Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 44 (2016) 67 – 72

www.elsevier.com/locate/procedia

6th CIRP Conference on Assembly Technologies and Systems (CATS)

Automated analysis of interdependencies between product platforms and assembly operations

Amir Hossein Ebrahimi*, Knut Åkesson*, Pierre E. C. Johansson*, Thomas Lezama*

[a] Department of Signals and Systems, Automation Research Group, Chalmers University of Technology, Gothenburg, Sweden
[b] Volvo Group Trucks Operations, Gothenburg, Sweden

* Corresponding author. Tel.: +0-000-000-0000; fax: +0-000-000-0000. *E-mail address:* amir.ebrahimi@chalmers.se

Abstract

Configurable products, like vehicles, face the challenge of handling all possible variants which are needed to answer the various customer needs. For these configurable products the support of all variants need to be addressed both by the design phase and the production phase. The product design phase and the production phase are linked together via operations. These operations model how each part of the bill-of-material is assembled to the final product. Operations also have inner relations among themselves, namely the precedence constraints, stating the order in which different parts can be assembled. Considering only the precedence constraints a product can generally be assembled in various different ways. It is through line balancing that the operations are assigned to different stations and/or assembly workers. The bill-of-material for each configurable product might be different with each variant, which will result in different set of operations. However, due to the precedence rules among the operations not all sets of operations might be possible to complete. The contribution in this paper is a automated method that can determine if all possible product variants can be successfully assembled while still satisfying precedence constraints between operations. The paper also includes an industrial example which further exemplifies the needed input for the method and the possible method outputs as a result of introducing a new variant to the product platform.

Peer-review under responsibility of the organizing committee of the 6th CIRP Conference on Assembly Technologies and Systems (CATS)

Keywords: Variability; Product platform; engineering Bill of Material; manufacturing Bill of Material; Assembly operation sequence

1. Introduction

In recent years there has been a dramatic increase in the number of product variants. This product variety can be seen across a wide range of products from simple products such as a light bulb through to extremely complex products like trucks and automobiles and airplanes. In some cases the number of product variants in the truck industry is around 500 variant families which lead to 10^{100} different valid product configurations. Some of the reasons behind this high increase in product variety include customers' demand for new product functions and features, different regional requirements, and large number of market segments with different need specifications [**?**]. A product variant usually requires a number of manufacturing operations to be assembled. These manufacturing operations can include operations for machining, assembly, painting, finishing and packaging. These manufacturing operations usually have precedence constraint relationship which can be other manufacturing operations [**?**]. Both in production management as well as in the supporting information technology, a central concept in product's design and manufacturing is each product's bill of material (BOM). Over the whole product lifecycle, a product will have more than one BOM describing its structure each from a different view point [**?**]. Some of the most important BOMs include engineering BOM and manufacturing BOMs. The high variety in products will cause the creation and updating of the BOMs to be a time consuming and error prone job.

In this paper we propose, an automated method that can determine if all possible product variants can be successfully assembled while still satisfying precedence constraints between operations. It will be shown how to model the product platform in a way to capture all the inherent variability in the product platform. Then this model can be used as a satisfiability problem. The goal of this model will be defined in a way which means that an "UNSAT" result of the analysis means all operation sequences are executable. While a "SAT" result will give a counter example of an operation sequence which can't be executed.

The paper is organised as follows. In section II the terms used for describing problems with product variability and the necessary operations modelling actions that are necessary to assemble the products are defined, while an introductory example is used to illustrate the concepts. Section III explains the en-

Peer-review under responsibility of the organizing committee of the 6th CIRP Conference on Assembly Technologies and Systems (CATS)
doi:10.1016/j.procir.2015.12.132

gineering documents which are mainly effected by variability and the problems which are the result of variability in creating some of these documents. Section IV discusses the system implementation in two steps. Firstly the modelling formulas which are needed to model the product platform are given. Secondly the result of implementing the introductory example in this method is examined. In Section IV contains the conclusions.

2. Product and production system variability

For a product platform the different products may be realized through various *variants*. These variants can be either hardware or software variants. Variants which are similar due to their functionality and/or nature are grouped together to form *variant groups*. Each set of chosen variants from different variant groups is named a *configuration*. *Configuration rules* or *constraints* are specific rules for how configurations can be formed that result in a product that can be ordered by a customer. Any product instance which satisfies all the specified configuration rules is named a *valid product instance*. The configuration for each valid product generate the product's *bill-of-material*.

