Proceedings of the 19th CIRP Design Conference – Competitive Design, Cranfield University, 30-31 March 2009, pp86

# Structured Design Automation

M.J.L. van Tooren[1], S.W.G. van der Elst[1], B. Vermeulen[2]
[1] Delft University of Technology, Kluyverweg 1, 2629 HS Delft, The Netherlands
[2] Stork Fokker AESP, Industrieweg 4, 3351 LB Papendrecht, The Netherlands
M.J.L.vanTooren@tudelft.nl, S.W.G.vanderElst@tudelft.nl, Brent.Vermeulen@stork.com

**Abstract**
The next stage in product development process evolution should be the automation of labour intensive repetitive steps. The design automation strategy should follow the trend of the supply chain management approach and deliver a flexible framework, allowing local specification and adjustment The proposed Adjoined Design Automation Process Trajectory (ADAPT) and associated tools provide such a framework. It covers the process from fuzzy front-end to the evaluation of automated processes. Considerable lead time and cost savings are shown for electrical component design and the feasibility of a Domain Specific Language approach is an important step towards acceptance of design automation.

**Keywords**:
Design automation, knowledge technologies, knowledge application, value stream mapping

## 1 INTRODUCTION

The value chain framework of Michael Porter [1], the Resource Based View developed by, amongst others, Wernerfelt and Barney [2], [3] and the Knowledge Management Framework described by Collison and Parcell, [4] provide us with a global and local view on what to organise a firm for. Of course knowing what to do leaves us with the question how to do it. In this paper a framework is proposed to use structured design automation in an engineering design environment to achieve better use of resources and implement true engineering knowledge management to improve the firms competitive position. The proposed framework approaches engineering knowledge as a resource which can and should be partially addressed as a tangible asset. It shows that the development of integrated design tools has blurred our view on the functionality of its components. Only through a proper understanding of this functionality and associated technology one can (re-)discover which part of the engineering supply chain is covered by these tools, how it is covered and finally if and why it should be covered in this way. The framework offers an alternative engineering knowledge management tool chain which is transparent and reconfigurable in a lean way to fit the actual need of the company and to allow insertion of one's own proprietary knowledge.

The framework borrows elements from the lean manufacturing approach. However, although lean manufacturing can be used as a guideline to organise the structured design automation process it needs supporting philosophies and tools to achieve the required functionality.

Starting point of the proposed approach is the assumption that it will become more and more important to apply and extent the abilities of current information technology towards knowledge application technologies which can take over many of the repetitive activities currently done by scarce and expensive human intellectual capital. Many of the necessary components for this transformation are available on the market but lack coherence and acceptance by the industry. This can be explained from the large gap between knowledge management as understood and implemented by managers and knowledge management as seen by knowledge engineering specialists from the IT-world and a small group of believers in the engineering world. Both aim for the same goals but are too far apart through lack of mutual understanding. Furthermore, the wide range of available tools and the complexity of understanding their functions and benefits in a local engineering environment lead to many disappointments and long lasting suspicion. It is our aim to narrow the gap and contribute to the further development of the methodology to implement knowledge engineering in industry.

## 2 THE ADAPT PROCESS

The proposed Adjoined Design Automation Process Trajectory (ADAPT) is an attempt to define a business model providing a generic framework for engineering knowledge management with a clear coupling to knowledge technologies. The ADAPT approach should ensure the application, implementation and added value of these knowledge technologies and help engineering design communities to continuously evolve by enabling re-use and extension of their knowledge base.

The implementation of knowledge technologies in an industrial environment in a controlled and useful way requires an integrated, programmatic and transparent approach. The ADAPT process shown in Figure 1 is an attempt to frame existing methodologies and tools in a coherent way.

