Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2016), *30*, 424–445.
© Cambridge University Press 2016 0890-0604/16 This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution, and reproduction in any medium, provided the original work is properly cited.
doi:10.1017/S0890060416000391

Automatic generation of design structure matrices through the evolution of product models

JAMES A. GOPSILL, CHRIS SNIDER, CHRIS McMAHON, AND BEN HICKS
Faculty of Engineering, University of Bristol, Bristol, United Kingdom

(Received October 1, 2015; Accepted May 31, 2016)

Abstract

Dealing with component interactions and dependencies remains a core and fundamental aspect of engineering, where conflicts and constraints are solved on an almost daily basis. Failure to consider these interactions and dependencies can lead to costly overruns, failure to meet requirements, and lengthy redesigns. Thus, the management and monitoring of these dependencies remains a crucial activity in engineering projects and is becoming ever more challenging with the increase in the number of components, component interactions, and component dependencies, in both a structural and a functional sense. For these reasons, tools and methods to support the identification and monitoring of component interactions and dependencies continues to be an active area of research. In particular, design structure matrices (DSMs) have been extensively applied to identify and visualize product and organizational architectures across a number of engineering disciplines. However, the process of generating these DSMs has primarily used surveys, structured interviews, and/or meetings with engineers. As a consequence, there is a high cost associated with engineers' time alongside the requirement to continually update the DSM structure as a product develops. It follows that the proposition of this paper is to investigate whether an automated and continuously evolving DSM can be generated by monitoring the changes in the digital models that represent the product. This includes models that are generated from computer-aided design, finite element analysis, and computational fluid dynamics systems. The paper shows that a DSM generated from the changes in the product models corroborates with the product architecture as defined by the engineers and results from previous DSM studies. In addition, further levels of product architecture dependency were also identified. A particular affordance of automatically generating DSMs is the ability to continually generate DSMs throughout the project. This paper demonstrates the opportunity for project managers to monitor emerging product dependencies alongside changes in modes of working between the engineers. The application of this technique could be used to support existing product life cycle change management solutions, cross-company product development, and small to medium enterprises who do not have a product life cycle management solution.

Keywords: Computer-Aided Design; Design Structure Matrices; Graph Theory; Metadata; Network Analysis

1. INTRODUCTION

Handling component interactions and dependencies remains a core and fundamental activity, with engineers resolving conflicts and satisfying constraints on an almost daily basis. Failure to consider these interactions and dependencies can lead to costly overruns, failure to meet requirements, product recalls, and/or lengthy redesigns. For example, issues in the length of cabling required in the A380 led to a $6.1 billion delay for the project, while Toyota has had to recall 625,000 vehicles because of faulty hybrid software (Calleam, 2011; Bruce, 2015).

In each of these examples, the companies were introducing innovative technologies to their product lines, which led to these complex products containing many more components that are more highly interconnected and dependent on one another, structurally, behaviorally, and functionally (Hamraz & Clarkson, 2015). This is further supported by Briggs (2012), who details that the Boeing 787 Dreamliner consists of over 300,000 parts modeled in computer-aided design (CAD), and the accompanying product data management system commonly saw between 75,000 and 100,000 accesses a week. This highlights the scale of CAD work being performed by a company that has locations across the globe and employs approximately 160,000 people. Due to this scale and the accompanying complexity, the ability to monitor and evaluate the interactions and dependencies between compo-

Reprint requests to: James A. Gopsill, Faculty of Engineering, University of Bristol, 0.36 Queen's Building, University Walk, Bristol BS8 1TR, UK. E-mail: J.A.Gopsill@bristol.ac.uk

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms. http://dx.doi.org/10.1017/S0890060416000391

nents at various system and subsystem levels remains a critical activity.

For these reasons, the development of supportive tools and methods continues to be an active area of research, with design structure matrices (DSMs) emerging as a common method. Originally conceived in the 1980s by Steward (1981) as a branch of graph theory, DSM seeks to understand the connected nature of engineering systems through a $N \times N$ matrix of interactions between system elements (Eppinger, 1997). The system elements can consist of individual components, assemblies of components, systems of components, engineers, teams of engineers, processes, and/or organizational structure to name a few. DSMs enable researchers to analyze and visualize product, process, organisational, and multidomain architectures of engineering products and projects.

Over the last 20 years, DSM analysis has been shown to provide insights into the product architecture of engineering products as well as the design process, and has helped increase engineering companies' understanding of their product (see, for e.g., Pimmler & Eppinger, 1994; MacCormack et al., 2006; Sosa et al., 2007; Jarratt et al., 2011). This understanding provides the basis for companies to better manage change propagation and risk, model change scenarios, and increase the modularity of their product designs. In addition, analysis of organizational architectures has resulted in companies being able to increase the performance of their development teams through restructuring in accordance with the product architecture (Eppinger, 1997; Browning, 2009). DSM has also been successfully applied to understand and evaluate supply networks as well as in integrating multiple complex systems such as the internal combustion engine and electric recovery and deployment system for hybrid vehicles (Fixson, 2005; Chen & Huang, 2007; Gorbea et al., 2008).

In order to generate DSMs, it is first necessary to understand the interactions between the system elements of interest. To capture these interactions, the majority of reported studies have used surveys, manual coding of engineering documents, structured interviews, and/or meetings with engineers (Fig. 1). In addition, many of the reported studies often use binary or ordinal values to determine the level and/or type of dependency between system elements. For example, Sosa et al. (2003) used a range from –2 to +2 in order to provide a weighted degree of dependency between system elements of a jet engine, while Gorbea et al. (2008) used a binary value to indicate the presence of dependencies between the component and function domains of hybrid vehicles. Although the capture of these data types has been shown to provide insightful results in terms of the dependencies within product architectures, it is argued that they may also be a limiting factor in providing even greater insights due to the granularity and resolution in which system dependencies can be assessed.

Because these data capture methods involve considerable interaction with engineers, there is inherently a cost associated with the engineer's time, potential error due to the subjectivity and precision of recall of the responses from engineers, and limitations in terms of determining the strength of the dependencies. Due to the time commitment required, data gathering is often a discrete activity that occurs at a single point in the project. Hence, a DSM can be considered a snapshot of the product architecture. For these reasons, it is contended that there is a need to explore alternative means to generate DSMs and, in particular, means that are automated, more ob-

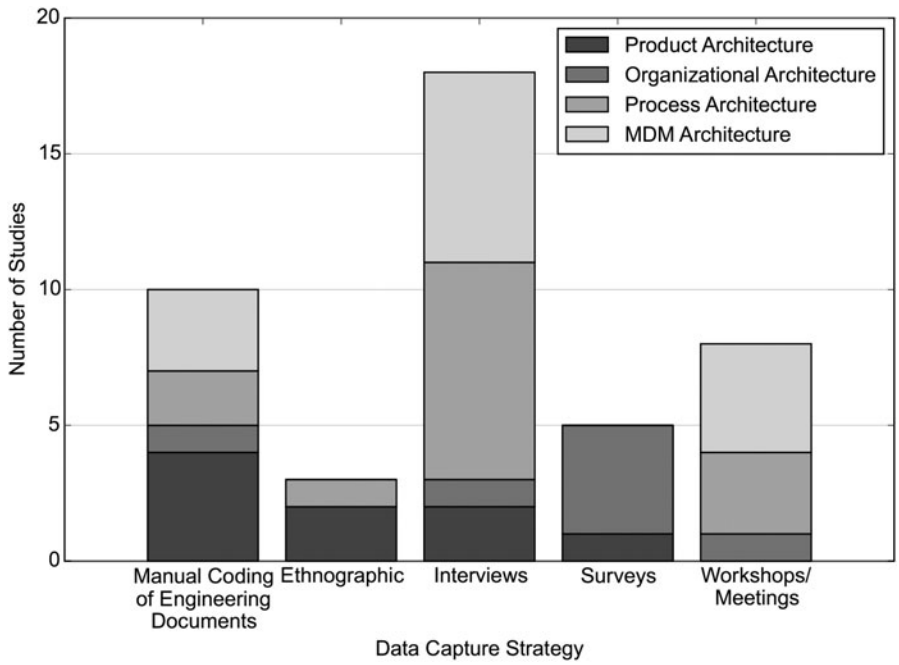


Fig. 1. Data collection methods from the studies in Eppinger and Browning (2012).

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

jective and real time, and that could support or supplement existing manual elicitation strategies (Senescu et al., 2012).

In the case of mature products such as aircraft and motor vehicles, the product architecture is generally well understood and the majority of dependencies are known. These are often captured through a combination of process models, product data management, product life cycle management (PLM), and associative CAD systems (Jarratt et al., 2011). However, such a proactive approach is not often possible for new, low value, or one-time large complex products/systems. It is also argued that an automated and real-time DSM analysis could provide additional support to existing solutions.

Many cases remain where a PLM change management solution may not be viable such as in small to medium enterprises where the expense may be too great. In addition, cross-company collaborative engineering projects can often provide additional challenges in monitoring dependencies due to the introduction of multiple disparate enterprise information technology systems having to interface within one another as well as issues with data security.

Therefore, the proposition for this paper is to investigate whether an automated and continuously evolving DSM can be generated from the changes to the digital models that represent the product. Such models include CAD, finite element analysis (FEA), and computational fluid dynamics (CFD). To explore this proposition, this paper presents the results from a DSM analysis of the evolution of product models for a formula student team. The objectives of this paper are threefold:

1. automatically identify component and subsystem dependencies across CAD models,
2. automatically identify cross-disciplinary dependencies through the analysis of the full range of product models, and
3. monitor the evolution of dependencies within the product models.

The paper continues by providing details of the method of data capture and the process by which DSMs have been automatically generated. The process uses the co-occurrence of product model edits to elicit potential candidates for dependency. This is then evaluated through the creation of a CAD, product model, and dynamic product model DSMs. The results of these have been verified through comparison with the product architecture as defined by the design team and results from previous DSM studies. In addition, particular attention is given to the affordances offered by dynamic product model DSMs. The paper then concludes by highlighting the key findings from the DSM analysis and areas of future work.

2. CAPTURING THE EVOLUTION OF PRODUCT MODELS

The data set used in this study has been generated from the evolution of product models produced by the Formula Student team at the University of Bath. Formula Student (also known as Formula SAE) is a motor-sport educational program whereby teams of students from competing universities create a single-seat race car that competes in various challenges set out by the competition organizers. The competitions are held worldwide and include events in the United Kingdom, United States, Australia, and Europe.

The team consisted of 33 engineering students in their final year of study who have undertaken a range of engineering courses, including automotive, aerospace, electrical, manufacturing, and mechanical. Therefore, it is argued that this reflects a highly multidisciplinary colocated engineering project environment.

In addition, the team have developed a bespoke lightweight CAD management tool that manages the naming convention, relationships, and organization of the CAD models. This is known as the bath automated parts system (BAPs). All work on the models is performed on the shared network drive.

In order to monitor the evolution of the product models, a Raspberry Pi (connected to the network) was used to monitor the status of the shared network drive at 20-min intervals (Raspberry Pi Foundation, 2015). More specifically, the folder structure alongside the metadata attributes of all the product models was captured. This included model size, date accessed, and date modified. The data capture was performed over a 13-week period from March 2014 to June 2014.

Table 1 summarizes the properties of the data set with respect to the various product models for the Formula Student car. CAD models total 1432 models, which have been up-

Table 1. *Summary of the formula student engineering models*

Description	Value
Weeks of data capture	13
Total CAD models created	1,432
Total CAD models assigned within BAPs	541
Brake system	13
Complete assembly	1
Electrical	9
Engine & drivetrain	142
Frame & body	145
Miscellaneous	1
Standard parts	45
Steering	46
Suspension	113
Wheels, wheel bearings, & tires	26
Unassigned CAD models	891
CAD model updates	10,145
CFD models	227
CFD updates	681
FEA models	43
FEA updates	121
WAVE models	95
WAVE model updates	245
OptimumK models	44
OptimumK model updates	88

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

Table 2. *Reduction in models through filtering based on level of edit activity*

Model Type	Filtered	Nonfiltered
CAD (assigned)	296	541
CAD (unassigned)	17	891
CFD	7	227
FEA	3	43
OptimumK	0	7

dated 10,145 times using the Siemens NX CAD package (Siemens, 2015). Table 2 shows that the BAPs CAD management tool classification covers 42% of the CAD models generated. Fifty-eight percent remains formally unclassified, and suggests that there is a high volume of temporary models as well as models that support other aspects of the design process, such as FEA and CFD modeling.

