

Knowledge Based Engineering across Product Realization

Version 1.0
June, 2011

Copyright Notice

© Geometric Limited. All rights reserved.

No part of this document (whether in hardcopy or electronic form) may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, to any third party without the written permission of Geometric Limited. Geometric Limited reserves the right to change the information contained in this document without prior notice.

The names or trademarks or registered trademarks used in this document are the sole property of the respective owners and are governed/ protected by the relevant trademark and copyright laws.

This document is provided by Geometric Limited for informational purposes only, without representation or warranty of any kind, and Geometric Limited shall not be liable for errors or omissions with respect to the document. The information contained herein is provided on an “AS-IS” basis and to the maximum extent permitted by applicable law, Geometric Limited hereby disclaims all other warranties and conditions, either express, implied or statutory, including but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, and of lack of negligence, all with regard to the document.

THERE IS NO WARRANTY OR CONDITION OF NON-INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO THE DOCUMENT. IN NO EVENT WILL GEOMETRIC LIMITED BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY WAY OUT OF THIS DOCUMENT, WHETHER OR NOT SUCH PARTY HAD ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

Confidentiality Notice

This document is disclosed only to the recipient pursuant to a confidentiality relationship under which the recipient has confidentiality obligations defined herein after. This document constitutes confidential information and contains proprietary information belonging to Geometric Limited, and the recipient, by its receipt of this document, acknowledges the same. The recipient shall use the confidential information only for the purpose defined above for which this document is supplied. The recipient must obtain Geometric Limited’s written consent before the recipient discloses any information on the contents or subject matter of this document or part thereof to any third party which may include an individual, firm or company or an employee or employees of such a firm or company. The recipient acknowledges its obligation to comply with the provisions of this confidentiality notice.

Contents

Abstract	4
Introduction	4
Scope	4
KBE: Past and present	5
KBE implementation roadmap.....	7
KBE architecture.....	11
Conclusion	13
References	13
About the Author	13
About Geometric	14

Abstract

Today the business ecosystems are transforming from an industrialization based economy to knowledge based economy. In knowledge based economy, you must continuously accumulate and use knowledge to create a sustainable competitive advantage.

Knowledge based engineering (KBE) is a technique that provides ways to capture and deploy knowledge across the complete product realization value chain. A proper implementation roadmap and maximum control at hands of engineers are the key factors for successfully leveraging KBE.

This paper explains the levels of KBE and then defines a roadmap for its implementation. It also proposes architectural options for implementing KBE within the cost and timeline, along with ways of offering flexibility for the engineers to enhance the knowledge.

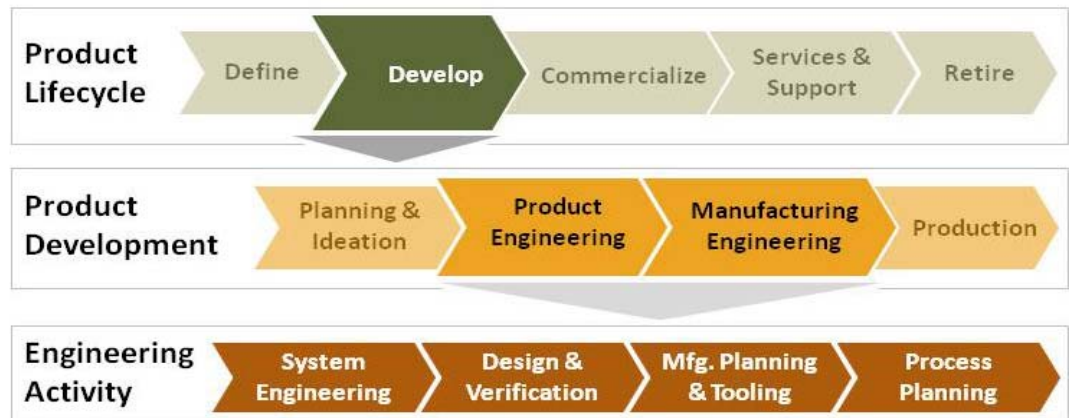
Introduction

Knowledge based economies recognize knowledge as a key driving factor of productivity and economic growth. While knowledge has long played an important role in the economic growth, modern manufacturing organizations are now exploring ways to accumulate and incorporate learnings and knowledge right from the product development and manufacturing stage for enhancing current or new products. In that sense, Knowledge Based Engineering (KBE) can be seen as a tool for capturing and reusing knowledge during the product development. It focuses on how to capture and formalize knowledge so as to enable shorter product development time.