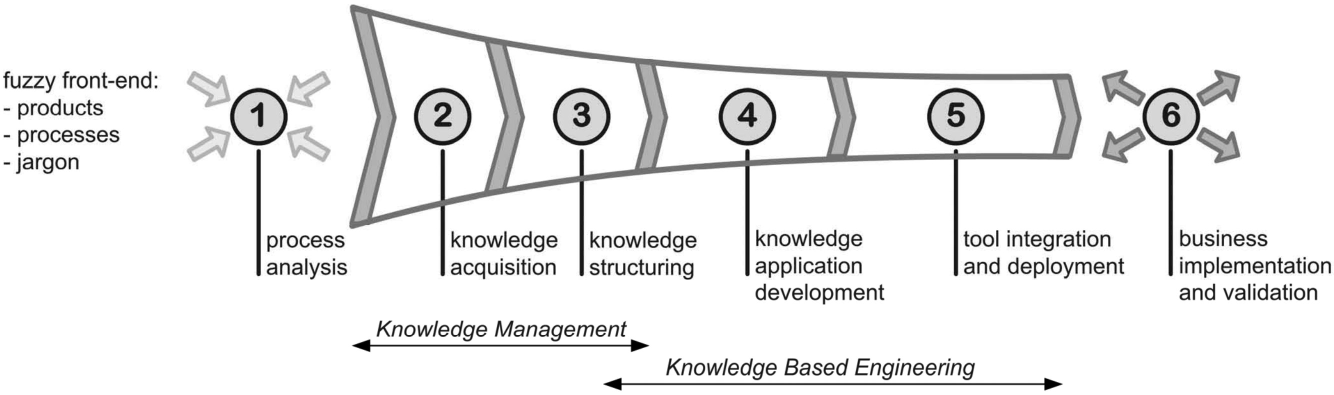


Figure 1: The ADAPT process

## 2.1 Process Analysis

The first phase of the ADAPT process concerns an in-depth analysis of the engineering processes performed by the engineers, the products/services they work on and the language they use in describing these products and processes (indicated with the fuzzy front-end). This analysis shall identify process improvement opportunities by applying lean principles to the product development activities. The process analysis is mainly focused on knowledge-intensive activities and products belonging to a larger set or family, ensuring a sufficiently large applicability of the resulting knowledge application. The deliverable of the process analysis is a *value stream map* (see section 3) that shows the engineering processes and the data, information and knowledge flowing through these processes or generated by these processes. The map is used to identify waste of (scarce) engineering resources as well as opportunities to reduce this waste using knowledge technologies that enable design automation. For the categorisation one can use the seven types of waste that can be identified during product development, as presented in Table 1 [1].

Of these seven types of waste identifiable in most product development processes, design automation mainly addresses 'processing' and 'correction' waste. Using Knowledge Based Engineering (KBE) techniques, expert knowledge can be captured and reused to automate repetitive and non-creative engineering activities, thereby reducing product development time and cost.

The VSM is performed in close cooperation with project managers and principal stake holders to create awareness of the weaknesses and possibilities within the engineering process. This way, a natural demand can established, rather than a push model for the relatively unfamiliar knowledge technologies. Furthermore, the involvement of project managers in the process analysis phase is important for a second reason: they are key enablers of and responsible for a successful implementation. The project managers are able to allocate engineering resources and therefore stimulate

| Waste | What is it? | PD Examples |
|---|---|---|
| **Overproducing** | Producing more or earlier than the next process requires | Batching, unsynchronised concurrent tasks |
| **Waiting** | Waiting for input, information or decisions | Waiting for management approval or decisions, information distribution and data handling |
| **Conveyance** | Moving knowledge, information or data from place to place | Hands-off / excessive information distribution or data handling |
| **Processing** | Doing unnecessary processing on a task or an unnecessary task | Stop-and-go tasks, redundant tasks, reinvention, process variation – lack of stadardisation |
| **Inventory** | A build up of information or data that is not being used | Batching, system overutilisation, arrival variation |
| **Motion** | Excess motion or activity during task or execution | Long travel distances, redundant meetings / superficial reviews |
| **Correction** | Inspection to catch quality problems, post-fixing errors and changed input | External quality enforcement, correction and rework |

Table 1: Applying the Seven Wastes to product development [5]

the use of knowledge applications to assure business advantages, e.g. reduced cost or increased quality.

During the analysis of the engineering design process the flow of information, the transformation of information and the required and applied expert knowledge is monitored. The analysis focuses on four main characteristics:

- Required engineering resources
- Repetitiveness of engineering process within product family
- Nature and maturity of expert knowledge
- Key performance indicators related to the identified processes (cost, time, quality etc.)

The required engineering resources and the number of process cycles provide insight in the cost involved in different recurring processes in the non-recurring part of the development of a product family. It should also offer information about the longevity of the applied knowledge. The domain expert knowledge is assessed to determine its nature and maturity. When processes are highly frequent, time-intensive, clearly defined and not subject to change, knowledge technologies can enable automation.

