7th International Quality Conference
May 24th 2013
Center for Quality, Faculty of Engineering, University of Kragujevac

Predrag Pravdic 1)

*1) Faculty of mechanical engineering Kragujevac, Industrial engineering, University of Kragujevac
thepera81@gmail.com*

QUALITY OF CAD/CAM SOFTWARES IN MANUFACTURING INDUSTRY

Abstract: *A usually highly controversial issue in life cycle impact assessment is the weighting of impact categories, as this is a subjective issue. The first part of the paper describes the proposed approached where analytic hierarchy process is used to calculate the weighting factors of impact categories in life cycle impact assessment. Further on, proposed approach is implemented on example where environmental impacts of material production for three products are compared. Three scenarios have been assembled with different weightings factors of impact categories and obtained results were discussed in conclusion of this paper.*

Keywords: *weighting factor, life cycle, impact category, analytic hierarchy process*

1. INTRODUCTION

In the manufacturing industry, as well as in other industry sectors, embedded computer systems are increasingly used to enhance and extend the functionality of previously purely mechanical or electrical systems. This development has led to more versatile, but also more complex systems, and system complexity increase with each new developed product generation. To cope with complexity and stringent requirements on availability, reliability and real-time response more emphasis is placed on the early stages of system development — i.e., systems engineering. Computer based specification, design and analysis tools have been introduced to assist systems engineers in this process. This has introduced new problems. Computer based tools typically cover specific aspects of the systems engineering process. Moving information between tools, in cases where there is an overlap in information coverage, is difficult due to the absence of accepted tool independent data formats. The resulting situation has been characterised as one of “islands of automation” [1]. Furthermore, in cooperation projects it is often the case that partner organisations uses different tools for accomplishing the same task. Also in this case automated information exchange is obstructed by the lack of tool support for data exchange. As a result, specifications sent for further refinement or analysis in foreign tools often have to be re-entered manually into the receiving organisation’s tool-set, a tedious and error prone process. The lack of exchange mechanisms also complicate upgrades from legacy tools to modern ones. Information entered or maintained in the legacy tool will not be accessible from the modern tool unless tool interfaces are developed. Tool integration and especially data integration methods ([2],[3]) have been proposed for enabling data exchanges between tools and a lot of effort have been placed into the development of tool neutral repositories such as PCTE [4] and infrastructure that facilitates for tool integration.

But these effort have neither been

widely accepted nor led to any significant increase in data exchange capabilities for systems engineering tools [5]. In a critical article, Brown and McDermid [6] attributes the lack of success for these methods to the misconception that open tools automatically leads to integrated tools. The view expressed is that open frameworks at best helps integration. Unless significant effort is placed on the development of common information models agreed by tool users and vendors the vision of an open environment will stay a vision. Today, with the use of computer technologies and communication technologies in the manufacturing industries, manual and semi-automatic methods are largely being replaced by Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) to implement concurrent engineering [7]. Widespread CAD/CAM systems will reduce human interaction and the result, should beincreased production, reduced costs and better quality of product. CNC machines now, utilize a variety of cutting technologies such as multi turrets and multi spindles in various axial configurations increasing the level of complexity compared to the machines of the previous decade [8].

A large number of CAD/CAM systems have been developed and implemented in recent years to support all stages of product life by computer systems and many can simulate virtual CNC machining with the complete machine toolpath [9]. Most of these systems are specialized to support certain applications, and are based on an information model that handles the application specific view of the product. These CAD/CAM systems do not share common databases for the product information. Since the first NC machine was introduced in 1947, various process planning packages have been developed and each system tried to interpret the part data format more reliably. To date there are more than 2000 CNC models around the globe, and turning centers need a single standard particularly in the area of machining to improve productivity by increasing the richness of interactions and transactions.