This whitepaper introduces role of KBE in the knowledge based economy. It describes a roadmap for KBE implementation to ensure sustainable returns on KBE. Further, this paper explains realization of the roadmap on modern design tools, with full control in the hands of engineers to maintain and enhance the knowledge.

Scope

Engineering functions define component specifications and manufacturing process in product development. In fact, 'Product and Manufacturing Engineering' contributes in excess of 80% towards the product development costs. Additionally, the engineering activity under product and manufacturing engineering has maximum engineering knowledge accumulated. Hence, this paper concentrates on these activities as our scope. These activities predominantly use the CAX (CAD, CAE & CAM, CAPP) tools.



KBE aspects change depending upon the perspective of IT or engineering. Depending on this perspective, KBE involves wider skill set from modeling to programming and even artificial intelligence. This paper focuses on KBE that can be managed predominantly with the engineering skills. It focuses on modeling based KBE, supported by scripting, automation, and spreadsheet based configurations.

KBE: Past and present

Initially, KBE in product development was intended to reduce the repetitive design tasks. Gradually the focus shifted to data management and collaboration systems. The shift could be attributed to the changes in priorities and limitations of KBE implementation.

Today, it is important to expand the capabilities of KBE by leveraging new tools and by adopting different approach for implementation. This section discusses issues that led to sidelining of KBE, followed by current drivers, enablers and key challenges.

Issues in the past

In the past, KBE was implemented on specialized systems as a point solution. Major drawbacks with this approach were:

- **Localized implementations:** KBE implementations captured knowledge at a particular stage of product development process and deployed it back to that stage itself. However, such localized implementation could not register a major impact on the complete product development process.
- **Unrealistic expectations:** Results from localized KBE implementations built unrealistic expectations. While every KBE investment produced results, those were not in line with the hype created by the top-end achievements.
- **High cost:** KBE is economically demanding to implement as it required specialized tools (viz. ICAD, Intent, etc), high-end hardware and specialized skills. This combination increased the

cost of ownership and necessitated exceptionally high gains in productivity for justifying the high investment.

- **Difficulties in adoption:** Majority of KBE implementations were black-box and had import-export integration with the design tools. Major drawback with this approach was that it lacked the finer controls required by engineers in iterative development.

Current drivers

Today, the rapidly changing scenario comes with multiple drivers, which necessitates a second look at KBE, from a different perspective.

- **Global product development:** To become lean and agile, organizations are adopting the principle of 'Design anywhere, Build anywhere'. Keeping that in mind, value of global product development is truly unlocked only when design practices are standardized and deployed across multiple locations.
- **Shorter development cycle:** Due to rising global competitive pressure, companies try to shorten the time to market as much as they can. To achieve that, it has become necessary to adopt new product development strategies and best practices.
- **Cost reduction:** It is essential that costs are managed effectively and efficiently in order to maximize Return on Investment. Organizations put forth all efforts to minimize both the recurring cost (i.e. materials & manufacturing cost) and the fixed cost (i.e. product development cost, new plant / machinery cost etc)
- **Set-based development:** Set approach creates and validates multiple options till late in the product development cycle. This requires creation of multiple options, without increasing the cost or the time to market.

Each of the above mentioned drivers are so complex that no single solution can address it in entirety. KBE helps to partially address each of these drivers.

Enablers

Today KBE is much more successful, due to combination of phenomenal changes in development processes and easy access to powerful hardware and software. Leveraging these enablers will help in quick RoI realization and give more control to end-users.

- **Concurrent development processes:** 'Over the wall' development approach is inadequate in today's fast-paced, high technology product development environment. Concurrent development process gathers knowledge encompassing broader process and can give much higher returns when deployed.
- **Enhanced documentation:** With global product development, there has been much greater emphasis on documentation, covering detailed instructions, rules, guidelines, etc. Such standard documentation makes it easier to identify need for KBE and comes handy for quickly deploying the solutions.