During the process analysis, possible knowledge technology architectures and applications are examined. Furthermore a risk analysis is performed to identify the risks involved in the development of knowledge applications. Together the required investments, the expected benefits and the risk analysis should justify the implementation of KBE techniques. The analysis phase is concluded by a selection of engineering processes to be automated, the level of automation and a first draft of the architecture for the constellation of knowledge tools suitable in the context.

### 2.2 Knowledge acquisition

During the knowledge acquisition phase expert knowledge involved in the engineering process is identified, captured and structured. The knowledge acquisition phase forms the foundation for the subsequent phases of the ADAPT process. The knowledge acquisition phase has an iterative character and consists of identifying, capturing, structuring and validating the expert knowledge. The deliverable of the knowledge acquisition phase is a *knowledge base*, a digital repository containing a detailed description of knowledge concerned with the selected engineering process.

The quality and completeness of the captured knowledge largely determines the success rate of the development process hence the resulting design automation. To guarantee a successful result, the acquisition process is performed in close cooperation with the domain experts. The involvement of domain experts is vital to the project for two main reasons:

- Identification and dissemination of relevant knowledge
- Validation of quality and completeness of the captured knowledge

Using different knowledge acquisition techniques a conceptual model of the selected engineering process is constructed, providing an informal but detailed description of the activities. In order to maximise the ability for future reuse of the captured knowledge, it is recommended that the knowledge base hence the conceptual model is not catered to one specific implementation (in this case: the development of knowledge applications) and embeds a neutral structure oriented to the engineers. This enables the knowledge base to act as a general-purpose fundamental base for reuse of knowledge. Other purposes of knowledge reuse are: provide expertise and increase awareness to stakeholders within an organisation or reduce the risk of knowledge loss in domains where only a small number of experts hold vital knowledge.

To obtain a neutral structure the captured engineering knowledge is represented using natural language, terminology from the domain under consideration and pre-defined forms to structure the different knowledge elements. The conceptual model contains a process diagram focusing on the activities performed by the engineers and is oriented to the ‘input-behaviour-output’ perspective. It mainly contains procedural knowledge and therefore encompasses a comprehensive activity diagram or flow chart. Besides a detailed description of the engineering activities under consideration, the conceptual model of the knowledge base also contains a product centric hierarchical decomposition of the system (i.e. product/service) into subsystems and components. This product model is oriented to the ‘object-relation-object’ (triple) perspective and mainly contains conceptual knowledge. The conceptual model of the knowledge base will form the basis for the subsequent development of the application.

### 2.3 Knowledge structuring

The third phase focuses on modelling the captured knowledge. The captured engineering knowledge is analysed and (re)structured to suit the knowledge technologies selected for subsequent development when the knowledge base is not the end product. The deliverable is considered a *redesigned engineering process* and provides structure and lay-out for the knowledge application to be developed. It is also referred to as a specification model since it provides a more formal definition of the engineering knowledge oriented towards software platforms. The specification model is used to support the communication between knowledge engineers and software developers. Together with the expert-oriented conceptual model created during the knowledge acquisition phase it comprises the knowledge base. The specification model provides a structure for the software classes representing the different product and process elements and acts as a blueprint for the knowledge application. It consists of two layers.

First, the specification model provides an architecture lay-out describing the software framework environment. Following a functional decomposition the knowledge application is divided into several self-contained software tools to increase the reuse and the expressiveness of the related software code. The set of software tools provides full functionality to execute the engineering activity under consideration. Furthermore, the framework enables communication between the software tools through agents and provides a loosely coupled demand-driven structure for the application. Within the framework, each tool is considered an engineering service providing functionality to the framework, for example optimisation packages, data bases and analysis tools.

Second, the specification model contains a representation of the central KBE application: the model generator. The model generator is responsible for definition and instantiation of a specific product model and is able to generate discipline specific report files as input for analysis tools defined in the framework environment.

### 2.4 Knowledge application development

The fourth phase, knowledge application development, addresses the software development of the actual knowledge application, e.g. the architecture and its constituting tools (model generator, agents, optimisers, product data management, analysis tools etc) [6].