Planning and developing the new products are considered as one of the most important part of the products life cycle. However, so many problems encounter when attempts are done for new product design [10]. These problems usually are related to the exchange of product data and information between different parts of the enterprises [11]. An intelligent interface between CAD and CAPP systems is imperative because the CAPP systems depend on correct data obtained from CAD systems to perform precise process planning [12]. Considering the CAD/CAM systems that use different data structures for processing; one of the most important weaknesses of these systems is due to their nonintegrated infra structures and information [11]. In general extracting manufacturing information from CAD database is not adequate, since current CAD systems do not provide all manufacturing and process planning data. Part model should include not only geometric and topological information but also technological information namely features, surface roughness, hardness, etc. for CAM systems [13]. The researchers express that the problem is the accurate and meaningful: product data is not visible to the personnel that need it [14]. Teams usually use their tools for work that generates product data in various formats. These formats make a problem for collaborative product development. This problem is a common problem just because there are so many different tools to work on the new products design and old designs revisions, it is believed that there are so many different CAX on the market; it is evident that the Product Data Technology objectives can only be realized via standards [15]. The extended enterprises involved with product development processes are heterogeneous environments beset with disparate CAD models [16]. In fact, the trend towards global manufacturing exacerbates it - producing a constantly shifting web of product development and delivery partnerships. The lack of effective interoperability among the extended enterprise threatens product quality, drives up costs, and lengthens time-to-market [16]. This is a new paradigm for CAD/CAM computer systems based in global environments, network-centered and spatially distributed, enabling the development of activities using e-Work. This will allow the product designers to have easier communication, making it possible the sharing and collaborative design during product development, as well as the teleportation and monitoring of the manufacturing equipment [17].

2. QUALITY OF CAD/CAM SOFTWARE PRODUCTS AND PROCESSES

Software products exhibit two general kinds of quality, which affect software schedules in different ways. The first kind of quality that people usually think of when they refer to „software quality" is low defect rate.

Some project managers try to shorten their schedules by reducing the time spent on quality-assurance practices such as design and code reviews. Some shortchange the upstream activities of requirements analysis and design. Others – running late – try to make up time by compressing the testing schedule, which is vulnerable to reduction since it's the critical-path item at the end of the schedule.

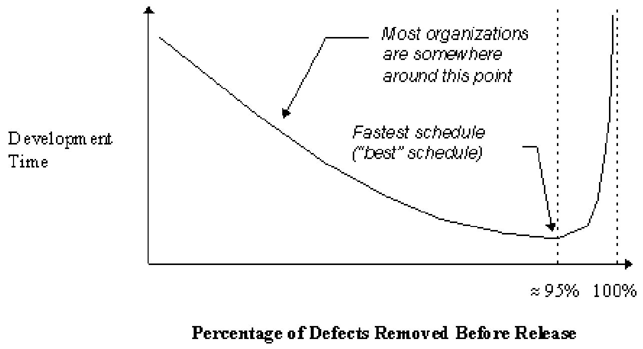

Figure 1. Relationship between defect rate and development time

These are some of the worst decisions a person who wants to maximize development speed can make. In software, higher quality (in the form of lower defect rates) and reduced development time go hand in hand. Figure 1 illustrates the relationship between defect rate and development time.

A few organizations have achieved extremely low defect rates (shown on the far right of the curve), and when you reach that point, further reducing the number of defects will tend to increase the amount of development time. This applies to life-critical systems such as the life-support systems on the space shuttle. It doesn't apply to the rest of us. The rest of us would do well to learn from a discovery made by IBM in the 1970s: Products with the lowest defect counts also have the shortest schedules [18]. Many organizations currently develop software with defect levels that give them longer schedules than necessary. After surveying about 4000 software projects, Capers Jones reported that poor quality was one of the most common reasons for schedule overruns (1994) [19]. He also reported that poor quality is implicated in close to half of all canceled projects. A Software Engineering Institute survey found that more than 60 percent of organizations assessed suffered from inadequate quality assurance [20]. On the curve in Figure 1, those organizations are to the left of the 95-percent-removal line. That 95-percent-removal line-or some point in its neighborhood-is significant because that level of pre-release defect removal appears to be the point at which projects achieve the shortest schedules, least effort, and highest levels of user satisfaction [18]. If you're finding more than 5 percent of your defects after your product has been released, you're vulnerable to the problems associated with low quality, and you're probably taking longer to develop your software than you need to. Projects that are in a hurry are particularly vulnerable to shortchanging quality-assurance at the individual-developer level. Any developer who has been pushed to ship a product quickly knows how much pressure there can be to cut corners because „we're only three weeks from shipping." The upshot is that a shortcut that was supposed to save time actually wasted time in the following ways:

- The original time spent designing and implementing the printing hack was completely wasted because most of that code will be thrown away. The time spent unit-testing and debugging the printing-hack code was also wasted.
- Additional time must be spent to strip the printing-specific code out of the display module.
- Additional testing and debugging time must be spent to ensure that the modified display code still works after the printing code has been stripped out.
- The new printing module, which should have been designed as an integral part of the system, has to be designed onto and around the existing system, which was not designed with it in mind.
- All this happens, when the only necessary cost-if the right decision had been made at the right time-was to design and implement one version of the printing module. And now you still have to do that anyway.