The second most active area in terms of product model updates are models generated from ANSYS CFX, with 227 unique simulation setup models and 681 updates (ANSYS, 2015). This is followed by the Ricardo WAVE one-dimensional gas dynamics modeling, which models the behavior of the internal combustion engine and has 95 simulation setup models and 245 updates (Ricardo, 2015). Last in terms of relative model activity (number of updates) is the FEA and OptimumK (a dynamics simulation package) modeling, which saw 43 and 44 unique models with 121 and 245 total updates, respectively.

3. DSM GENERATION

In general, the generation of DSMs has involved the elicitation of dependencies between systems and subsystems through expert opinion. In this paper, it is posited that models edited within a certain time period of each other can be considered to be candidates for a potential dependency. The probability of this dependency being true, alongside the potential strength of this dependency, is increased by a consistent reoccurrence of the models being edited within defined time periods.

An initial proof-of-concept investigation was conducted by Senescu et al. (2012), who sought to generate DSMs through the writing, viewing, and exporting of engineering documents within a cloud-based information management tool. The investigation concluded that dependencies were able to be identified; however, a significant amount of false positives were also identified due to pure chance of a document change co-occurring and/or the concurrency of work within the project. Therefore, this paper builds upon these results and proposes a seven-stage process for generating DSMs through the analysis of the co-occurrence of model edits, with each stage featuring techniques to reduce the false positive identification within the DSM (Fig. 2).

Stage 1 involves selecting the appropriate models that will be used as the basis for the DSM and is based on the level of activity recorded. The process then enters an iteration loop encompassing Stages 2 to 6, where a number of potential DSMs are generated by varying the time period in which co-occurrences can occur and the level of pruning. Pruning refers to the removal of edges (dependencies in this case) that are of a low weighting and is a method of removing noise from the matrix. Stage 2 involves generating the dependency matrix through the co-occurrence of model activity. In the first instance, it has been assumed that the matrix is directed (i.e., if Model A changes, then Model B changes). Stage 3 then examines this assumption and assesses the "directedness" of the matrix through pairwise comparison of the directed matrix elements (i.e., Model A to B and Model B to A). A decision is then made as to whether to continue the analysis with a directed or undirected matrix. This leads into Stage 4, where the matrix is weighted. The objective of the weighting is to evaluate the likelihood of a dependency while attempting to filter potential false positive dependencies introduced by the method of measuring co-occurrence through model edits. Once the weighting scheme has been applied to the matrix, pruning of the matrix can occur (Stage 5). This is particularly important as the level of pruning can greatly affect the partitioning of the matrix and a compromise must be sought because excessive pruning of the matrix may result in models becoming isolated. After the pruning, it is then a case of partitioning the matrix to reveal the structure and models with a high likelihood of being dependent (Stage 6). Stage 7 of the process then evaluates the generated DSMs in terms of the modularity, number of partitions, number of components, and remaining matrix size in order to determine the most appropriate time period and level of pruning for the given data set. This also reveals the sensitivity of the analysis with respect to the level of pruning and time period at which a co-occurrence is detected. This section continues by discussing each stage of the process in further detail alongside an example of the generation of the CAD DSM presented in this paper.

3.1. Initial model selection

The first stage of the process requires the specification of the models to be analyzed. In this study, the CAD, CFD, FEA, WAVE, and OptimumK models were specified as potential candidates for the DSM. This is achieved by detecting the respective file extensions for the product models. In addition, the number of updates for each model is also used as a filter, and Figure 3 shows the distribution of model activity for the models in terms of the number of changes made to the model and the number of days the model has been actively worked upon (i.e., an update has to have been made on the day for it to be counted). It can be seen that there is a slight positive correlation with the number of days the model has been actively worked upon and the number of edits to the model. In addition, one model in particular appears to exist outside of the main group of model activity. Upon inspection of the file name, it was revealed that this model embodies the full CAD assembly of the car, and it is likely that the number of

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

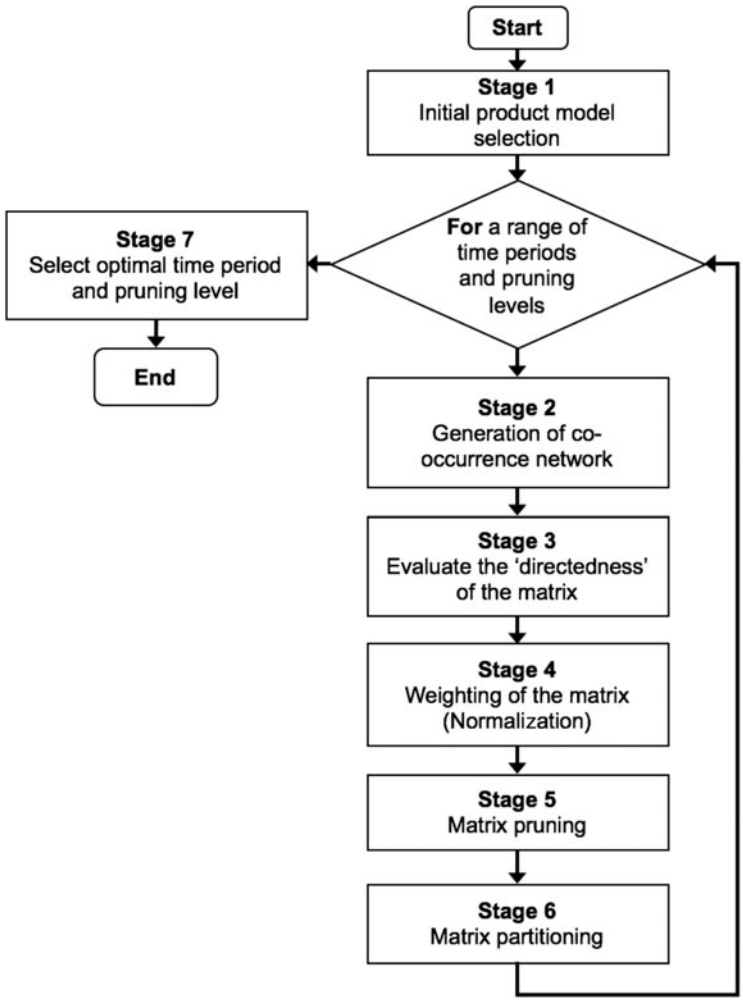


Fig. 2. Process of automatically generating design structure matrices.

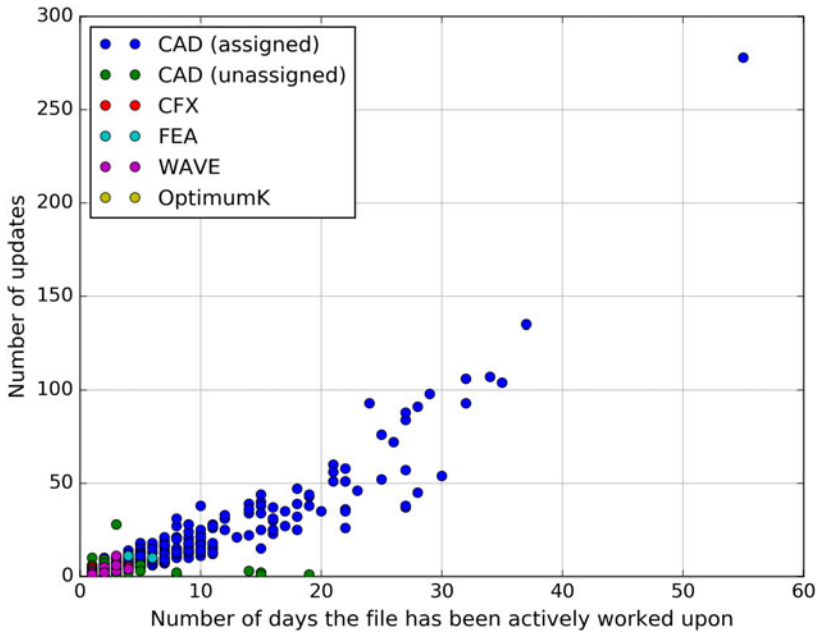


Fig. 3. Distribution of product model activity.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

updates to this model reflects the number of model updates arising from the various subassemblies of the product.

A closer inspection of the main groups model activity (Fig. 3) reveals that the CAD models embody a much greater activity and days of activity compared to the CFD, FEA, and WAVE models. This provides evidence to suggest that the CAD model plays a fundamental role in the Formula Student project with the CFD, FEA, and WAVE models used to support the development of the product rather than lead the design of the product.

To continue the analysis further, this stage requires the user to decide upon a suitable level of co-occurred activity given the context. In this case, models with an activity of greater than four updates have been selected. This includes the creation followed by three more updates. The aim is to remove the models that show little to no activity and, thus, may represent areas of the product that cannot be altered by the team (e.g., a bought in part such as the engine block and/or gearbox). Imposing a low threshold for the number of updates helps prevent an overreduction in the size of data set. For the case of the CAD models, the data set consists of 541 formally classified models reduced (cf. BAPs; Table 2) to 296 (55%) for further analysis while the unclassified CAD models are reduced from 891 to 17.

3.2. Co-occurrence matrix generation

Once the filtered data set has been defined, the generation of the DSM matrix can commence. This is achieved through the analysis of the co-occurrence of product model changes within the filtered data set. Figure 4 shows the process of identifying the co-occurrence of model changes. Taking Model A as the model of interest, the process identifies all the timestamps where the model has changed and generates a period in time where the occurrence of changes to and creation of other models are noted. Thus, if Model B has changed within this period, then this is defined as a co-occurrence and is a potential candidate for a dependency. The more often that this occurs, the greater the likelihood of the existence of a dependency. Currently, the impact of a change and the creation of a new model within the time period are treated equally. The maximum time period is defined by the user although the period can be shorter if model *A* changes again within its own time period as demonstrated by the 4, 3, and 2 h in Figure 4.

Figure 4 also shows the three possible scenarios that can occur. Scenario (i) occurs where Model A has changed and within the time period, Model B has also changed. A co-occurrence of model activity is consequentially recorded. Scenario (ii) occurs where Model B changes again within the same time period. In this scenario, a co-occurrence has already been recorded, and therefore, further changes are not recorded because they exist within the same time period. Scenario (iii) occurs where Model B changes within a period of inactivity of Model A. In this case, there is no co-occurrence of model activity.

The analysis of co-occurrence results in a DSM as illustrated in Figure 5, where a change in the model in the row vector has a likelihood of leading to a change in the model in the column vector. As a result of the generation process, the DSM is inherently directed where Model A can result in a change in Model B, but Model B may not lead to a change in Model A. This forms the initial matrix that, in the example of the CAD DSM with a max time period of 1 h, has led to a DSM consisting of 296 models and 32,508 directed co-occurrences ranging from 1 to 77 in weight.

In creating this matrix, there is an inherent assumption that the dependencies are directed, and thus, there is a need to assess the truth of this assumption and whether an undirected matrix could be equally effective in representing the data set (Stage 3). There is currently a wider range of validated partitioning algorithms in relation to undirected matrices, while directed partitioning algorithms remains a maturing field of research. Therefore, it is argued that, where possible, an undirected matrix would be a preferred result.

In addition, with many co-occurrences having a weight of 1 also highlights the likelihood that a number of false positive dependencies have been captured. Therefore, there is a need to weight the resulting undirected/directed matrix so as to reduce the noise and help reveal key co-occurrences (Stage 4).