Due to the framework approach and the modular build-up of knowledge applications, the reuse of the different tools is ensured to a large extend. The tools composing the knowledge application are either already available (commercially of the shelf (COTS) or developed and applied during previous applications) or will need to be developed.

Developing knowledge applications using dedicated KBE platforms require the programming of the central model generator: defining (new) design options and constructing configurations within a product family. Exploiting dedicated KBE development platforms, for example Genworks' GDL or the former ICAD from KTI, an object-oriented and functional programming language is used to encode the knowledge. The engineering knowledge is stored into modular software objects, called High Level Primitives (HLP). The primitives represent different design options and can be created, tailored and assembled to define new product configurations. The object-oriented characteristic allows developers to resemble the decomposition of the product defined by the conceptual model using a network of classes.

Besides the conceptual knowledge, object-oriented programming also allows the incorporation of procedural knowledge using so-called facets: specific class attributes that contain procedures (methods and references) that are automatically invoked when the value of the slot is requested or changed during runtime. The specific procedures are derived from the rules in the activity diagram of the conceptual model.

The encoding of the primitives and software modules is considered an iterative process. During the development of the application, additional, undiscovered or changed knowledge might be identified and the associated models from the knowledge base need adjustment to ensure that they accurately represent the engineering activity as well as the structure and process of the application.

Using object-oriented and high-level programming languages, the resulting code volume is considered very low. Furthermore, programming languages with a high level of abstraction require lower entry-level programming skills.

### 2.5 Tool integration and deployment

The fifth phase addresses the integration of the software modules to form the knowledge based architecture and its components. It includes the development of communication interfaces and the distribution of the application itself. The deliverable is an *automated design application* based on the knowledge techniques offering engineering services. The architecture and tools shall support performance indicators to support their evaluation with respect to the key performance indicators identified in the first phase of the ADAPT process.

### 2.6 Business implementation

The last phase concerns the implementation of the knowledge application in the design process. Since the flow of information within a process will change when deploying knowledge technology applications, a process wide re-design is needed to prevent the occurrence of bottlenecks creating waste [7]. Configuration management and maintenance are conducted to ensure traceability of the knowledge rules invoked and reproducibility of the resulting solutions. Furthermore, an essential step in the implementation of knowledge technologies is to recognise that they imply an important change in the work of engineers. Therefore, more practical attributes to a successful implementation are support and training of end-users. Overall, five groups of key success factors for the implementation of KBE applications can be identified [8]:

**can**

- Provide training in the operation of the application
- Provide a useful and usable user manual
- Stimulate users to share best practices in using the application

**want**

- Focus on topics important to the business and engineers
- Communicate KBE vision, need for the business, results and experiences of users
- Evaluate the usefulness and usability of the application on a regular basis

**have**

- Plan the development of the application in terms of required resources and release date
- Make reservation in project planning to practice using the application
- Provide support during the lifetime of the application

**must**

- Convince management of possible payback in terms of lead-time and resources
- Have a well-respected engineer promote the use of the application

**measure**

- Application performance in relation to the identified key performance indicators

The last and very important aspect is the monitoring of the performance of the application using the information supplied by the system in relation to the key performance indicators identified. Adaptation, cancellation and expansion where required should be an integral part of the process.

## 3 FROM VALUE STREAM MAPPING TO STRUCTURED KNOWLEDGE

In the ADAPT framework Value Stream Mapping (VSM) is a key tool to gain insight into the local engineering processes. The graphical representation of the engineering activities as well as the flow of data, information and knowledge flowing through and generated by those activities helps the communication about and the understanding of the local engineering practice.

VSM originated in the manufacturing industry. Applying proper modifications to the original VSM, this tool can also be applied in order to improve product development processes [9]. Where the original VSM looks critically at the flow of material, the modified VSM looks at the transformation and generation of data, information and knowledge as a series of process steps interrupted by waste: consuming engineering resources without adding value for the customer.

By considering and mapping the current state of product development value streams and identifying waste, VSM defines a more efficient or lean future state while eliminating waste that interrupts a continuous and even flow of data, information and knowledge. The future state diagram provides the foundation for a future process and the subsequent action plan to implement it.