This example is not uncommon. Up to four times the normal number of defects are reported for released products that were developed under excessive schedule pressure. Projects that are in schedule trouble often become obsessed with working harder rather than working smarter. Attention to quality is seen as a luxury. The result is that projects often work dumber, which gets them into even deeper schedule trouble. One aspect of quality assurance that's particularly important to rapid development is the existence of error-prone modules, which are modules that are responsible for a disproportionate number of defects. On its IMS project, IBM found that 57 percent of the errors clumped into 7 percent of the modules. Modules with such high defect rates are more expensive and time-consuming to deliver than less error-prone modules. Normal modules cost about $500 to $1000 per function point to develop. Error-prone modules cost about $2000 to $4000 per function point to develop. Error-prone modules tend to be more complex than other modules in the system, less structured, and unusually large. They often were developed under excessive schedule pressure and were not fully tested.

If development speed is important, make identification and redesign of error-prone modules a priority. Once a module's error rate hits about 10 defects per thousand lines of code, review it to determine whether it should be redesigned or reimplemented. If it's poorly structured, excessively complex, or excessively long, redesign the module and reimplement it from the ground up. You'll shorten the schedule and improve the quality of your product at the same time. If you can prevent defects or detect and remove them early, you can realize a significant schedule benefit. Studies have found that reworking defective requirements, design, and code typically consumes 40 to 50 percent of the total cost of software development [21]. As a rule of thumb, every hour you spend on defect prevention will reduce your repair time from three to ten hours. In the worst case, reworking a software requirements problem once the software is in operation typically costs 50 to 200 times what it would take to rework the problem in the requirements stage [22]. It's easy to understand why. A 1-sentence requirement can expand into 5 pages of design diagrams, then into 500 lines of code, 15 pages of user documentation, and a few dozen test cases. It's cheaper to correct an error in that 1-sentence requirement at requirements time than it is after design, code, user documentation, and test cases have been written to it. Picture 2 illustrates the way that defects tend to become more expensive the longer they stay in a program.

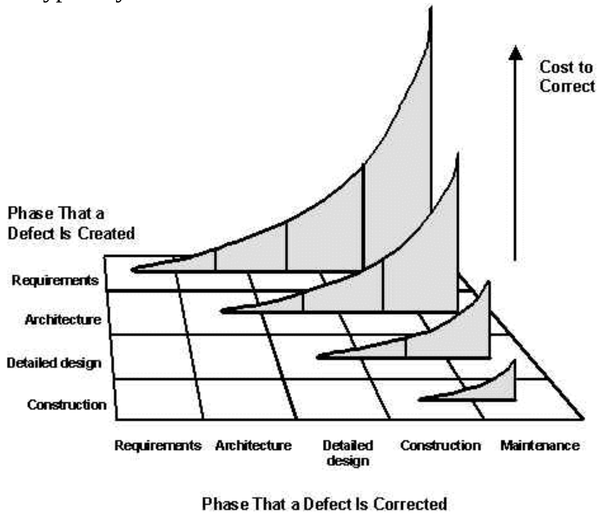

Figure 2. The longer a defect remains undetected, the more expensive it becomes to correct