- **Easy to use KBE tools:** The technological advancements allow embedding of knowledge in frequently used engineering tools such as CAx, the spreadsheets and databases. This enables engineers to create, update and deploy their own knowledge.
- **Better integration across tools:** Today CAx tools are parametric and generative. They can be manipulated using rules and formulae and can also be linked to external data. Synergy between CAx modeling, KBE functions and external data makes it easier to deploy knowledge with finer control on the design process.

Key challenges

A successful KBE implementation needs to take care of a few critical challenges. Every implementation program attempts to address following issues with different level of success.

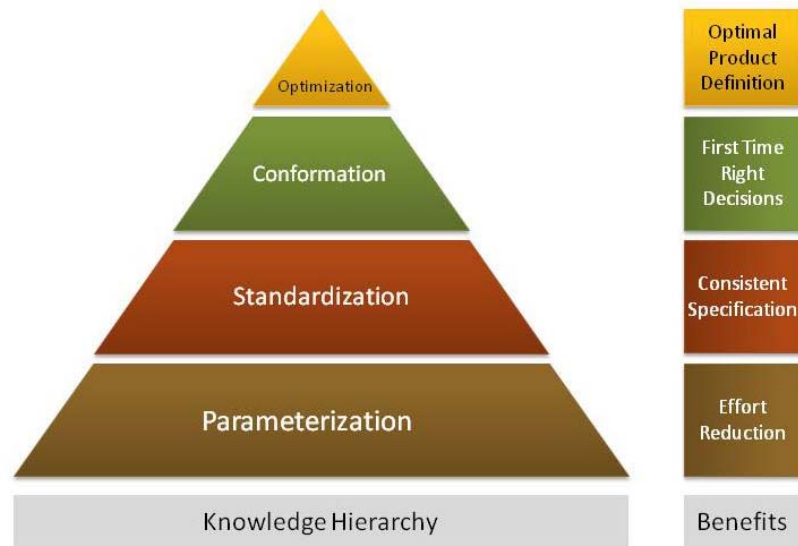
- **Shorter time:** Fast paced innovations endanger obsoleting of knowledge at a much faster rate. This shrinks the available window of opportunity for any KBE initiative. Thus, it becomes necessary to deploy knowledge in shortest possible time, with flexibility to dynamically tweak it later on, if need be.
- **Total cost of ownership:** Typically, applications developed on CAx-KBE integration require separate runtime licenses. This increases the running cost in proportion with the number of users. Hence it becomes necessary to explore ways to optimize the incremental cost.
- **Knowledge protection:** While exchanging data with the supplier/ OEM, knowledge embedded in design data may go outside the organization. Thus, KBE implementation needs to find ways to either control visibility of knowledge elements or to remove them completely before the data goes out.
- **Sustainability and extensibility:** Software systems are prone to changes and sometime there is a need to migrate from one system to another. Sustenance of knowledge could prove risky in such systems and they could also become restrictive for extension.

KBE implementation roadmap

The complex nature of KBE implementation makes big-bang KBE deployment quite difficult. To ease this, KBE implementations can be better planned with proper understanding of knowledge hierarchy and their dependencies. The hierarchy can be used for phased deployment of KBE in a particular function, while dependency can be used to align implementation of various phases across functions.

Knowledge hierarchy

- Engineering knowledge is expressed with parameters, standards, rules, formulae, etc. Knowledge hierarchy represents knowledge in the form of a pyramid, in particular order. Bottom up progression on the pyramid represents increasing intelligence and corresponding reduction in low level details manipulation, as shown below:



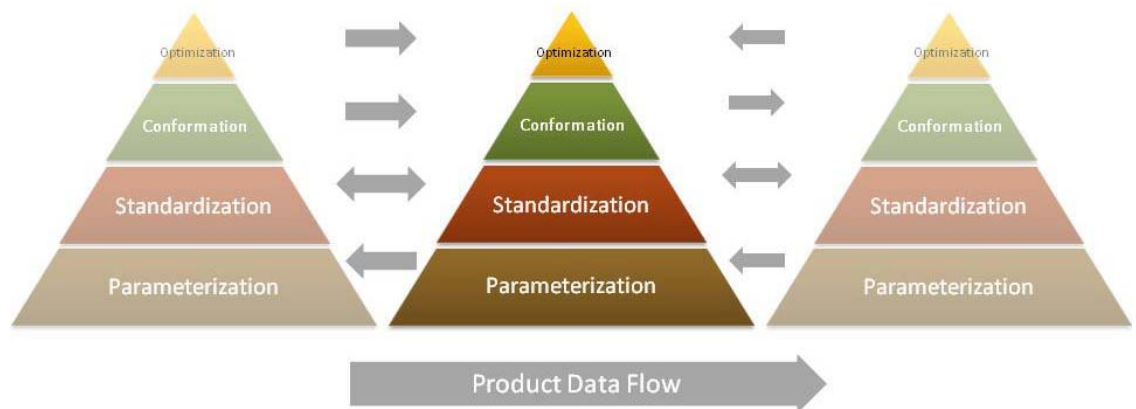
Implementing and deploying layers of pyramid hierarchy one at a time reduces adoption issues and also retains flexibility for course correction. Each layer brings out different facets of incremental RoI, which helps sustain organizational support for KBE. Let us look into the functions of each layer.

- **Parameterization:** Defines methods for creation of parts, features, manufacturing processes, etc and parameterizes these entities using generative capabilities of the CAx tools. Deploying parametric templates (viz. configurable parts, drawing templates, user defined features, manufacturing operations template, etc) helps reduce engineers efforts on these tasks.
- **Standardization:** Defines valid parameter ranges (e.g. dimensions, speed/feed, etc) and combination of parameter values (i.e. parameter sets). The ranges and sets are captured in databases, tables (aka family-table) or in the form of library of standard components. Driving parametric models with these standards bring consistency in design specification, manufacturing operations, among others.
- **Conformation:** Enforces rules and guidelines and expert's judgment by capturing them in the form of knowledge rules. Automated rule validation filters out the parameter ranges and sets that conform to the rules and engineers then get to pick from the filtered choices only. This process helps in reaching right decisions in the first attempt itself.
- **Optimization:** Builds optimization to minimize cost, weight, etc while meeting the constraints of design targets and design / manufacturing rules. The optimization process can be simplified by using various analytics and thumb-rules, derived from past experiences. Use of optimization helps in achieving the best possible product definition.

Knowledge dependencies

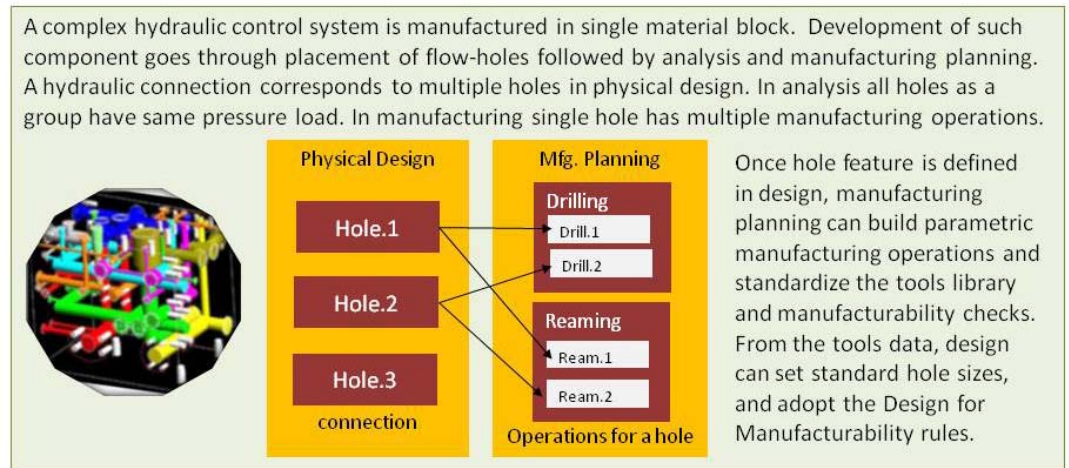
While each function in the product development captures and leverages its own knowledge, it also needs to leverage knowledge from other functions to achieve the full potential of KBE. However, for effective leverage, it is necessary to understand dependencies at each layer of knowledge hierarchy. Each layer in the hierarchy pyramid depends on - upstream functions for the data, and upstream/ downstream for rules, guidelines, etc. Since change in data form and processes affects the form of knowledge, the KBE implementation must align with these dependencies as well.