As opposed to serial value streams typical of manufacturing, typical product development processes consist of numerous interdependent value adding activities. This interwoven character makes it difficult to

| **Product development process** | **Traditional Manufacturing process** |
|---|---|
| Virtual data flow | Physical product flow |
| Weeks and months | Seconds through hours |
| Primarily knowledge intensive work | Physical manufacturing |
| Nonlinear and multidirectional flows | Linear and serial evolution |
| Large and diverse group of domain experts | Primarily manufacturing organisation |

Table 2: PDVSM versus VSM

define flow and identify forms of waste. The key to superior product development is to analyse the complex network of activities into definable 'work streams' or sets of subsequent process steps transforming input into output. The work streams will not only identify the waste of resources in between the diverse streams, they will also pinpoint waste interrupting the process steps within the individual streams. Numerous distinctions between traditional VSM and product development VSM (PDVSM) are represented in Table 2.

Within the ADAPT framework PDVSM is used to select the engineering practices that will benefit from automation. In addition it is the first step towards the knowledge acquisition and knowledge structuring phases which are leading to the knowledge base, a crucial product of the ADAPT process.

In this knowledge base we will have process maps to formalise the identified processes, trees to formalise products and product families and taxonomies to formalise the terminology used in the maps and trees.

The relation between the products and processes are formalised with ontologies (also named diagrams), Table 3. This way the fuzzy front-end, which is the not explicitly and consistently defined collective of local engineering activities and their objectives (the processes, products and language which define the local engineering practice) is transformed into a well defined body of knowledge suitable for further development into KBE applications or a Design and Engineering Engines (DEE) [10].

The re-use of the knowledge in multiple KBE applications or DEE's needs an additional step. Most of the designers and engineers are not willing to spend most of their time programming, even in a high level language as normally used in a KBE platform. Therefore a proper interface language is needed through which knowledge can be re-used. This will be discussed in the next section.

## 4 DOMAIN SPECIFIC LANGUAGES

In general products and services are designed through a synthesis of existing and new design options into known or new configurations. The associated design options and processes are described by both generic, domain and discipline specific terminology.

In order to encode all design options effectively and correctly into the knowledge application, the representation of the related classes and objects need to complete multiple objectives, also known as knowledge representation roles. Where the conceptual model enables the communication and visualisation between knowledge engineers and the domain experts, the specification model is used as means of communication for both human expression and computation (execution of activities by knowledge applications). Especially this latter category requires modifications and explicit specifications (hence the name specification model) to the underlying language in order to suite the correct interpretation by virtual machines:

- The specification model should follow programming language syntax
- The model should provide a visual representation for ease of construction
- Rules governing the value of class properties are defined
- Class and object descriptions should be intelligible for humans (experts, knowledge engineers and software developers)

To alleviate the required effort involved in the development of knowledge applications a Domain Specific modelling Language (DSL) is developed, enabling the symbolic representation of products or systems of the problem domain while satisfying the abovementioned requirements.

With the help of generic language concepts like the Unified Modelling Language (UML) a DSL is carefully defined to enable the representation of conceptual classes of the physical world to be meaningful to both humans and intelligent systems. The DSL is considered a visual dictionary of noteworthy abstractions, domain vocabulary and knowledge content of the domain under consideration [11]. In addition to an ontology defining the types of elements that exist and their relations within a particular domain, a DSL should contain not only class types but also instances of objects and rules in order to construct new specification models. These knowledge elements are considered the building blocks for the specification model, like words are to natural languages.

During the knowledge acquisition and structuring phases it is important to get a thorough and formal description of the different knowledge elements. The structuring of the objects and rules applied during the design processes can benefit from a standardised categorisation. An example of a general categorisation for design rules is shown in Table 4.

During knowledge acquisition a systematic discovery of the rules applied in each of these categories is performed. The subsequent knowledge structuring should prepare for the DSL as the interface towards the formalised knowledge and the re-use of this knowledge (e.g. when

| **Fuzzy Front-end** | **Knowledge Base** | | **Knowledge Re-use** |
|---|---|---|---|
| Processes *(engineering practices and rules)* | Process maps | Diagrams *(built from concepts and relations)* | KBE applications *(object oriented)*<br>DEE's<br>High Level Primitives<br>Domain Specific Languages |
| Products *(design options)* | Trees | | |
| Jargon *(discipline specific language)* | Taxonomy | Ontology | |

Table 3: Relation between product and process knowledge during different phases

| | Product related | Process related |
|---|---|---|
| **Internal** | Engineering<br>Design (functional, aesthetics etc)<br>Management | Engineering<br>Tool operation (including work-arounds)<br>Tool interfacing<br>Management |
| **External** | Mathematics<br>Physics<br>Engineering<br>Design<br>Law<br>Market | Mathematics<br>Physics<br>Engineering<br>Design<br>Tool operation (including work-arounds)<br>Tool interfacing |

Table 4: Origins of rules in a design organisation: internal and external related to organisation boundaries

building a KBE application or a DEE).