The savings potential from early defect detection is huge-about 60 percent of all defects usually exist by design time [23], and you should try to eliminate them by design time. A decision early in a project not to focus on defect detection amounts to a decision to postpone defect detection and correction until later in the project when they will be much more expensive and time-consuming. That's not a rational decision when time is at a premium. The most common quality-assurance practice is undoubtedly execution testing, finding errors by executing a program and seeing what happens. The two basic kinds of execution testing are unit tests, in which the developer checks his or her own code to verify that it works correctly, and system tests, in which an independent tester checks to see whether the system operates as expected. Testing is the black sheep of QA practices as far as development speed is concerned. It can certainly be done so clumsily that it slows down the development schedule, but most often its effect on the schedule is only indirect. Testing discovers that the product's quality is too low for it to be released, and the product has to be delayed until it can be improved. Testing thus becomes the messenger that delivers bad news. The best way to leverage testing from a rapid-development viewpoint is to plan ahead for bad news--set up testing so that if there's bad news to deliver, testing will deliver it as early as possible. Technical reviews include all the kinds of reviews that are used to detect defects in requirements, design, code, test cases, and other work products. Reviews vary in level of formality and effectiveness, and they play a more critical role in maximizing development speed than testing does. The least formal and most common kind of review is the walkthrough, which is any meeting at which two or more developers review technical work with the purpose of improving its quality. Walkthroughs are useful to rapid development because you can use them to detect defects earlier than you can with testing. Code reading is a somewhat more formal review process than a walkthrough but nominally applies only to code. In code reading, the author of the code hands out source listings to two or more reviewers. The reviewers read the code and report any errors to the code's author. A study at NASA's Software Engineering Laboratory found that code reading detected about twice as many defects per hour of effort as testing [24]. That suggests that, on a rapid-development project, some combination of code reading and testing would be more schedule-effective than testing alone. Inspections are the most formal kind of technical review, and they have been found to be extremely effective in detecting defects throughout a project. Developers are trained in the use of inspection techniques and play specific roles during the inspection process. The „moderator“ hands out the material to be inspected before the inspection meeting. The „reviewers“ examine the material before the meeting and use checklists to stimulate their reviews. During the inspection meeting, the "author" paraphrases the material, the reviewers identify errors, and the "scribe" records the errors. After the meeting, the moderator produces an inspection report that describes each defect and indicates what will be done about it. Throughout the inspection process you gather data about defects, hours spent correcting defects, and hours spent on inspections so that you can analyze the effectiveness of your software-development process and improve it. Because they can be used early in the development cycle, inspections have been found to produce net schedule savings of 10 to 30 percent [25]. One study of large programs even found that each hour spent on inspections avoided an average of 33 hours of maintenance, and inspections were up to 20 times more efficient than testing [26]. Technical reviews are a useful and important supplement to testing. Reviews find defects earlier, which saves time and is good for the schedule. They are more cost effective on a per-defect-found basis because they detect both the symptom of the defect and the underlying cause of the defect at the same time. Testing detects only the symptom of the defect; the developer still has to isolate the cause by debugging. Reviews tend to find a higher percentage of defects [21]. And reviews serve as a time when developers share their knowledge of best practices with each other, which increases their rapid-development capability over time. Technical reviews are thus a critical component of any development effort that aims to achieve the shortest possible schedule. Software frameworks and programs that facilitate the distributed product design and manufacturing are becoming more and more important in product development processes [27, 28]. Different solutions and software have been developed based on these frameworks including different CAD/CAM application **software tools. Technologies developed for CAD/CAM application software**

tools and CNC post processors are customized within each of their own application domains named as automation islands [29-31]. So, the application of these software tools in different enterprises will make trouble where it is necessary to exchange product data among engineers and designers who are geographically spread and have different goals, knowledge, experiences, tools and resources-to support collaborative product development within an integrated product data structure [32-34]. These problems can be classified in three group:

1) Management of collaboration between different Cax application software tools in product development processes.
2) Enabling the interoperability among different Cax software application tools for product data exchange.
3) Integration of product data through enterprises' product development processes while different CAx application software tools modify the product data.

To solves these problems, researchers proposed different CAx platforms and information systems. Theses platforms and information systems struggled to solve above problems. Qin in 2004 [35] introduced information system with the ability to manage the CAx application software collaboration and integrated data structure but it was not an interoperable platform and lacked for the ability to enable the product data exchange among different CAx application software. XU in 2005 [30], 2009 [29] and Nassehi in 2006 [36] suggested information systems with the ability for product data integration based on standards. However, these systems lacked to manage the CAx application software collaboration and enabling the interoperability for product data exchange between different CAx application software. On the other hand, Peng in 1998 [32] and LO´ Pez-Ortega in 2005 [37] proposed platforms to facilitate the product data exchange between different CAx application software but they lacked for an integrated product data structure. Lee in 2006 [38], Canciglieri Junior in 2005 [39], Pisarciuc in 2007 [40], Nguyen Van in 2007 [41] and Newman in 2007 [42] proposed platforms that had an integrated product data structure and also enabled product data exchange among different CAx application software tools but they had no procedures and structures in their platforms to manage the CAx application software collaboration. Considering the mentioned platforms, this article suggests the basic needs for an interoperable and collaborative platform with an integrated product data structure that is enable to solve the mentioned problem. The basic needs for such a platform are:

- CAD Interoperability:
  CAD interoperability is known as a critical factor for using different CAx application software in different enterprises [35, 43-46]. There are many situations where it is essential to exchange CAD model data between different CAD systems, or between a CAD system and some other computer-based application system [47, 48]. The platform architecture should be designed in such a way to support exchange of different data structures related to CAD application software working collaboratively through enterprises. A long time as a key concept for concurrent engineering which tries to coordinate product development processes involved designers from different departments in the same company, as well as from different enterprises [49, 50]. The interoperability of different CAM application software reduces the production cycle time and costs as well as increasing product quality [51-54]). Different CAM application software should be able to work

within the platform base on their data structure format. Researchers believe [55, 56] that different CAD software tools must communicate with different CAM software tools to provide the smooth transition from design to manufacturing through enterprises. CAM must communicate with computer numerically controlled (CNC) machine tools to execute the manufacturing activities.

- Management of CAx collaboration:
  Product development processes, comprised of product design, product process planning and manufacturing, is a complicated process which involves groups of designers and manufacturers often with conflicts [52]. It is necessary to share design information within a manufacturing company through the different infrastructures, to enable the company to exchange product information with parts suppliers and collaborating partners in addition to sharing information among internal departments [56, 57]. This need emphasises on managing the product data modification among different CAx application software through enterprises. On the other hand, globalization in the manufacturing sector is increasing the need for more effective means of collaboration and communication among extended product development teams - including the ability to share product data [50, 54].
- Supporting distributed enterprises CAx development teams:
  Modern manufacturing enterprises are comprised of several CAx development teams spread around the globe, which often contain equipment and software tools from different manufacturers. Immense volume of product information must be transferred between the various facilities and Cax software tools at different locations [56]. It's necessary for a collaborative platform to support distributed Cax development teams. To complete a design, negotiations among Cax development teams should take place continuously [49]. It gets more important where designers are geographically dispersed; the platform needs to create a naturally collaborative environment in which each authorized participant can communicate with each other without obstacles [49, 58]. Therefore the platforms should have a structure to support distributed product data. The platform should secure the exchange of product data among authorised CAx development teams.

3. CAD AND CAM SOFTWARES IN THE MANUFACTURING PROCESSES

In order to manufacture a part, nowadays typically three different software programs are used (Figure 3):

- First the **CAD software** to make the design of the part
- Next the **CAM software** to calculate the toolpaths based on the design, compensating for the cutter's geometry, adding feedrate and spindle commands, etc
- Third the **control software** to read the toolpaths and let the machine actually move along these paths.