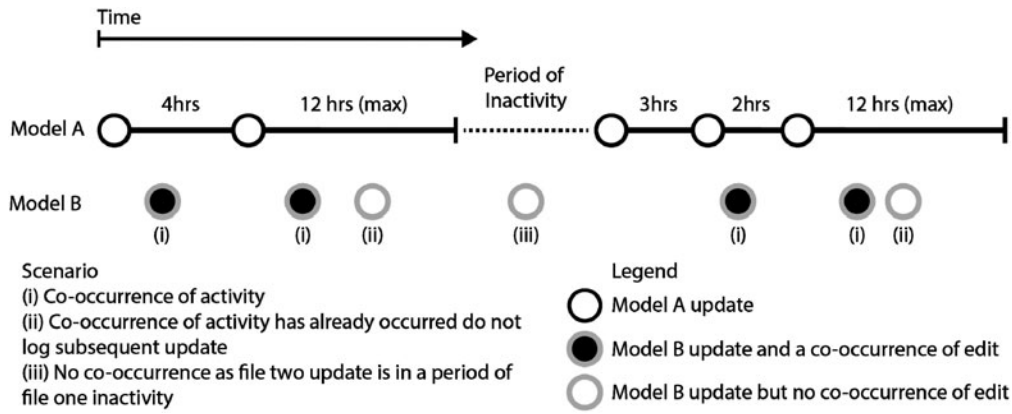


Fig. 4. Determining co-occurrence of model activity.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

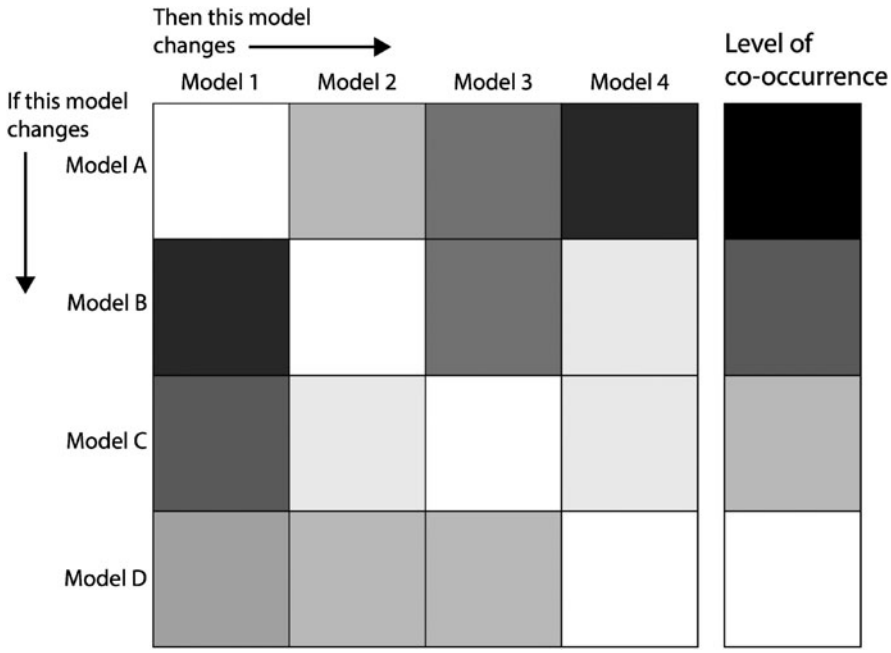


Fig. 5. Illustration of the design structure matrix from the co-occurrence of model activity.

3.3. Evaluating the "directedness" of the matrix

Stage 3 assesses the directedness of the generated matrix. This involves examining the number of one-way co-occurrences (10,850; i.e., a co-occurrence has been indicated from Model A to B but not from Model B to A) and the number of two-way co-occurrences (10,829; i.e., co-occurrences have been detected for both directions from A to B and from Model B to A). This is examined through the pairwise comparison of the two-way co-occurrences (i.e., the weighting of the co-occurrence from Model A to B and B to A) and the distribution of the one-way co-occurrence values. The results are shown in Figure 6. The pairwise comparison of the two-way co-occurrences (Fig. 6a) reveals that there is little difference between the co-occurrence values. It can be seen that 72% of all the two-way co-occurrences have a difference of less than 2 (e.g., if the co-occurrence value for Model A to B is 10, then the co-occurrence value for Model B to A would be 8 or 12 for a difference to be 2). Similarly, Figure 6b shows that the one-way co-occurrence values have relatively low co-occurrence values, with 77% of one-way co-occurrences also having a value less than 2 (e.g., if the co-occurrence value for Model A to B is 2 with no co-occurrence detected in the direction of Model B to A). In both cases, it is argued that there is little directionality within the matrix, and

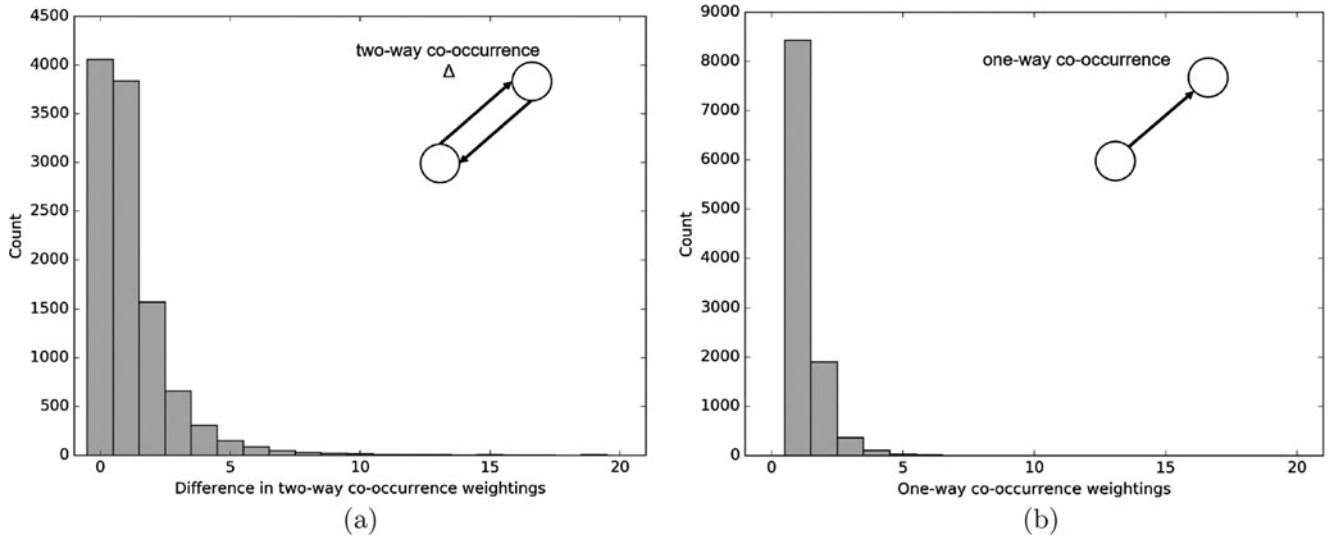


Fig. 6. Analyzing directedness of the computer-aided design structure matrices matrix.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

the one-way co-occurrences are likely to be false positive dependencies because they have low co-occurrence values.

From these results, it is argued that an undirected matrix can be equally representative of the data as a directed matrix. Therefore, all DSMs in this paper have been translated from directed to undirected networks prior to weighting by combining the number of co-occurrences for both directions (i.e., the sum of the weights from Model A to B and Model B to A).

3.4. Matrix weighting

In order to further reduce the likelihood of false positives and reveal the strongest candidates for dependencies, a matrix weighting scheme has been applied. The aim of the scheme is to reduce the potential influence of two or more designers modifying one or more unrelated documents at the same time. Two schemes are available, and the selection depends on whether a directed or undirected matrix has been generated from Stage 4.

For a directed matrix, each row vector is normalized by dividing the vector by the total number of changes made to the model that the row represents; that is, the co-occurrence between Model A and Model B would be divided by the total number of changes to Model A. This gives the ratio of co-occurrence between the two models in relation to the total model activity of Model A.

For an undirected matrix, each cell is divided by the sum of the total number of changes made to both models that the cell represents, that is, the number of co-occurrences between Model A and B divided by the sum of the total number to changes made to Models A and B. The premise is that higher ratios highlight a greater likelihood of the existence of a dependency between the two models. The weighting scheme also normalizes the co-occurrence of models and, thus, enables comparison between the range of model activities observed.

In the case of the CAD DSM, the undirected weighting scheme has been applied, and Figure 7 shows the impact of the weighting scheme on the distribution of the model co-occurrence values. It is immediately apparent that the weighting scheme increases the relative importance of some of the co-occurrences, while in the original matrix, there is a sharp drop off due to the large variability in the model activity across the CAD models. Consequentially, it is argued that the weighting scheme has potentially highlighted more of the positive dependencies.

3.5. Matrix pruning

Pruning is where the co-occurrences of given weights are removed from the DSM. In graph theory, this would be the removal of edges of a certain weighting from a network. The key objective of the pruning is to reduce the false positives in the matrix while maintaining the structure of the matrix and, in this case, revealing the candidate dependencies of higher likelihood. In this case, co-occurrences of a low weighting are removed because it is deemed that these are false positive candidates for dependency. The appropriate value at which to prune is determined later in Stage 7, where an optimization takes place in order to determine the most appropriate values for the time period and level of pruning.

Pruning the matrix also has the ability to disconnect models and groups of models from the rest of the matrix. In the first case, the process removes this model from the rest of the analysis, while in the second case, the group of models remain within the matrix as a separate group. These are commonly referred to as components in graph theory and can be considered to be a form of partitioning for the DSM. Even with a matrix that has been separated into a number of components, further partitioning can take place in order to uncover partitions within the components that remain.

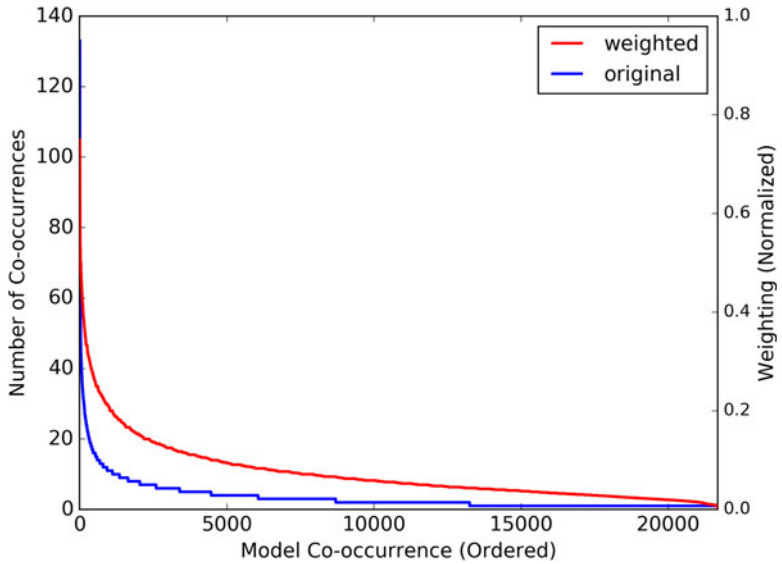


Fig. 7. Distribution of weighted co-occurrences within the computer-aided design structure matrices.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

In the case of the CAD DSM presented in this paper, the pruning level was set at 0.25, and this resulted in the matrix being reduced from 296 to 253 models due to disconnection and the number of co-occurrences being reduced from 21,679 to 613. The significant reduction in the number of co-occurrences further highlights the level of false positive that are introduced using this technique. In addition, the high number of remaining models in the matrix after pruning (85.4%) shows that the 613 remaining co-occurrences have the potential to reflect the main structural elements of the matrix. In addition, the pruning led to the creation of 13 components within the matrix. Components are akin to modules within engineering products, where the models within the component are highly connected and dependent from the rest of the product architecture. The number of components can be considered to be an indicator of how modular a product design is.

3.6. Matrix partitioning

Because the formation of the matrix involves the measurement of the co-occurrence of model activity and correspondingly results in potential false positive dependencies appearing, it is argued that typical DSM partitioning techniques may not be as suitable as other graph theory partitioning techniques. This is because DSM partitioning techniques are usually performed on data that is typically binary or ordinal, and there is an assumed confidence in the values attained from data capture because it is often through expert opinion.