When applied to the knowledge base using knowledge management tools, the DSL provides domain experts, knowledge engineers and IT specialist a means of communication to visualise, structure and validate their conceptual ideas. The DSL can be applied to define new product configurations and variations within the product family. Since the syntax of the DSL suites object-oriented programming languages it enables the application of the same abstractions and vocabulary to define the different software classes underlying the knowledge applications. Therefore it an be stated that the DSL increases the insight in the knowledge application and the coherence between the different knowledge application technologies. Combined with dynamic source code generation, the knowledge base can be applied to structure new product configurations using existing or new design options and automatically generate the software code representing the associated generative product model for the knowledge application [12].

## 5 KNOWLEDGE APPLICATION CASE STUDY

Following the discussion on the ADAPT process an example of a knowledge application will be addressed. The application has been developed applying the automation process trajectory applying the DSL.

### 5.1 Wiring Harness Design Application

Electric aircraft wiring harnesses can be comprised of hundreds of cables and ten thousands of wires, providing connectivity between all the mission and vehicle systems ensuring sufficient redundancy and reliability. Electrical wiring design is often performed in parallel with structural design. Consequently, the wiring harness design is subject to changes in the aircraft structure that occur with subsequent design iterations, requiring time consuming rework for any harnesses affected. The routing for all wires is determined manually and strongly dependent on personal knowledge and experience. Besides, the electric wiring design is governed by numerous regulatory and functional design rules. The repetitive, time consuming and rule-based nature makes aircraft wiring design a key opportunity to develop knowledge applications

Figure 2: Connectors applied at a wiring harness production break

The development of the application is performed in close corporation with Stork Fokker Elmo, a main international player on the aircraft electric wiring market, regarding both design and manufacturing.

*Process Analysis*

For the wiring harness design process, one of the key opportunities resulting from the initial VSM involves the pin assignment process. It involves the assignment of electric signals at production breaks, where connectors connect the different wiring harnesses (Figure 2). Each wiring harness connector can include up to 150 slots, called pins, to accommodate a signal. The pins can vary in size, as do the signals to be assigned.

For each production break the signals are assigned to a pin and associated connector, one by one consecutively. This process of pin assignment is highly repetitive and time-consuming due to several reasons:

- Separation of signals across multiple wiring harness segments or cables is enforced by numerous opposing design rules and regulations, for example redundancy of flight controls, electromagnetic compatibility or heat dissipation of power cables.
- The increasingly vast quantity of signals to be assigned (‘processing’ waste).
- Rework caused by changes in the input data, for example governed by design iterations for the aircraft structural design (‘correction’ waste).

For the development of the application, the dedicated knowledge system GDL from Genworks is selected. GDL is a new generation knowledge system that combines the power and flexibility of the former ICAD system with novel web technologies. Its object-oriented programming language is based on the standard ANSI Common Lisp and allows the definition of generative product models. Furthermore, ILOG CPLEX is selected to act as search engine: the COTS linear programming optimisation tool will analyse models provided by the generative product model and drive the search process to a feasible and optimal design.

*Knowledge Acquisition*

The iterative knowledge acquisition process of capturing, structuring and validating the expert knowledge is supported by Epistemics’ PCPACK a software package supporting the process of acquiring, storing and representing knowledge. A separate ontology is developed, specifically built to suit the wiring harness domain. A comprehensive description of the involved engineering activities is defined, together with a conceptual product decomposition of the system. Furthermore the design rules and best practices guiding the activities are captured, many of which are opposing. Some examples of applicable design rules are:

- The ratio of occupied pins over available pins has a settable maximum (design requirement)
- Signal types should be grouped among connectors to fulfil separation requirements (authority regulations)

- Per connector, signals subtypes should be centred and grouped together (manufacturing requirements)

The informal model functions as a detailed engineering handbook decreasing the knowledge entry level required to perform the pin assignment processes.