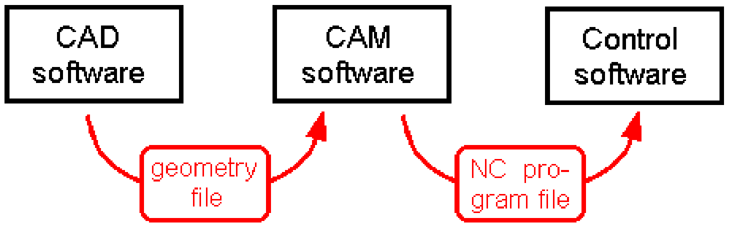

Figure 3. Manufacturing with CAD/CAM

This subdivision of tasks by three different programs is the same for both 2D and 3D applications. The Control software comes with the milling machine, while in contrast the CAD and the CAM software have to be bought separately. In case an NC controller with built in special purpose computer is used, the Control software is integrated. In case of a PC the Control software has to be installed, still it "belongs" to the machine as the machine cannot function without the Control software. Do note that many lowcost CNC machines are delivered with MS Dos based Control software, as the realtime control needed for machining is difficult to program under Windows. Communication between the three programs is done using files. From CAD to CAM the design is transferred using a file format for geometry data exchange. For instance file types like IGES, STL for 3D, DXF for both 2D and 3D, and Postscript and HPGL for 2D applications. These are standard formats that in most cases can be used without any special configuring needed. Communication from CAD to Control software is done using NC program files, for which many formats do exist. In most cases the format will be a (minor) variation on the ISO / DIN G-code format. G-code is supposed to be a standard, however in practice each manufacturer chooses a bit different implementation. So for this communication the CAM software has to fine tune its output in order to meet the requirements of the NC controller used. This fine tuning is done by the Postprocessor. The Postprocessor is the part of the CAM software that translates the toolpath data into the correct file format when saving (in fact an export filter). This functionality is the same as used in the (Windows) device driver that comes with any printer, to translate the wordprocessor's output to the format required by that printer. In many current CAM systems the postprocessor can be configured by the user, making it easy to connect to any new machine. This in contrast to the older CAM packages where the user has to pay (much) money to the supplier to order a new postprocessor. For instance some machines can be used without control software (like the small Roland models, where a plain "print" command is sufficient to start the machine). Or in some setups a plot file (or even a 2D DXF file) from the CAD system can be immediately sent to the control software, skipping the CAM step. Still it makes sense to distinguish the three basic steps, for a clear picture of the process involved. Many different CAM software packages are available, showing large differences in price. It is a very difficult job to get a clear view about each system's capabilities and it's price/performance ratio. A clear difference is present between CAM software for 2D and for 3D applications. With 2D is meant that the CAM system imports a 2D drawing file and calculates a toolpath with all movements taking place on a constant Z-level. Obviously several toolpaths on different Zlevels can be combined to create a 3D result, which is called 2.5 D machining (note that more definitions of the notion 2.5D do exist). In that case to user has to enter the correct Z-level to be used for each toolpath. A 3D CAM system in contrast imports a full 3D CAD model and calculates toolpaths to create a 3D result. Note that in this case also toolpaths on constant Z-level may be used (waterline machining), however these are automatically generated. Many CAM packages do offer both 2D and 3D, however still have their clearly recognizable foundation in one of both fields. A second distinction is between simple and high-end CAM software. The high-end stuff is meant for professional toolmakers, who know about all possible milling parameters, want to be able to control any parameter for an optimum result, and are willing to pay for that. These extra high-end parameters do include options like:

- Support for a fourth axis, or for full 5 axis machining.
- Optimization for High Speed machining (constant tool load).
- Special sequences for approaching and leaving the geometry (lead-ins).
- Automatic stepover calculation.
- A wide choice of machining strategies, like parallel, spiral, radial, pencil tracing, flat surface recognition, offset machining, plunge milling and automatic smoothing of almost vertical surfaces.
- Automatic detection and removal of rest material.
- Management of undercuts.
- Rendered machining simulations.

The more simple programs offer less options, and are thus both cheaper and easier to use. The word "NC Program" can be used in two senses, which can be quite confusing when communicating about CAM. As the word "Program" in most cases means Software, sometimes the word "NC program" is used for a 'Program to calculate NC data', so for the CAM software. The capabilities of your machine must of course match the requirements of the CAM software. One requirement should be checked, as some older CNC machines do not offer this: 3D line interpolation. This means the possibility to travel from point A to point B in a straight line in full 3D. This is not easy, as all three axes will have to keep up a different speed. Some machines are only capable of straight lines involving 2 axes (2D line interpolation). A second important machine capability is on-line machining: the capability to handle large NC program files directly from the computer's hard disk. This involves some handshaking between the PC and the controller, as in most cases the data transfer will be faster than the actual machining.

4. CONCLUSION AND DISCUSSION

Computer aided design software is becoming more sophisticated all the time. As research into new levels of engineering and production continues to be explored, CAD and CAM software is being developed to streamline the processes and give designers more flexibility. CAD software can easily be integrated into your current operations with the right kind of systems.