In contrast, the co-occurrence measurements in this analysis are transactional and continuous, as well as having a chance of some of the candidate dependencies being false positives. Therefore, the structure of the data can be considered to be more akin to communication transactions such as e-mail and social media (Blondel et al., 2008; Pujol et al., 2009). In this field, the Louvain community-partitioning algorithm has been shown to perform well in identifying partitions of communities. For these reasons, the Louvain method has been adopted in this DSM generation process.

The Louvain community algorithms' objective is to generate a set of partitions for the matrix that returns the highest modularity value. Modularity (Q) is an assessment of the quality of the matrix partition and is defined as (Newman, 2004):

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\partial(c_i, c_j), \tag{1}$$

where $m = (1/2)\left(\sum_{ij} A_{ij}\right)$ and is the number of co-occurrences within the matrix; ∂ is the Kronecker delta function, which is 1 if a co-occurrence exists between two models and 0 otherwise; $(k_i k_j)/2m$ is the probability that a co-occurrence may exist between two models, where k_i is the number of models that have co-occurrences with model i and k_i is the number of models that have co-occurrences with model j; and A_{ij} is the weighted co-occurrence between the two models in the matrix.

In order to obtain the highest modularity partition, the algorithm iterates between two steps. The first assigns each model to its own partition. This is then followed by the algorithm sequentially moving one model to a different partition and calculating the change in modularity. From this, the maximum modularity change can be identified.

The second step merges the models together to form a partition of models and combines the co-occurrences of model activity to form single co-occurrence links to the rest of the matrix, and self-loops are used to identify internal co-occurrences between the models within the partition. The aim is to achieve a partitioning whereby each partition is highly connected internally and weakly connected to one another.

Thus, it can be considered a form of hierarchical clustering, and the algorithm iterates until the modularity can no longer be increased by partitioning the models. This paper uses the community application programming interface implementation of the Louvain community-partitioning algorithm within the NetworkX python package (Hagberg et al., 2008).

In the case of the CAD DSM, with a time period of 1 h and pruning of 0.25, the partitioning achieves a modularity score of 0.851 with 26 partitions being generated. This value is comparatively high when compared to many other network data sets where a modularity greater than 0.3 has been considered to be a "good" level of partitioning (Newman, 2004). Although modularity has been typically been used as an indicator of a "good" partition, it cannot provide complete confidence because random networks are able to attain a range of modularity scores (Newman, 2006). In this study, the high score of 0.851 is expected given the typical hierarchical nature of product architectures that often have well-defined system and subsystem boundaries.

3.7. Sensitivity of DSM generation to time period and pruning values

As previously stated, one of the main challenges in automatically generating DSMs through the co-occurrence of model changes is determining the appropriate values for the time period in which co-occurrences are detected and the level of pruning. This is unlike a normal DSM approach, where there is an assumed confidence in the validity of the dependencies because they are typically elicited from a number of experts. The process therefore incorporates a stage to determine the most appropriate values and assess the sensitivity of the DSM to these parameters. This involves optimizing the matrix with respect to the following:

1. the quality of the partitioning achieved (i.e., a "good" system and subsystem product architecture can be identified),
2. the number of partitions detected (i.e., the level of subsystem granularity that can be attained),
3. the remaining size of the matrix after pruning (i.e., how representative the DSM is to the entire product architecture), and
4. the number of matrix components (i.e., how modular the product design is).

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

First, the quality of the partitioning achieved by the process is assessed by taking the modularity into consideration where a higher value is preferable because this highlights that the analysis is able to identify clear system and subsystem boundaries. Second, the number of partitions detected is also taken into consideration where the greatest number of partitions is sought because it provides the greatest granularity within the matrix. In terms of the CAD files, the greatest granularity would provide the lowest level of subsystem within the product architecture. Third, the effect of pruning is taken into account because increasing the pruning often leads to a greater loss of models due to disconnection from the rest of the matrix. Hence, it is desirable to ensure that a large proportion of the original matrix remains after pruning; thus, the DSM is as representative of the entire product architecture as possible. Fourth, the number of matrix components produced by the pruning is considered. Here, fewer components is preferable because components are disconnected from one another, and thus, there is a potential loss in understanding of the dependency between these components, yet it is also an indicator for the number of separate modules within a product's design.

To achieve this, a range of time periods and levels of pruning are considered exhaustively, and the modularity, number of communities, remaining network size, and number of components are determined. Figure 8 shows the results for each of these parameters where the dashed rectangles highlight the desired areas for each of the parameters. The effect of sensitivity with respect to the four matrix metrics are the following:

1. The partition modularity (Fig. 8a) has a positive correlation between the time period and level of pruning in order to sustain a high modularity score.

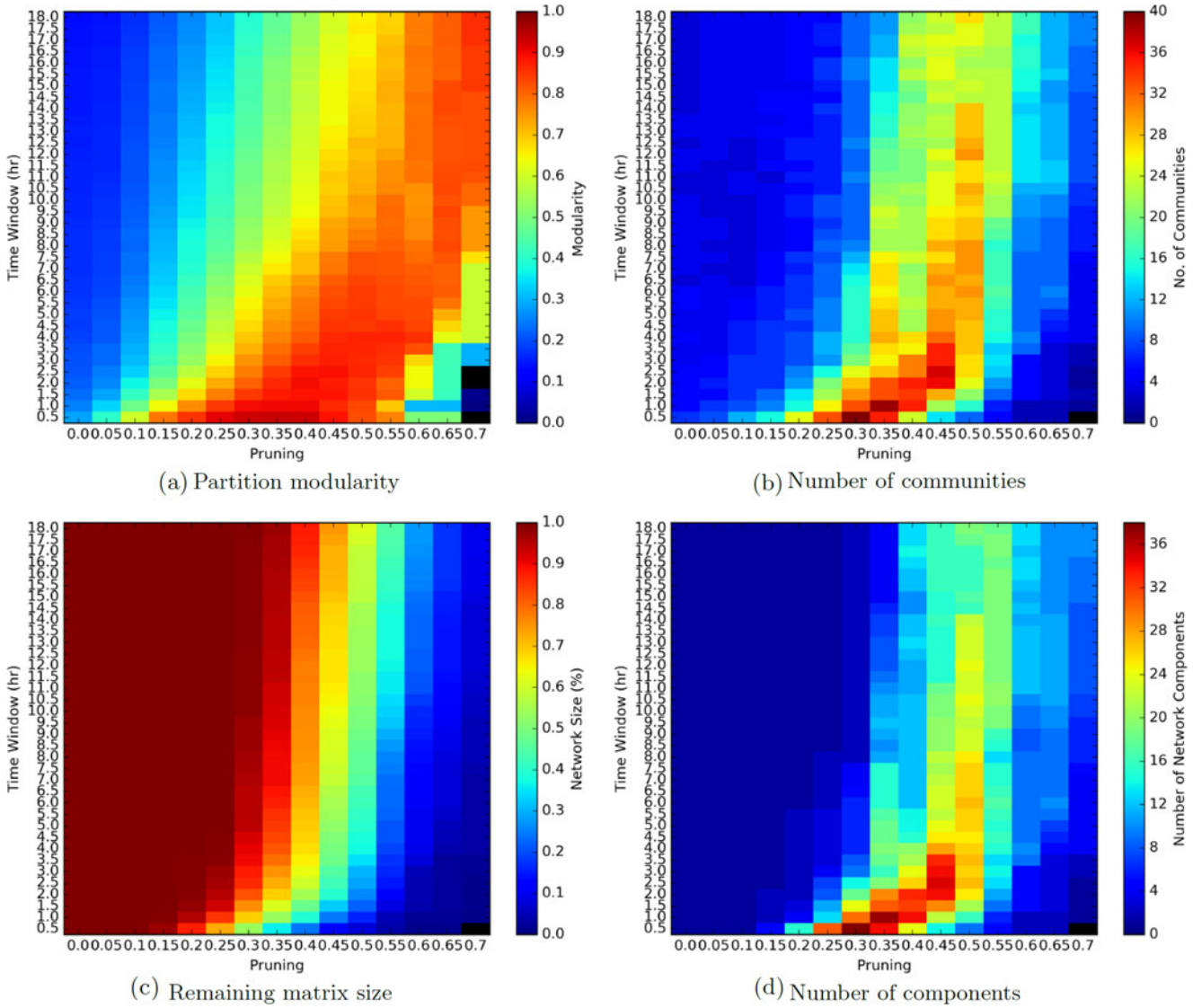


Fig. 8. Effect of time period and pruning on the community partition analysis.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

2. The optimum number of communities is more sensitive to both the level of pruning and time period with a maximum attained at 0.5 h and level of pruning of 0.3 (Fig. 8b).
3. Beyond a pruning level of 0.35, the remaining number of models within the DSM decreases (Fig. 8c).
4. The number of components (Fig. 8d) follows a similar pattern to the number of communities whereby it is both highly susceptible to changes in the time period and level of pruning.

Because the ideal time period and level of pruning differs for each of the parameters, it can be seen that a compromise must be made. In addition, black areas are a result of the entire removal of the matrix by pruning.

To achieve this, the four metrics are combined as shown in Equation (2) to determine an overall measure O, where M is the modularity of the matrix partition, N_p is the number of matrix partitions, R is ratio of the network that remains after pruning, and N_c is the number of matrix components. The aim is to achieve the highest O, given a range of time periods and pruning levels.

$$O = \frac{MN_pR}{N_c}. \quad (2)$$

Figure 9 shows the result of this search for an optimal set of parameters and reveals a number of regions, such as 1 h and a 0.25 level of pruning and 6.5 h and a 0.35 level of pruning. In this paper, it has been decided that 1 h and a pruning level of 0.25 is the most suitable parameter set. For the purpose of this study, the four matrix assessment metrics have not been weighted, but it may be desirable to do so.

3.8. Summary

The previous sections have presented a detailed description of the seven-stage process undertaken to generate the DSMs alongside an example of generating a DSM for the CAD models from the Formula Student data set. Although much of the process can be automated, a number of decisions have to be made given the context of the data set that is to be analyzed. First, the initial model filtering was based upon activity. In this paper, this was specified as a minimum of four updates. Second, it is also necessary to decide whether to continue with a directed or undirected co-occurrence matrix. Through investigation of the one- and two-way co-occurrence values, an undirected matrix was chosen. Third, this decision concerns the time period and level of pruning of the matrix. In order to make a reasoned decision on these values, the process introduces a search stage where these values can be chosen based on four measures: the modularity, the number of partitions, the number of components, and the remaining matrix size. The sensitivity of these parameters has been discussed based on equal weightings of the four measures. This revealed a suitable time of 1 h and pruning level of 0.25.

4. RESULTS AND DISCUSSION

As previously discussed in Section 2, the data set considered in this study involves the design of a Formula Student car and

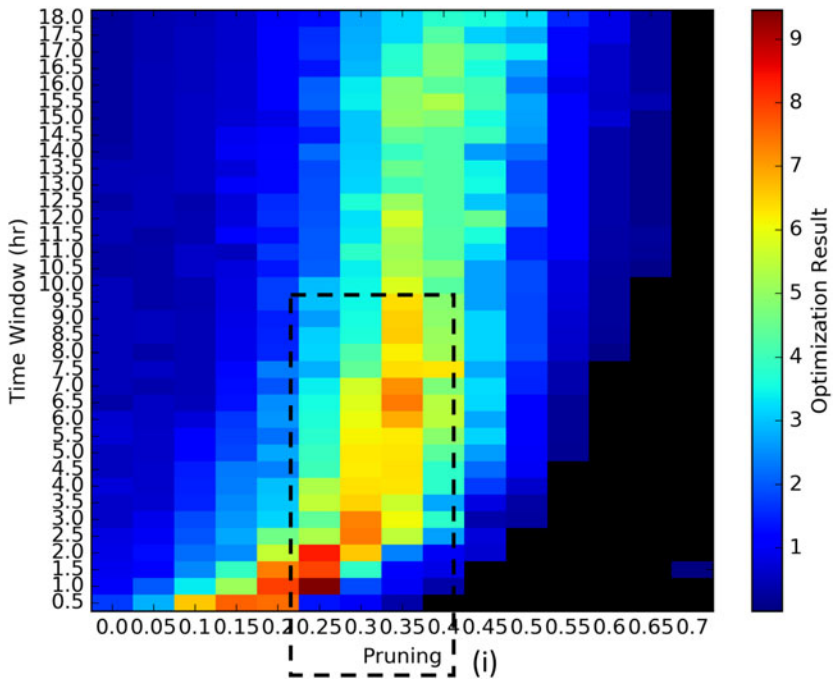


Fig. 9. Determining the appropriate time period and pruning values.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

the evolution of the associated CAD, CFD, FEA, and WAVE models. Correspondingly, three DSMs are constructed using the process described in Section 3. The first DSM is a CAD model only DSM. The second comprises CAD, CFD, FEA, and WAVE product models, with the OptimumK models not being considered due to a lack of file activity being recorded. The third DSM is a dynamic product model DSM that captures the evolution of dependencies across all product models in real time as the project progresses from week to week. The following discussion centers on the validity of the DSMs through comparison with the existing product architecture as defined by the team and results from previous DSM studies, and the additional insights that could be brought by an automatic DSM process.