Each layer of knowledge pyramid has different dependencies, as follows.



- **Parameterization:** Parametric modeling in a particular function depends on upstream. When parametric modeling in a particular function matures, it sends parameterized features to downstream, after which the downstream functions implements its parameterization.
- **Standardization:** Depending on the type of standards, standardization has a combination of upstream and downstream dependency. e.g. standardization in product design depends on upstream (e.g. stock sizes) as well as on downstream (e.g. manufacturability standards).
- **Conformation:** If product development happens in successive manner, downstream functions keep on accumulating their leanings. As the learning matures, it is transferred to upstream functions in the form of rules and guidelines. Thus, conformation layer depends on downstream for knowledge transfer.
- **Optimization:** Optimization requires assimilation of knowledge from all the functions in product development. All the knowledge can be integrated to build the optimization functions, derive analytics and more. Therefore, optimization layer brings in complex simultaneous knowledge dependency.

The high level dependencies across functions percolate to feature / sub-feature level with Feature Connectivity Analysis. This analysis then maps features / sub-features across various functions. Parametric feature in one function can drive corresponding feature in downstream or the rules identified in downstream can be associated with corresponding features in upstream.

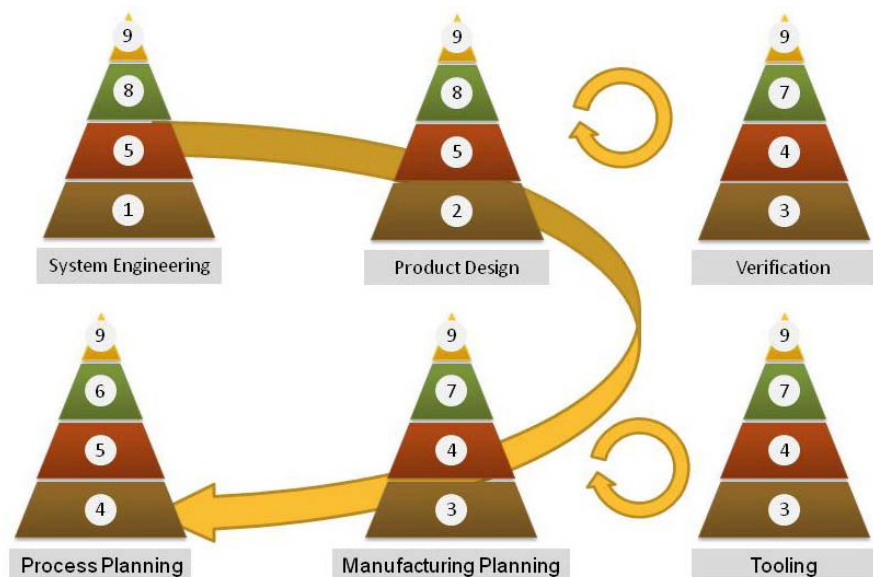


Implementation of the roadmap

KBE implementation across functions requires a careful consideration of various aspects, including what part of KBE can be implemented in a particular function, and how it can be leveraged in other functions; as well as how to build the next levels of KBE there onwards.

The roadmap for KBE implementation uses the knowledge hierarchy and knowledge dependencies. The roadmap defined at this stage considers system engineering, product design, verification, manufacturing planning, tool design and process planning as sub-functions.