*Knowledge structuring*

During knowledge structuring, a large amount of specific domain knowledge is crunched into a more formal model, reflecting deep insight into the resulting knowledge application. The formal model of the knowledge base provides an architecture lay-out describing the software framework environment for the application. To that purpose, it takes into account the roles and capabilities of the GDL and CPLEX software tools.

Although inheriting the functionality of the original process, the redesigned process might consist of entirely different sub-processes and activities. For example, when the objective is to assign 70 signals across 90 available pins fulfilling all requirements and incorporating best practices, a human engineer will require a vast amount of time to explore most if not all possibilities. Applying the CPLEX optimisation software results in a much more efficient exploration of the solution space, solving the problem concurrently for all signals thus increasing the reduction in recurring process time.

The object classes that constitute the product decomposition represent the generative product model which will be programmed during the subsequent development phase. The object classes will encompass the design rules and best practices, suiting the object-oriented approach of the GDL knowledge system. Together, these object class definitions form the DSL representing functional building block called High-Level Primitive (HLP) [10]. The HLPs can be tailored and assembled enabling engineers to define new product configuration and new design options. For instance new connector types or pins with alternative gauges can be defined easily.

*Knowledge application development*

Once the knowledge structuring of the expert knowledge is finished and the architecture for the application fully defined, the software modules constituting the application are developed. For the application supporting the pin assignment process, the application will consist of two modules:

- A generative product model called a Multi-Model Generator (MMG) developed using the GDL knowledge system [10].
- A converger and evaluator, represented by the linear programming optimiser CPLEX.

Since CPLEX is a COTS tool, the development focuses on the MMG. The product decomposition as defined in the formal model of the knowledge base represents the structure of the software classes. The modular building blocks or HLPs are programmed using the object-oriented programming language. Each object class or HLP defined in the formal model has an equivalent software class. It becomes apparent that the formal model is a diagrammatic representation of the software structure and source code: it makes the code more expressive and clarifies the processes and rules invoked by the knowledge application.

Besides the HLPs, the MMG consists of elements, called Capability Modules (CM) capable of extracting certain discipline specific 'views' in order to facilitate the analysis tools. In this particular case, the only discipline involved is mathematics. The related CM extracts a mathematical model of the connectors composing the production break, defining the supply of pins as well as the demand generated by the signals per separation code. The CM defines the objective function (minimise the number of pins occupied by a signal) and generates all constraints derived from the applicable design rules. The output is a report file, specifying a linear programming problem modelled after the instantiated pin assignment problem. This problem can be analysed and solved by Cplex efficiently.

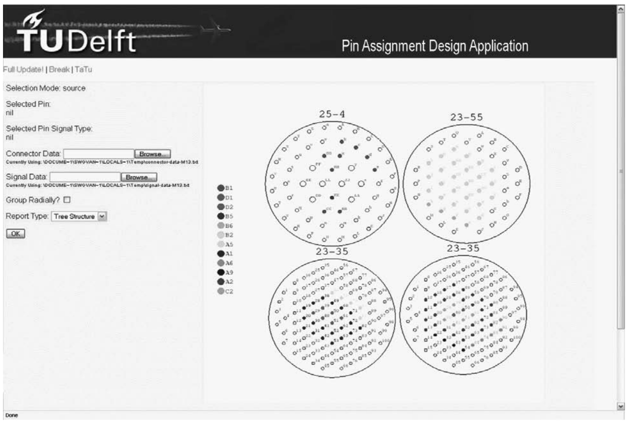


Figure 3: Graphical User Interface for the pin assignment application

The development of the KBE software modules is performed iteratively and can be considered domain driven. After each iteration cycle, the formal model is adjusted to ensure the model accurately represents the structure and process of the knowledge application.

*Tool integration and deployment*

To empower the automation of repetitive tasks for of the pin assignment problem the framework concept of the Design and Engineering Engine is applied [10]. The DEE integrates the self-contained software modules and provides communication between the modules through the application of software agents [13].

Considering the pin assignment problem, the resulting framework functions as a stand-alone knowledge application and has not yet been connected to the other engineering corporate software packages. The MMG and associated agent have been deployed on-site at Fokker Elmo, whereas the CPLEX optimiser is executed remotely as engineering service, on request.