4.1. CAD DSM

The properties of the CAD DSM are summarized in Table 3. The table shows that a high proportion of the models remain after the pruning stage (85.5%) with a core number of co-occurrences remaining (2.83%). This highlights that a significant number of candidate dependencies are likely to have been false positives. The modularity score of 0.85 highlights that a high level of structure exists within the matrix and suggests that the product architecture can be clearly separated into a number of subsystems. In addition, the high number of components (13) demonstrates that the products design is also highly modular in parts, an insight that would be difficult to ascertain through the existing hierarchical structure of the CAD models as the dependencies could span system and subsystem boundaries. However, a total of 12 additional partitions were generated through Louvain partitioning, which highlights that there are a number of highly integrative subsystems within the product architecture.

Table 3. *Summary of CAD analysis*

Description	Value
Undirected matrix	
Models	293
Co-occurrences	21,679
Pruned matrix	
Models	253 (85.5%)
Co-occurrences	613 (2.83%)
Components	13
Partitions	25
Modularity	0.850

Figure 10 shows the impact of the partitioning on the DSM structure. Initially, it appears that there is no structure to the DSM (Fig. 10a), but postpartitioning, a high-level of structure is shown to exist (cf. Fig. 10b). This high level of structure within the generated DSM is comparable to the DSMs generated by Sosa et al. (2003) and Gorbea et al. (2008) for a jet engine and hybrid vehicle, respectively. This result is also in line with manually curated DSMs that have been produced by Van Beek et al. (2010) and Maurer (2007) within the Formula Student context.

It has been noted by Eppinger and Browning (2012) that one of the key benefits of DSM is the ability to visualizse and interrogate subsystem dependencies through figures such as Figure 10b. However, it is also noted that this DSM can be considered very large in terms of the number of elements being visualized when compared to more traditional DSMs, which commonly feature between 10 and 100 matrix elements (Eppinger & Browning, 2012) Consequentially, such a matrix might require additional support in terms of navigation in order for an engineer to comprehend all the dependencies but also demonstrates the greater level of granularity afforded by the automatic process.

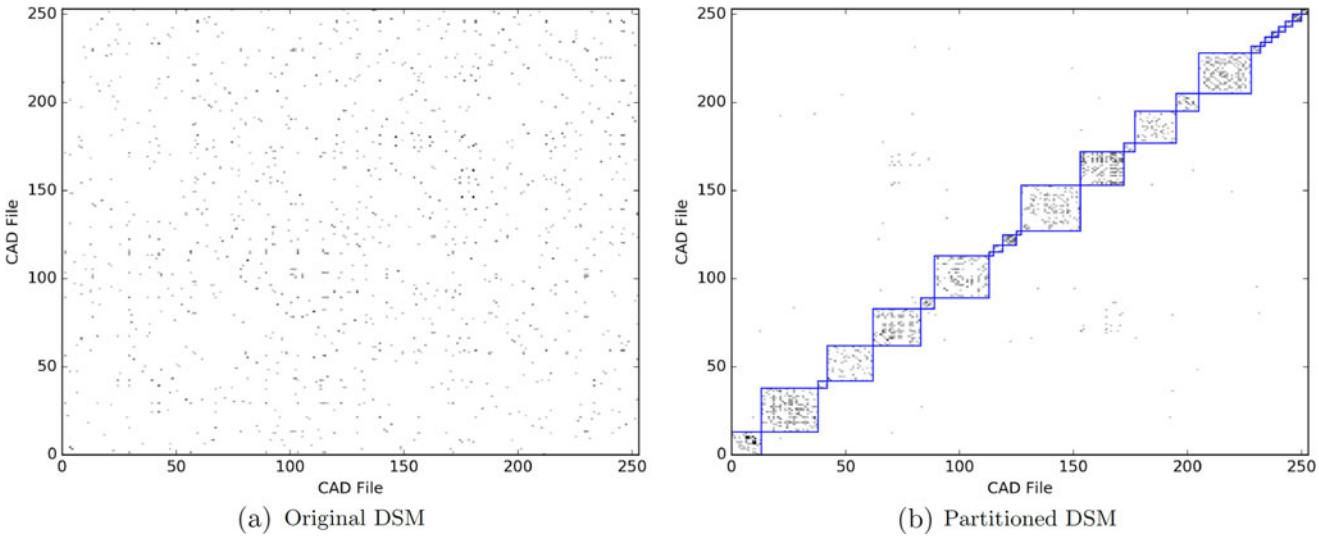
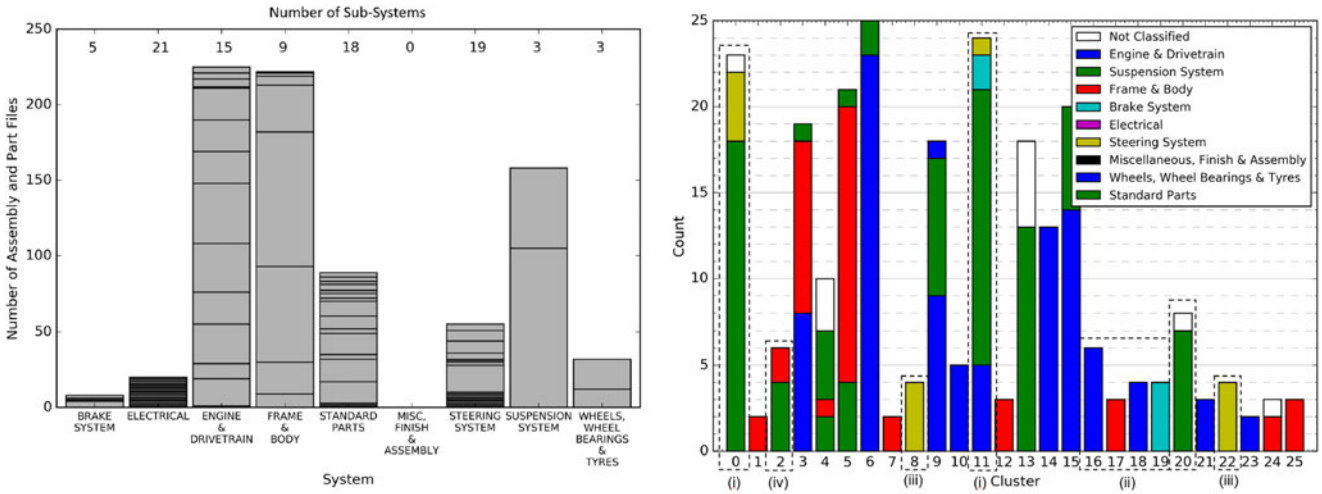


Fig. 10. Community partitioning on the co-occurrence of computer-aided design models.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391



(a) Bath automated parts system subsystem structure

(b) Composition of computer-aided design partitions based on the team designated product architecture

Fig. 11. Comparison of bath automated parts system and design structure matrices structures.

It is immediately apparent that the partitions generated from the co-occurrence of file activity have identified many dependencies that are cross system and subsystem. Therefore, it is argued that the analysis provides further insights into the design of the Formula Student car that the current CAD management system does not provide. While BAPs manage and monitor the product architecture, the automatic DSM process offers potential for identifying dependencies beyond purely architectural (Figure 11). These partitions could relate to functional, structural, and/or process dependencies.

Focusing on the partitions generated by the automated DSM process, four key insights can be drawn. A more complete overview of the partition breakdown is presented in Table 4.

1. Areas of the product exist that are highly interdependent across all the team-defined subsystems. Hence, a change in one of these models could require significant rework across many subsystems and might be an area that requires careful monitoring, particularly in later stage design.
2. There are a number of smaller partitions that consist of only one team-defined subsystem. Thus, a change in one of these partitions may only require changes to a small subset of parts in the product and potentially an engineer from a single discipline.
3. There are a number of partitions that arise from one subsystem, and this may indicate that the subsystem can be

Table 4. *Summary of CAD analysis*

Partition	Dependency Interpretation
0	Functional dependency between the steering system and suspension system, CAD models may influence the steering and dynamic performance of the car
1, 7, 12, 17, & 25	Interdependent frame and body system, a potential single module of product
2, 4, & 5	Dependency between frame and body, potential structural interfaces between systems
3	Dependency between engine and drivetrain and frame and body components, potential engine fixtures locations with respect to the frame
6	Engine and drivetrain module with interfaces with some standard components
8 & 22	Interdependent steering system partition, potential single module of the product
9	Engine and drivetrain, suspension system, and wheels dependency, potential functional dependency affecting dynamic vehicle performance (e.g., body role)
10, 14, 16, 18, 21, & 23	Interdependent engine and drivetrain or wheel system, potential single module of product
13, 20	Interdependent standard parts partition, potential identification of standard parts from nonclassified parts
15	Dependency between suspension and wheel systems, potential structural and functional dependency
19	Interdependent braking system partition, potential single module of the product
25	Dependency between frame and body systems, potential identification of standard parts from nonclassified parts

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

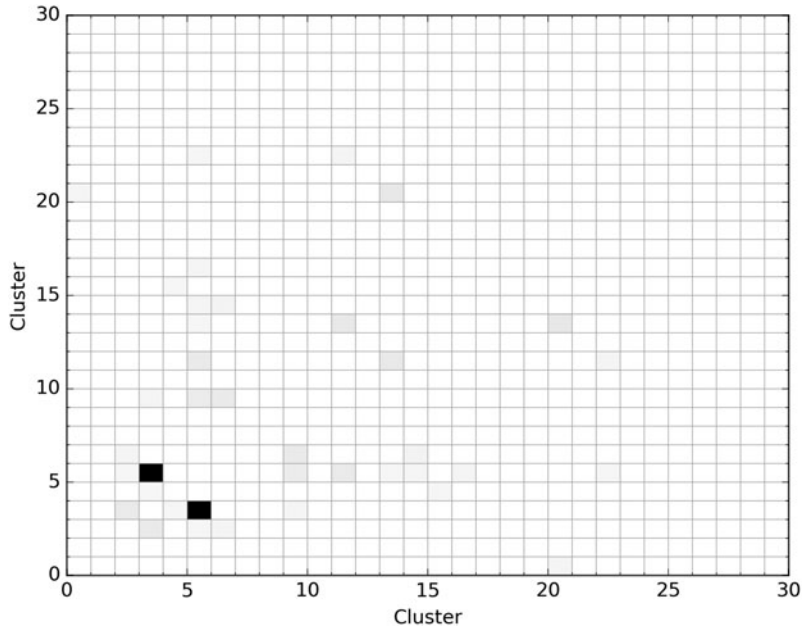


Fig. 12. Induced matrix from grouping the partitioned computer-aided design models.

further decomposed. For example, further decomposition might be useful for the purpose of resource allocation.

4. The partitioning has identified distinct subsystem dependencies. Partition 3 is one such example, where specific models of the Engine & Drivetrain are related to models of the Frame & Body. This is a logical outcome because the engine fixtures are located to specific areas of the frame.

From this evaluation, it is argued that the DSM provides insights that are consistent with the known structure of the Formula Student vehicle, and the results are consistent with DSMs generated from more traditional approaches. In addition, a number of additional benefits have been revealed by using an automatic DSM generation process. Therefore, this provides confidence in the proposed approach to produce DSMs.

