture	Systems Block Diagrams	Function Block Diagrams,	Product Structure Model Module Interface Graph	• Partitioning of product functions or components into subsets
Step 7. Component Alternatives	Manual	Function Component Matrix methods, Variety Allocation Model	Component Variety Management, MFD Matrix	• List of alternative sizes or technologies for components • Evaluation of components on module drivers
Step 8. Module Boundary Definitions	Manual	Module Function Heuristics	Hierarchical and DSM Clustering, Strategic Decision-Model, Module Process Chart, MFD cluster analysis	• Schematic module boundaries
Step 9. Architecture Roadmap and Future Uncertainty Management	Skip	Technology Roadmapping, Module Roadmapping, Product Family Roadmapping		• Plan for successive module and product upgrades • Future prediction of expected performance by reflecting future technology
Step 10. Commonality Assignment	Manual Match to Segmentation	Hierarchical clustering, Commonality metric Optimization, Heuristics		• Number of different sizes of each module (number of module variants) • Sharing of module variants by product variants
Step 11. Architectural Module Sizing	Manually Listing Out of Component and Interface Functional Requirements	Multi-Product Multidisciplinary Design Optimization (MDO) Trade Studies, Roadmap Robustness MDO Trade Studies		• Functional specification of each module variant • System specification of product variants • Functional Interface Specification for each module interface
Step 12. Architecture Concept Layout	Manual layout and definition of Interface Requirements	Module Interface Graph		• Each product variant concept-level layout geometry • Interface specification for each module
Step 13. Architecture Downselection	Skip (Consider only one Architecture)	Multi-Product Scoring Charts, Multi-Product MDO Trade Studies		• Documentation on relative benefits of selected modular architecture over alternatives on all product offers

TABLE 4: RECOMMENDATIONS FOR SUBSYSTEM COMMONALITY (GRAY AREA)

Subsystem	Design Parameters	Small UGV	Medium UGV	Large UGV
Chassis	Length			
	Width			
	Height			
Mobility	Wheels/Tracks			
	Wheel Diameter			
	Track Width			
	Wheelbase			
Batteries	Length			
	Width			
	Mass			
Manipulators	Outer Arm Radius			
	Arm Length			
	Number of Links			

TABLE 5: RECOMMENDATIONS FOR SUBSYSTEM SIZES [32]

Subsystem	Design Parameters	Small UGV	Medium UGV	Large UGV
Chassis	Length (m)	0.56	0.59	0.66
	Width (m)	0.23	0.22	0.30
	Height (m)	0.32	0.33	0.34
Mobility	Wheels/Tracks	Tracks	Tracks	Tracks
	Track Diameter (m)	0.26	0.26	0.26
	Track Width (m)	0.03	0.03	0.13
Batteries	Length (m)	0.11	0.11	0.11
	Width (m)	0.06	0.06	0.06
	Mass (kg)	1.40	1.40	1.40
Manipulators	Arm Radius (m)	0.02	0.02	0.02
	Arm Length (m)	0.57	0.52	0.31
	Number Links	3	3	3