A Graphical User Interface (GUI) is designed to enable interaction with engineers. The GUI allows the engineers to specify the input data (problem description) and provides identified best practices as execution options, such as grouping of signals. The GUI also enables the engineers to manually adapt solutions as suggested by the application through incorporated selection functionality and provides different types of output files to accommodate manufacturing as well as design engineers. The GUI is presented in Figure 3 and illustrates the front view of the set of connectors composing the production break of the wiring harness. The different signal types are colour-coded by separation code, to enable easy verification by the engineers.

*Business implementation and validation*

The implementation of the pin assignment application into the business environment is not yet performed and scheduled for next year.

## 6 CONCLUSIONS

The ADAPT process presented offers a structured approach towards a practical implementation of knowledge management and Knowledge Based Engineering in an engineering environment. It is based on a sequence of well defined technologies, supplemented with additional tools to complete the chain. The Value Stream Mapping technique adapted for a product development environment is well suited to analyse the fuzzy front end of local engineering communities and prepares well for subsequent knowledge acquisition and analysis. With the use of Domain Specific Languages the UML concept is extended to form an interface to the re-users of the formalised knowledge. A case study showed that the methodology works and can lead to structured waste elimination and cost saving.

## 7 RECOMMENDATIONS

The case study presented in this paper addressed mainly the elimination of 'processing' and 'correction' waste associated to individual process steps belonging to the value-adding or core business process. However, ADAPT could also support the elimination of other types of waste. A large part of the waste in the overall business process is likely to occur in between different process steps and is considered to be greater than the waste within single process steps. Typical types of waste occurring in between process steps are 'waiting', 'overproducing' and 'inventory'.

Knowledge technologies also enable the partial elimination of other types of waste. Integrating multiple design automation and Commercial Of-The-Shelf (COTS) tools into a framework structure, communication and data handling ('conveyance' waste) can be controlled in a demand-driven approach reducing 'waiting' 'inventory' and 'overproducing' waste.

## 8 ACKNOWLEDGMENTS

The authors would like to express their gratitude to Genworks, Stork Fokker AESP and Fokker Elmo for their support and contributions.

## 9 REFERENCES

[1] Porter, M., 1985, Competitive Advantage: Creating and Sustaining Superior Performance, Free Press, New York, NY.

[2] Wernerfelt, B., 1984, The Resource-Based View of the Firm, Strategic Management Journal, vol. 5, No. 2: 171-180.

[3] Barney, J., 1991, Firm Resources and Sustained Competitive Advantage, Journal of Management, vol. 17, No. 1: 99-120.

[4] Collison, C., Parcell, G., 2001, Learning to Fly: Practical Knowledge Management from Leading and Learning Organisations, Capstone Publishing Ltd., Chichester, United Kingdom.

[5] Morgan, J., Liker, J., 2006, The Toyota Product Development System, Productivity Press, New York, NY.

[6] Van der Elst, S., Van Tooren, M., Vermeulen, B., Emberey, C., Milton, N., 2008, Application of a Knowledge Based Design Methodology to Support Fuselage Panel Design, Aircraft Structural Design Conference, Liverpool, UK.

[7] Vermeulen, B., 2007, Knowledge Based Method for solving complexity in design problems, Delft University of Technology, Delft, The Netherlands.

[8] Van der Spek, R., Kelleher, M., Knowledge management, reducing the costs of ignorance, *www.dnv.com/services/consulting/knowledge_management/Publications*

[9] Morgan, J., 2002, High Performance Product Development; a Systems Approach to a Lean Product Development Process, The University of Michigan, Ann Arbor, MI.

[10] La Rocca, G., van Tooren, M., Enabling Distributed Multi-disciplinary Design of Complex Products: a Knowledge Based Engineering Approach, J. Design Research, vol. 5, No. 3: 333-352.

[11] Evans, E., 2004, Domain-Driven Design, Addison Wesley, Boston, MA.

[12] Van der Elst, S., Van Tooren, M., 2008, Domain Specific Modelling Languages to Support Model-Driven Engineering of Aircraft Systems, 26th Congress of the International Council of the Aeronautical Sciences, Anchorage, AK.

[13] Berends, J., van Tooren, M., Schut, E., 2008, Design and Implementation of a New Generation Mulit-Agent Task Environment Framework. 49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Schaumburg, IL.