The partitioning of the matrix also enables the models to be meaningfully grouped by the intensity of the dependencies. This is in effect moving up a level in the hierarchy of the product architecture, and the results are shown in Figure 12. The rows and columns represent the partitions from the previous matrix (Fig. 10), and the color provides a visual indication of the level of dependency between the partitions. The matrix is symmetric as it remains undirected.

It can be seen in Figure 12 that partitions beyond 15 appear to not have dependencies with the other partitions, and this is due to the fact that these are small components that were generated by the pruning of the matrix, and the potential dependencies were removed as false positives. Partitions 1 to 9 do show dependencies between one another, and in particular, there appears to be a high dependency between Partitions 3 and 5, and 5 and 9. This suggests, that Partition 5 is a fundamental integrating partition of CAD models and, hence, likely to be a core subsystem of the product, which in this case is the Frame & Body. A change in this partition is likely to have a substantial impact across many other partitions within the DSM.

4.2. Product model DSM

Similarly to the CAD DSM, the properties of the product model DSM are given in Table 5. Consistent with the CAD DSM, the pruning of the matrix achieved a greatly reduced number of dependency candidates (80%) while maintaining a high proportion of the original matrix (75%). Again, the modularity value of 0.769 is comparable with the CAD DSM with 30 components being generated and 45 partitions.

Figure 13 shows the results from the partitioning of the matrix using the DSM process. Similar to results shown in the

Table 5. *Summary of CAD, CFD, FEA, and WAVE DSM*

Description	Value
Undirected matrix	
Models	488
Co-occurrences	51,245
Pruned matrix	
Models	367 (75.3%)
Co-occurrences	1,020 (1.99%)
Components	30
Partitions	45
Modularity	0.769

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

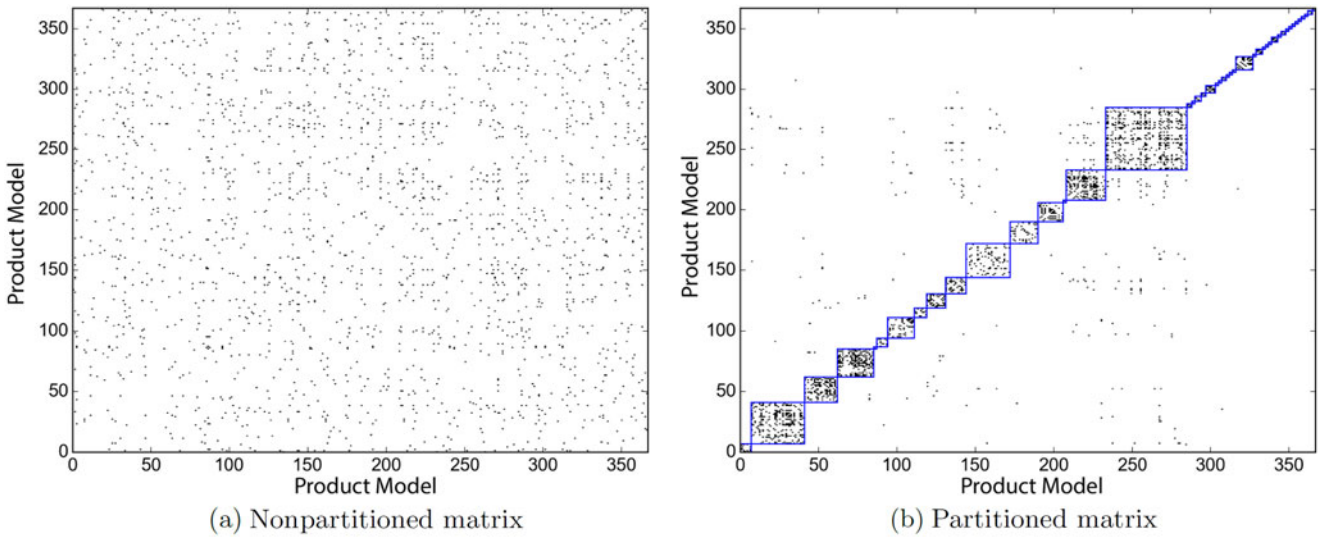


Fig. 13. Community partitioning on the co-occurrence of product models.

CAD DSM (Fig. 10), Figure 13a shows a seemingly random matrix while the partitioning result in Figure 13b reveals that there is a significant level of structure within the product model DSM. However, in contrast to the CAD DSM, there appears to be greater interdependency between the partitions, which is shown by an increase in co-occurrences that exist outside the partitions denoted by the blue squares.

To investigate this result further, the model types have been mapped to the partitions generated from the DSM. Figure 11 shows the results of this mapping with four interesting features being exposed.

1. Each model type has a partition that consists solely of its respective model. This is a logical output because many of the components within the product may not require an in-depth analysis in relation to CFD, FEA, or WAVE modeling.
2. The existence of a single WAVE model partition is unrelated to the rest of the other modeling types. Because WAVE models involve the analysis of the internal combustion parameters of the internal combustion engine, the modeling looks to optimize the engine mapping rather then look to change the design of the engine itself. Therefore, this is confirmation that the DSM has the ability to detect this as a separate partition and further provides evidence of the technique's validity. This is may also be the case for some of the CFD models where they may be testing new components that did not become part of the final Formula Student car.
3. Identification of dependencies between areas of the CAD model and the FEA and CFD models in this analysis highlights that a change in any of these models may require further FEA and CFD analysis to be performed.
4. It is argued that a single partition consisting of CAD, CFD, and FEA models may be an area that will require monitoring by project management because any change with these models may lead to considerable rework across multiple engineering disciplines

As with the CAD DSM, the product model DSM can use the partitions to group the models and form a secondary matrix of these partitions. Figure 14 shows the results of this induced product model matrix, and the result is similar to that of the CAD DSM induced matrix. Above Partition 16, there are no dependencies observed between the partitions because these are the components that have been generated from Pruning 16, and below Partition 16 show the inner partitions of the larger components within the network, and it can be seen that there exists potential dependencies between these partitions. In particular, Partition 16 shows the greatest number of dependencies between the other partitions, and looking back to Figure 15, it can be seen that this partition is the largest in terms of number of models, and this may highlight the key interfacing components within the product.

4.3. Dynamic network analysis of models

The final aspect that has been investigated is the potential of automatically generating DSMs during the engineering project. This has been achieved through the generation of product model DSMs based on the weekly model activity during the Formula Student project.

Figure 16 provides a summary of the matrix statistics and how they evolve throughout the weeks of the project. It is immediately apparent that a gap exists within the updates of the product models between Weeks 7 and 8, and this is due to the students going for Easter vacation and work ceasing on the project. Figure 16a shows that there is a consistency in the level of work being performed on the CAD, CFD, and FEA models, while the WAVE model activity only appears between Weeks 4 and 6. Figure 16b shows the statistics for

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

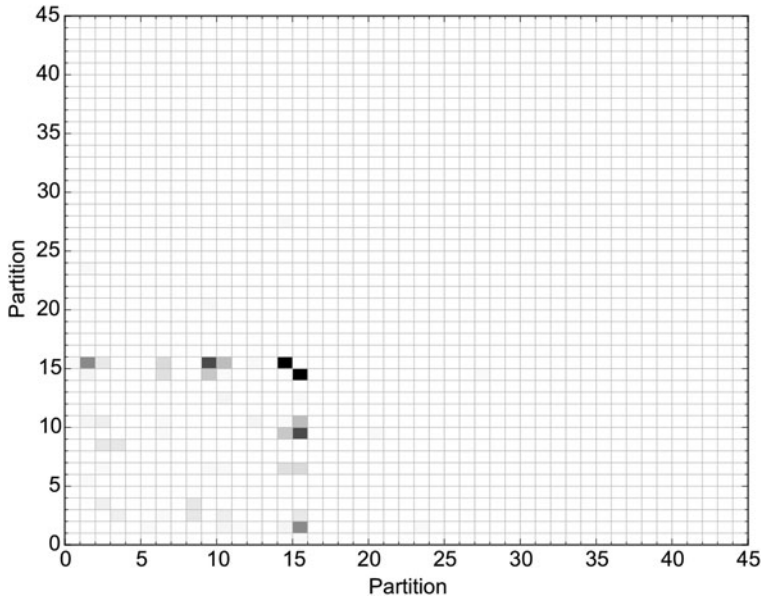


Fig. 14. Induced matrix from grouping the partitioned product model design structure matrices.

the DSMs created for each week. An interesting feature of the DSMs is the variability in the size of the matrix between the weeks and modularity score for the partitioning. It appears that there is a limit in generating "good" partitions with low model activity. This is a logical result, and the lack of model co-occurrence activity makes the identification and distinction between positive and false positive dependencies more challenging. It could also be more difficult to detect subsystem structures on a weekly basis because the team are focusing on different elements of the product architecture. Thus, the partitions may be more reflective of the areas of work rather than identifying a particular product subsystem as with the previous CAD and product model DSMs.

Figure 17 shows the DSMs for Weeks 2 to 7 of the Formula Student project. The challenges of low model activity and the ability to generate a DSM is more apparent when one compares Figures 17c and 17d. Figure 17c clearly has fewer active models, and if it were to be compared to the entire DSM from Section 4.2, this may show the mode of working of the team as well as the specific area of work. In addition, comparing these DSMs to the previous CAD and product model DSMs, it can be seen that there appears to be a great deal

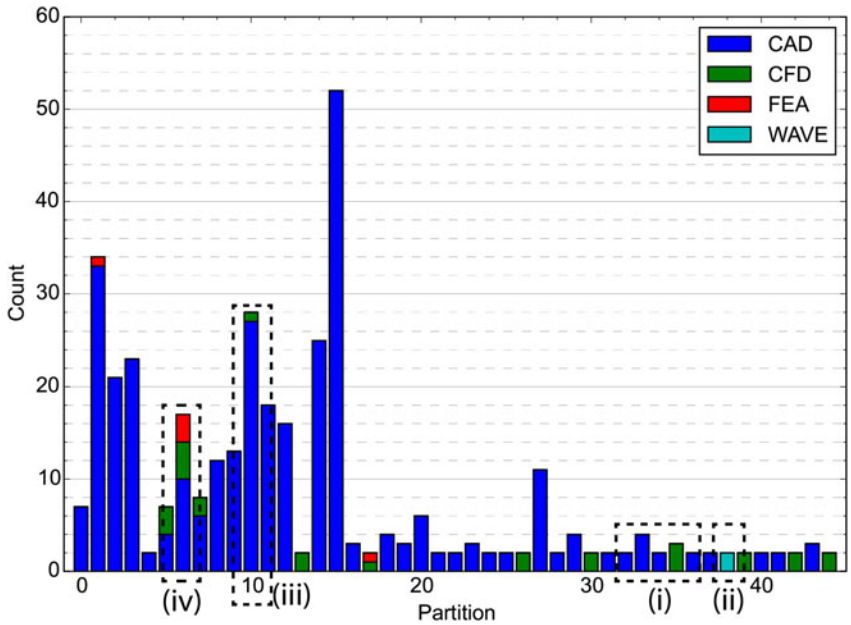


Fig. 15. Composition of product model design structure matrices partitions based on model type.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

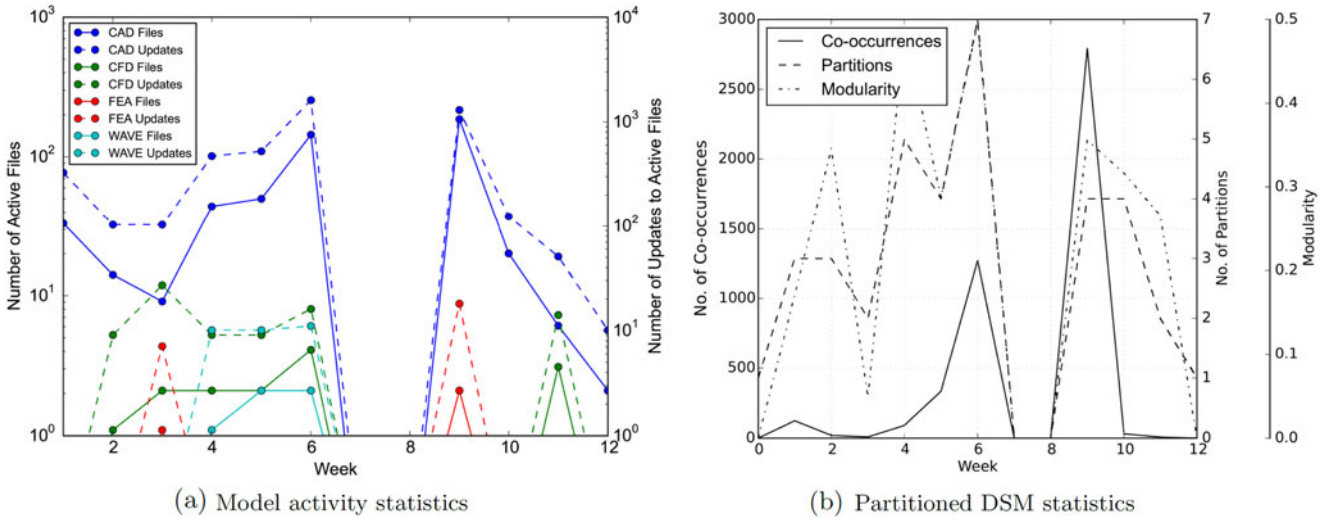


Fig. 16. Model activity and design structure matrices statistics for the dynamic design structure matrices.

more interdependencies between the partitions that are generated. This may be because the analysis is focusing upon a week of activity where it is suggested that the team are focusing on integrating models within a few areas of the product at a time. Because the analysis has used the same time period and pruning for the consistency across the analysis, it may be the case that these values may not be suitable for the analysis of individual weeks, and hence, there may be more false positive within these DSMs.