Each variant will need one or more manufacturing operations to be assembled. Hence for each valid product instance there will be a set of needed manufacturing operations for the assembly of the product. Considering that each product instance has a unique bill-of-material each product instance will also have its own set of manufacturing/assembly operations. However, not all sequences of operations are feasible, for example due to geometrical constraints. But among those that are feasible some are not desirable for other reasons, for example due to the need for extensive fixturing or the risk of part damage or because they result in sequences that causes fatigue or discomfort for the operator. In [**?**] a method is presented for generating the desired precedence relations by considering the relations between parts, the liaisons, and by letting an engineer answers questions related to the desired sequences between operations realizing the assembly. In [**?**] it is shown that the set of feasible and desired assembly sequences are not compactly represented by precedence diagrams, instead an AND/OR graph are proposed to implicitly define all possible and desired assembly sequences. The AND/OR assembly graph can be represented using the general precedence constraints between operations that are introduced in [**?**]. In this work we will follow the operation concept as defined in [**?**] but extend the approach to allow the analysis all precedence constraints for a set of possible product instances implicitly defined by product platform modelled as a feature diagram.

We will use an example to illustrate how to model the product variability and the corresponding operations. An example of product platform for trucks is introduced to illustrate the problem and the approach. The product platform consists of variant groups and variants and constraints relating these. These variant groups include both hardware and software groups. The hardware variant groups include *cab*, *frame*, and *accessories* with their relevant variants. While there is also a software variant group which includes software variants for handling the *Load indicator* and the *AI clock*. Table **??** shows the truck product platform where the corresponding feature diagram are represented using a table. Each variant group has a specific characteristic named *"Group Cardinality"* which shows the number of variants that should be chosen from that specific variant group for each valid truck. For example the *Cab* variant group has the group cardinality of *"choose exactly one"*, which means that for a valid truck one of the variants *Cab V1* or *Cab V2* should be picked. On the other hand, the variant group *accessories* has the group cardinality of *choose at least one*, meaning at least one variant from this variant group should be picked for each valid truck. Each combination of these variants from the different variant groups is one configuration, some of which will result in a valid truck instance. As an example consider the two possible configurations:

Table 1. Feature model expressed using a table of the truck product platform. VG denote a variant group, while V denote variants within a variant group.

Truck Platform		
ID:	Name:	Group Cardinality:
VG_1	Frame	
V_1	Frame rigid	Choose exactly one
V_2	Frame tractor	Choose exactly one
ID:	Name:	Group Cardinality:
VG_2	Cab	
V_3	Cab VI	Choose exactly one
V_4	Cab V2	
ID:	Name:	Group Cardinality:
VG_3	Accessories	
V_5	Lower light bar	
V_6	Head lamp protector	Choose at least one
V_7	Wind deflector	
ID:	Name:	Group Cardinality:
VG_4	Software	
V_8	Load indicator	Choose exactly one
V_9	AI clock	

$$\text{Configuration 1} :\{V_1, V_3, V_4, V_5, V_8\}$$
$$\text{Configuration 2} :\{V_1, V_4, V_5, V_8\}$$

Configuration 1 is NOT a valid configuration according to the constraints in Table **??** as it has chosen both V_3 and V_4 while the *group cardinality* of variant group *cab* states that only one of the variants of this group should be present in any valid truck. But configuration 2 is a valid configuration as it satisfies all the stated configuration rules.

Table **??** shows some manufacturing oriented configuration rules which should be satisfied by each valid truck. As it can be seen each configuration rule is using at least one of the previously defined variant groups or variants. For example C_1 configuration rule states that each truck should have either the combination of *lower light bar* (V_5) and *load indicator* (V_8) or *head lamp protector* (V_6) and *AI clock* (V_9). Each configuration rule is a kind of a restriction which each valid truck should satisfy. If a configuration of a truck does not satisfy all of the defined constraints that configuration will not result in a valid truck.

Previously it was mentioned that Configuration 2 is a valid configuration according to the group cardinality constraints mentioned in table **??**. But if you consider the constraints in table **??**, then Configuration 2 is no longer a valid configuration as it contradicts C_2. This is because C_2 specifically states that