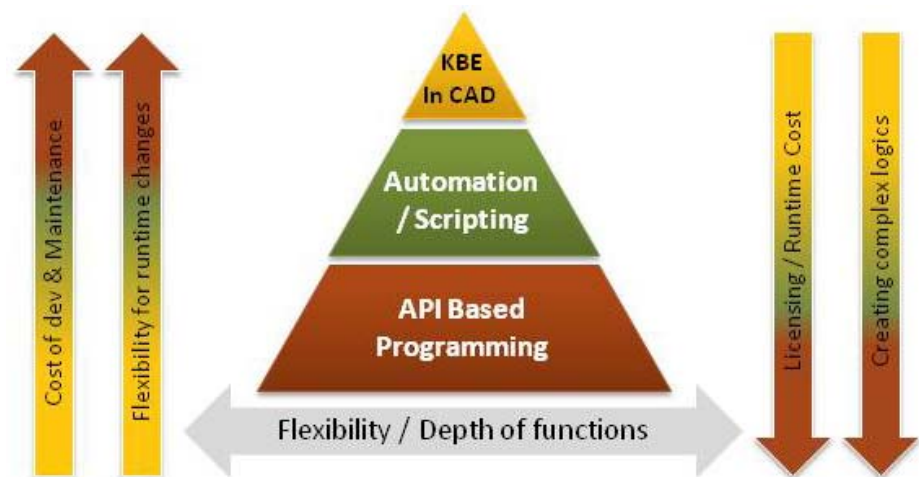
In these sub-functions, the product data flow is not sequential. System engineering data goes to product design, which is iteratively fine-tuned with verification. The fine-tuned design goes to manufacturing planning and tool design and eventually to process planning.



The above image depicts a KBE roadmap, defined with knowledge hierarchy and knowledge dependencies. The numbers indicate sequence of implementation. To reduce the overall duration of implementation, the sequencing advocates simultaneous implementation of a few layers of pyramid, when there are no direct dependencies.

KBE architecture

For developing KBE applications, various architectures are used. When KBE is initiated by engineering users, the architecture is predominantly based on KBE features available in CAD. On the other hand, when KBE is initiated by IT or by a KBE specialist, the architecture uses API based programming (on top of C or C++ API toolkits). Some KBE initiative, however, find middle-ground with automation (e.g. VB, VB .NET, etc) and scripting based architectures. Each one of these architectures bring in their own set of pros and cons, in terms of development cost, ease of maintenance, knowledge protection, ability to manipulate low level details, among others.



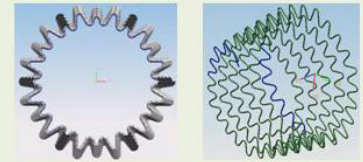
Of the three alternatives mentioned, API based programming is difficult to maintain within the purview of Engineering organization. Remaining two are typically managed under engineering organizations. As this paper focuses on Engineering organizations, next part of this section explains use of KBE tools with its advanced features and how it is complimented by scripting and spreadsheet based formulations.

KBE tools in CAD

Today most CAD tools provide some form of KBE support. It comprises of parametric features, formulae, rules, and other types of relations for driving the parameters. Users can make use of these functions to modify the dimensions and to activate / suppress features. They can also set different configurations of model through parameter-sets (aka design-tables or family-tables). Additionally, by using these functions, users can build their parametric features and enforce standardization across various functions.

Heating coil in consumer electric equipments is designed based on heating capacity and packaging requirements. Heating capacity corresponds to resistivity, wire diameter and total length. To fit long wire in smaller space, wire is formed in peaks & troughs and mounted on fixture in helical shape.

The heater coil is created as a parametric part, driven by wire diameter, peak & trough specification, coil diameter. For any new configuration, the engineer modify the driving parameters and check wire length, which is set as output parameter.



As the number of parametric features increase, scalability issues start showing up. Typically, this results in performance degradation and data size inflation, sometimes making the model impossible to use. To avoid such situation, users should start exploring features available in the KBE functions that are similar to programming. These features provide ways for conditional, on the fly feature creation. This greatly speeds up the operations and reduces the data size [Refer 1].

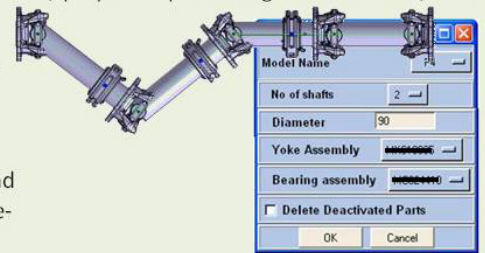
Scripting and automation

CAD tools provide scripting and automation based support for automating the routine tasks. Thus, if it allows access to each feature and its parameters, it is possible to control parametric values and state of each feature (active / suppress), based on some complex formulae or rules coded in the scripting language. As it does not access any of the KBE features, it can work with even basic CAD functions without any special KBE license. Depending on programming language used (Perl, VB, etc), it is possible to connect with external applications, databases, etc to implement some complex logic in KBE.