Figure 18 provides further details on the composition of the partitions from Weeks 2 to 7 of the Formula Student project. From comparison of these weekly compositions, four interesting features emerge.

1. Week 2 (Fig. 18a) reveals partitions consisting of models from all the subsystems in the product. In particular, Partition 1 focuses on the Suspension, Frame & Body and Steering System, so it is suggested that these components are forming the initial chassis for the car. Moving to Partition 2, this solely consists of standard parts and parts that could not be classified through their naming convention. This provides evidence to suggest that the nonclassified parts are part of the standard library of parts of the car. Partition 2 consists of parts from the rest of the subsystems for the car, and this could be the initial placeholders for the arrangement of the subsystems within the car.
2. Concentrated working on particular subsystems has been identified in Week 3 (Fig. 18b) and Week 4 (Fig. 18c) though the majority of the partitions consist of one model type. This may also indicate that these subsystems require a certain level of maturity before being introduced to the rest of the product architecture.
3. Partition 1 in Week 4 (Fig. 18c) highlights a particular area of multidisciplinary collaboration between the Frame & Body development of the CAD alongside the associated CFD and CFD and FEA analysis.
4. A change in the mode of working within the team emerges after Week 4. Weeks 3 and 4 (Fig. 18b,c) focused on specific areas alongside CFD and FEA analysis; Weeks 4–7 (Fig. 18d–f) show a high-level of integrative working between the various subsystems of the car. Initially, the majority of partitions within Weeks 4 and 5 (Fig. 18c,d) consist of either two or three different subsystems, which could be used to identify specific dependencies between subsystems. The level of integrative working continues to grow in Week 7 where both the number of partitions and number of subsystems within each partition increase. This shows that as the project and product develop, the more integrative the work becomes.

These results highlight the potential of automatically generating DSMs as engineering projects progress and the potential insights that they could provide to support engineering project management and the design process. In addition, a better understanding of how the multitude of dependencies evolve and develop within a product could also be produced, and having a library of DSMs from past product developments could be used to support and monitor the development and progression of new products.

4.3. Summary

The results from DSMs generated from the Formula Student project has provided considerable evidence to suggest that the process of generating DSMs through the co-occurrence of product model updates provides valid and useful information for engineering project management. In addition, the results show the potential for this process to produce dynamic product model DSMs, which would have previously been a time consuming and potentially costly endeavor. Table 6 provides a summary of the key features that have been identified through the generation of the CAD, product model, and dynamic product model DSMs.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

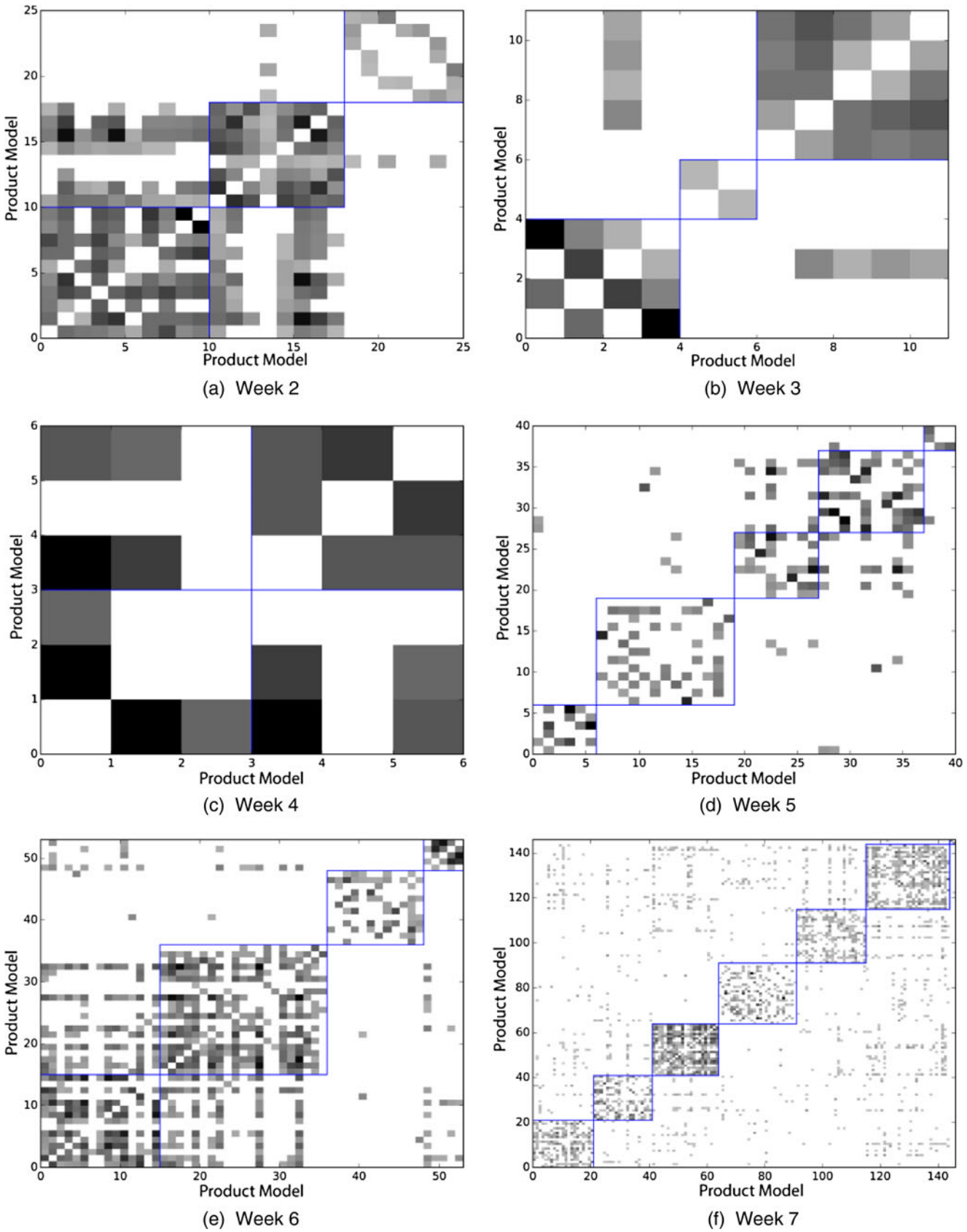


Fig. 17. Partitioning results for Weeks 2 and 7 of the formula student project.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

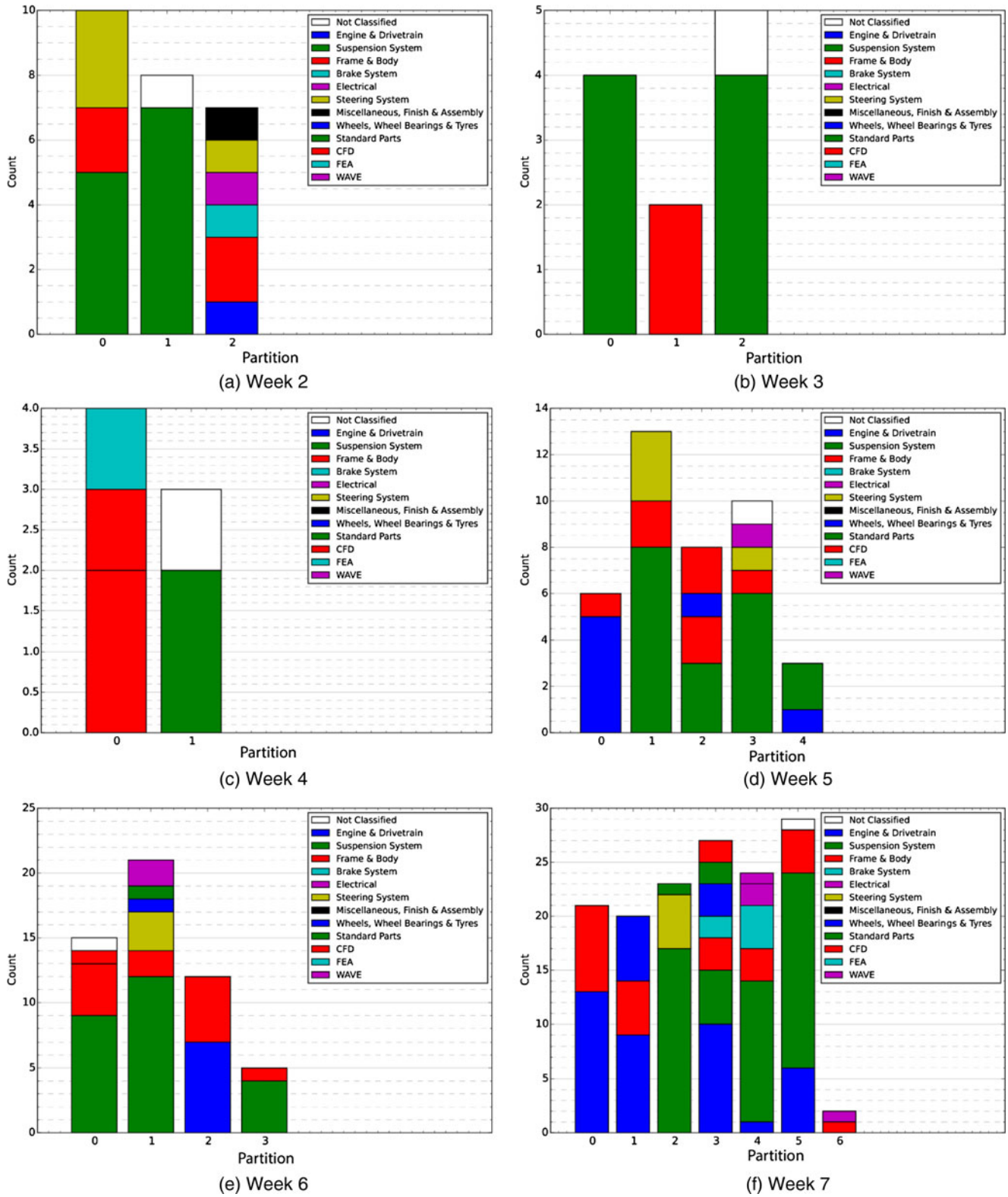


Fig. 18. Partitioning composition for Weeks 2 and 7 of the formula student project.

5. FUTURE WORK

In generating the process and interpreting the results, a number of avenues have been revealed for potential future work. The first avenue lies in applying this technique within an industrial context as a current limitation of this study is that the data set is one of a university-based project. Two challenges are foreseen. The first is that for the study reported in this paper, the team

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

Table 6. *Summary of CAD, CFD, FEA, and WAVE DSM*

CAD DSM
1. Areas of the product exist that are highly interdependent across all the subsystems that have been defined.
2. There a number of smaller partitions that consist of only one subsystem.
3. There are a number of partitions that arise from one subsystem and this may indicate that the subsystem can be usefully further decomposed.
4. The partitioning has identified distinct subsystem dependencies.
Product Model DSM
1. Each model type has partitions that solely consist of their respective models.
2. The existence of a single WAVE model partition, which is unrelated to the rest of the other modeling types
3. Identification of dependencies between areas of the CAD model and the FEA and CFD models where this analysis highlights that a change in any of these models may require further FEA and CFD analysis to be performed
4. A single partition consisting of CAD, CFD, and FEA models
Dynamic Product Model DSM
1. Partitions in Week 2 consist of models from all subsystems in the product that may indicate the generation of the initial layout of the vehicle
2. A change in the mode of working in Weeks 3 and 4 indicating concentrated work on areas of the vehicle
3. Identification of areas in the product architecture that require multidisciplinary collaborative work
4. A change in the mode of working that indicates a high-level of integrative working between the various subsystems of the vehicle

was colocated, and thus, the co-occurrence of work was performed within the same time zone. In contrast, large engineering projects may potentially have many concurrent work packages that are being worked upon in a distributed manner across multiple time zones. The second concerns the time it takes for changes to occur within various product models. For example, small CAD changes can take within the region of 2 min while CFD analyses could take days to run and complete. Thus, CAD models are open to more co-occurrence events than CFD, which the analysis has yet to adjust for.

Although there remain a few limitations, it is also argued that the analysis in its current form could support small to medium enterprises where these challenges may not be manifest and the cost of implementing PLM systems is too great. In addition, it would be interesting to compare this process alongside existing DSMs practices within engineering companies as well as their change management processes.

The second avenue of future research lies in the delivery of the results. As discussed previously in Section 4.1, the DSMs generated by this technique are much larger than more traditional DSMs, where the number of elements are typically in the hundreds, while this technique could produce DSMs with thousands of elements. Therefore, novel methods in interacting and navigating these DSMs to support engineers in their understanding of the multitude of dependencies that have been identified would be interesting to explore further.

The third is in the assessment of the impact of providing this information into Formula Student projects in future years. An interesting research question is to understand how the results of this analysis could be used to inform and support future engineering projects. In addition, expert evaluation of the DSMs generated through this process could be used to ascertain whether the dependency is functional, structural, or procedural. With this additional information, future work could seek to automatically determine and further verify the type of dependency alongside the existence of a dependency as the engineering project evolves. The Formula Student projects also provides a consistent project and engineering process where DSMs can be generated year on year, and it would be interesting to investigate the consistency of the DSM from team to team to explore how teams performing the same task impact the DSM.

The fourth and final avenue concerns whether the type and locality of the dependency can be uncovered through the analysis of the changes in content between product models. In addition, the dynamic DSM analysis has focused on the time slices of engineering activity, and analysis of the cumulative activity may provide additional insights in terms of the evolution of the dependency structure as well as the engineering process that has been followed.

6. CONCLUSION

Managing component interactions and dependencies remains a core and fundamental element of engineering and is faced by engineers on an almost daily basis. Being able to detect and monitor these interactions and dependencies continues to be a challenging area for engineering project management, and one that is set to become even more challenging as the number of components, component interactions, and component dependencies of products is continually increasing.

This paper has sought to complement existing techniques by extending the well-established DSM method through the creation of a process that automatically generates DSMs from the evolution of product models. This process has been applied to the evolution of the product models associated with a Formula Student project in order to assess the validity and addition insights that could be brought by an automatic DSM process.

Three DSMs were generated from the Formula Student data set, a CAD DSM, a product model DSM, and a dynamic product model DSM. The results reveal that the process is able to produce a DSM that is representative of the vehicle when compared to the existing subsystem definition and is comparable to DSMs generated by traditional methods. In addition, areas of high dependency and isolated subsystems of the vehicle have been identified, and this information could be used to support resource allocation and change management within the project. Finally, the key affordance offered by the process is the ability to provide dynamic DSMs in real time to an engineering project. This would be previously impractical using traditional DSM generation practices, and the results have shown that they can indicate changes in the

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391

mode of working within the project and provide insights into how dependencies between subsystems have emerged.

ACKNOWLEDGMENTS

The work reported in this paper was undertaken as part of the Language of Collaborative Manufacturing Project at the University of Bath and the University of Bristol, which is funded by Engineering and Physical Sciences Research Council (EPSRC) Grant EP/K014196/2.

REFERENCES

ANSYS. (2015). *Ansys cfx*. Accessed at http://www.ansys.com/Products/Simulation+Technology/Fluid+Dynamics/Fluid+Dynamics+Products/ANSYS+CFX

Blondel, V.D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment 2008(10)*, P10008.

Briggs, D. (2012). Establish digital product development (dpd) low end viewer (lev) and archival standard for 787 project. *Proc. Collaboration & Interoperability Congr*. (CIC), Denver, CO, May 21–23.

Browning, T.R. (2009). Using the design structure matrix to design program organizations. *Handbook of Systems Engineering and Management* (Sage, A., & Rouse, W., Eds.), 2nd ed., pp. 1401–1424. Hoboken, NJ: Wiley.

Bruce, C. (2015). *Toyota recalls 625k Prius models for faulty hybrid software*. Accessed at http://www.autoblog.com/2015/07/15/toyota-recalls-625k-prius-faulty-hybrid-software/

Calleam. (2011). *Airbus—a380*. Accessed at http://calleam.com/WTPF/?p=4700

Chen, S.-J.G., & Huang, E. (2007). A systematic approach for supply chain improvement using design structure matrix. *Journal of Intelligent Manufacturing 18(2)*, 285–299.

Eppinger, S.D. (1997). A planning method for integration of large-scale engineering systems. *Proc. Int. Conf. Engineering Design*, pp. 199–204, Tampere, Finland.

Eppinger, S.D., & Browning, T.R. (2012). *Design Structure Matrix Methods and Applications*. Cambridge, MA: MIT Press.

Fixson, S.K. (2005). Product architecture assessment: a tool to link product, process, and supply chain design decisions. *Journal of Operations Management 23(3)*, 345–369.

Gorbea, C., Spielmannleitner, T., Lindemann, U., & Fricke, E. (2008). Analysis of hybrid vehicle architectures using multiple domain matrices. *DSM 2008: Proc. 10th Int. DSM Conf*., pp. 375–387, Stockholm, November 11–12.

Hagberg, A.A., Schult, D.A., & Swart, P.J. (2008). Exploring network structure, dynamics, and function using NetworkX. *Proc. 7th Python in Science Conf. (SciPy2008)*, pp. 11–15. Pasadena, CA, August 11–15.

Hamraz, B., & Clarkson, J. (2015). Industrial evaluation of fbs linkage a method to support engineering change management. *Journal of Engineering Design 26(1–3)*, 24–47.

Jarratt, T.A.W., Eckert, C.M., Caldwell, N.H.M., &, P.J. (2011). Engineering change: an overview and perspective on the literature. *Research in Engineering Design 22(2)*, 103–124.

MacCormack, A., Rusnak, J., & Baldwin, C.Y. (2006). Exploring the structure of complex software designs: an empirical study of open source and proprietary code. *Management Science 52(7)*, 1015–1030.

Maurer, M.S. (2007). *Structural awareness in complex product design*. PhD Thesis, University of Munich.

Newman, M.E.J. (2004). Analysis of weighted networks. *Physical Review E 70(5)*, 056131.

Newman, M.E.J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences 103(23)*, 8577–8582.

Pimmler, T.U., & Eppinger, S.D. (1994). *Integration Analysis of Product Decompositions*. Cambridge, MA: MIT Press.

Pujol, J.M., Erramilli, V., & Rodriguez, P. (2009). *Divide and conquer: partitioning online social networks*. Accessed at http://arxiv.org/abs/0905.4918

Raspberry Pi Foundation. (2015). *Raspberry pi—home page*. Accessed at http://www.raspberrypi.org

Ricardo. (2015). *Wave—1d engine gas dynamics, 2015*. Accessed at http://www.ricardo.com/en-GB/What-we-do/Software/Products/WAVE/

Senescu, R.R., Head, A.W., Steinert, M., & Fischer, M.A. (2012). Generating a network of information dependencies automatically. *Proc.14th Int. Dependency and Structure Modelling Conf*., DSM12, pp. 139–152, Stanford, CA, September 12–13.

Siemens. (2015). *Nx: Siemens plm software*. Accessed at http://www.plm.automation.siemens.com/en_us/products/nx/

Sosa, M.E., Eppinger, S.E., & Rowles, C.M. (2003). Identifying modular and integrative systems and their impact on design team interactions. *Journal of Mechanical Design 125(2)*, 240–252.

Sosa, M.E., Eppinger, S.D., & Rowles, C.M. (2007). A network approach to define modularity of components in complex products. *Journal of Mechanical Design 129(11)*, 1118–1129.

Steward, D.V. (1981). The design structure system: a method for managing the design of complex systems. *IEEE Transactions on Engineering Management 28(3)*, 71–74.

Van Beek, T.J., Erden, M.S., & Tomiyama, T. (2010). Modular design of mechatronic systems with function modelling. *Mechatronics 20(8)*, 850–863.

James Gopsill is a Senior Research and Teaching Associate in the Department of Mechanical Engineering at the University of Bristol. As a member of the Design & Manufacturing Futures Lab, his research interests cover the application of artificial intelligence, machine learning, and big data analytics to support engineering design and knowledge management. In addition, he has an interest in rapid prototyping technologies and the development of tools to support their application within the engineering design process. He has published more than 25 journal and conference papers and has received two outstanding contribution awards.

Chris Snider is a Senior Research and Teaching Associate in the Department of Mechanical Engineering at the University of Bristol. Working within the engineering design and manufacture group and extensively with industry, his research interests cover development of engineering design processes, design support tools and methods, design thinking and behavior, engineering informatics, virtual and physical prototyping, and engineering management. In particular, his research focuses on monitoring and analysis of engineering processes throughout their life cycle, as well as effective design and development within constrained design situations. He has published more than 25 journal and conference papers and has won three international best paper awards.

Chris McMahon is a Professor of engineering design in the Department of Mechanical Engineering at the University of Bristol, a post he has held since September 2012. He previously worked at the University from 1984 to 2002. From 2002 to 2012, he worked at the University of Bath as Reader and then Professor and Director of its Innovative Design and Manufacturing Research Centre. Prior to 1984 he was a production and design engineer in the railway and automotive industries. His research interests are in engineering design, especially concerning the application of computers to the management of information and uncertainty in design, design automation, product life cycle management, design education, and design for sustainability, areas in which he has published over 250 refereed papers, a textbook, and a number of edited books. Professor McMahon is a Chartered Engineer, Fellow of the Institution of Mechanical Engineers (UK), and a founder

Downloaded from http://www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http://www.cambridge.org/core/terms. http://dx.doi.org/10.1017/S0890060416000391

member of the Design Society, for which he was President from 2010 to 2013. He is an active member of the scientific committees of various international journals and conferences.

Ben Hicks is a Chartered Engineering and Professor of mechanical engineering at the University of Bristol. He is Head of Engineering Systems and Design and leads the Design and Manufacturing Futures Lab. His expertise spans machine and manufacturing systems design, including self-replicating machines, modeling machine–material interaction, virtual prototyping, and engineering informatics. Dr. Hicks has published over 190 journal and conference articles, is a past member of EPSRC's Strategic Advisory Team, and recipient of the IMechE Water Arbitration Prize for best original paper.

Downloaded from http:/www.cambridge.org/core. University of Bristol Library, on 05 Oct 2016 at 08:42:38, subject to the Cambridge Core terms of use, available at http:/www.cambridge.org/core/terms.
http://dx.doi.org/10.1017/S0890060416000391