Drive shafts for heavy construction machines are made up of multiple segments and universal joints. Design and detailing of the shaft is based on shaft diameter, polyline representing drive axis and, and specification of universal joints and bearings.

The KBE solution for drive shaft consists of parametric models for shaft, bearings and universal joints, and a master shaft assembly.

Engineer specify the driving parameters, based on which VB script manipulates master shaft assembly and parametric models to create full definition of the drive-shaft.




However, creating complex logics in automation requires experienced programming expertise, which may not be intuitive for engineers. Hence, it is better to use spreadsheets to resolve dimensions and activation state through complex logics. The automation synchronizes parameters from spreadsheet to CAD and vice versa. Engineers who are more comfortable with spreadsheets can harness the full power of spreadsheet application to put various complex logics, and validate the same without going into CAD systems.

Welding fixture components have multiple drilled holes for dowel pins and fasteners. Fastener holes are manufactured by drilling, counter-boring, and tapping. Dowel holes are manufactured by drilling and reaming. For each hole, machining speed and feed depends on diameter and depth of holes.

A KBE solution for machining time evaluation consists of a Excel spreadsheet and a script. Spreadsheet has speed / feed data for each operation and is configured to machining time for hole diameter and depth set in a fixed cell.

Script scans through the 3D part for hole features. For each hole, parameters are set to excel and the manufacturing time is read-back. Script adds up time for all holes to evaluate total manufacturing time.



Fixed cells to set dia / depth and read time

	A	B	C	D	E	F	G	H	I
1	Diameter	Depth	Type	Time					
2	10	25	98.33333	98.33333					
3									
4									
5									
6	Diameter	Feed	Time						
7	10	1	25	0.75	13.33333	0.25			
8	12	1	25	0.75	13.33333	0.4			
9	15	1	25	0.5	20	0.4			

Speed / Feed data configured to evaluate time for each operation

Conclusion

This whitepaper presented an approach for KBE implementation considering interdependencies of product development functions and constrained budgets for such initiatives. It presented architectures that engineers can use for developing and maintaining their KBE solutions. With the suggested roadmap, it is possible for engineers to control the KBE solutions to bring maximum return on the investments.

References

- Shintre, Nikhil and Brian Prasad, "Effective Use of Knowledgeware for Managing Model Size and Improving Performance ", COE NewsNet, Jan-Feb 2010.
- Shintre, Nikhil and Brian Prasad, "Hybrid Architecture for Editable PowerCopy", COE NewsNet, Jul-Aug 2010.

About the Author

Nikhil Shintre is responsible for Knowledge Based Engineering practice at Geometric. He has over 12 years of experience in engineering process optimization and automation. Nikhil holds a Masters Degree in Machine Design from the Indian Institute of Technology, Mumbai. He can be reached at nikhil.shintre@geometricglobal.com

Aslam Shakir is handling the CAD/KBE projects in Geometric. He has over 10 years experience in working with various Automotive OEMs and executing engineering design projects. He can be reached at aslam.shakir@geometricglobal.com

About Geometric

Geometric is a specialist in the domain of engineering solutions, services and technologies. Its portfolio of Global Engineering services and Digital Technology solutions for Product Lifecycle Management (PLM) enables companies to formulate, implement, and execute global engineering and manufacturing strategies aimed at achieving greater efficiencies in the product realization lifecycle.

Headquartered in Mumbai, India, Geometric was incorporated in 1994 and is listed on the Bombay and National Stock Exchanges. The company recorded consolidated revenues of Rupees 6.21 billion (US Dollars 136.47 million) for the year ended March 2011. It employs over 3900 people across 10 global delivery locations in the US, Romania, India, and China. Geometric was assessed as CMMI 1.1 Level 5 for its software services and is ISO 9001:2008 certified for engineering operations. The company's operations are also ISO 27001:2005 certified.

The copyright/ trademarks of all products referenced herein are held by their respective companies.