With the expansion of global markets and the need for quick decisions and model improvements, along with the value of the resources involved in the creation of new products, you’ll need to stay on top of all the innovation and emerging information available about computer aided manufacturing software. To remain competitive you’ll have to find ways to include the latest CAD/CAM programs in your operations. It starts by encouraging companies to assess their needs for increased productivity and faster programming, the CAD systems they need to communicate with, and the type of machining they carry out. 5-axis CNC machining, for example, requires a much more advanced system for efficient operation, and will also benefit from collision avoidance rather than just collision detection. Testing CAM software on real components checks that the CAD interface is reliable, and lets engineers see the difference in the quality of toolpaths. The importance of this cannot be overestimated, as it has a huge impact on design translation, tool life, cycle times and the quality of the finished part. When the system is in use, the higher savings to be made with a more efficient system will pay huge dividends. An intuitive interface shortens the learning curve and enables a CAM system to be used by more people in an organization, increasing flexibility and reducing the skill levels required. The guide also emphasizes the importance of the quality of the CNC toolpath. The easiest system is only as good as the results it produces. Other areas considered include how the software adapts to use in different countries, how it interfaces with CAD systems, and how it can expand to meet the evolving needs of users. Key benefits include the ability to cut with a short tool and to manufacture metal parts in one operation. The software needs to be able to detect where material is left on the job, incorporate the kinematics of the machine tool, and avoid collisions. It also shows how automated 5-axis systems can bring CNC machining within the capabilities of most companies. Automated CAM software is the next step for many companies. Machine operators understand metal cutting, so utilizing their skills in the CAM software produces a considerable boost in manufacturing efficiency. It also explains how automation can result in consolidation of the tools used and greatly increase the consistency and quality of components. Also known as computer-aided design/computer-aided manufacturing software, CAD-CAM software is a computer system that many manufacturing companies find extremely beneficial. Using this software, engineers and machinists can design and, at the same time, control various manufacturing processes. The specialized software system provides them with computer-aided visual support making it easier for them to map out how machines cut or hew designs, as well as perform other machining and manufacturing processes. At the same time, the system allows them to design virtual parts on a computer screen efficiently and accurately. The specialized computer system can guarantee more efficient processes, as well as top quality products. It is also the perfect software that will help manufacturing industries quickly adapt to their client's changing needs and expectations. Many of today's suppliers of CAD-CAM systems ensure that the system features various machining methods and allows engineers and machinists to easily handle and manage complex programming tasks. Some of the basic machining methods that users can do with this software are threading, groove finishing and roughing, drilling, tapping, boring and reverse boring.

If you are thinking about utilizing the software, you should be aware that the key to getting successful with it is to ensure that your CAD-CAM operators are knowledgeable, skilful and have high experience level in handling and operating the software. The more knowledgeable and skilful the operators are, the more benefits you will get from utilizing the software. Many manufacturing companies have been impressed with the huge benefits that CAD-CAM software provides. Utilizing the CAD-CAM computer system, whether in 2D or 3D environment, is sure to make a certain business outpace its competitors. The software allows engineers and machinists to develop and produce top-quality machines in a short span of time and at a lower cost.

REFERENCES:

[1] Brown, A., Carney, D., Morris, E., Smith, D., & Zarrella, P. (1994). *Principles of CASE Tool Integration.* OXFORD Press.

[2] Schefström, D., & van den Broek, G. (1993). *Tool Integration - Environments and Frameworks.* John Wiley and Sons.

[3] Wasserman, A. I. (1989). *Tool integration in software engineering environments*. In Long, F., editor, Software Engineering Environments, LINCS 467, 137-149.

[4] Long, F., & Morris, E. (1993). *An overview of pcte: A basis for a portable common tool environment.* Technical report, CMU/SEI.

[5] Sutherland, J, Hill, D., Milne, A., & Cowan J. B. (1996). *Functionality integration within and between software development tools*. In Baise, editor, Proceeding of the 5th Software Quality Conference, volume 1, pages 135-145, July.

[6] Brown, A., & McDermid, J. (1992, March). Learning from ipse's mistakes. *IEEE SOFTWARE,* 23-28.

[7] Newman, S. T., Nassehi, A., Xu, X. W., Rosso Jr, R. S. U., Wang, L., Yusof, Y., … Dhokia, V. (2008). Strategic advantages of interoperability for global manufacturing using CNC technology. *Robotics and Computer-Integrated Manufacturing, 24*, 699-708.

[8] Nassehi, A., Allen, R. D. & Newman, S. T. (2006, June). *Intelligent Replication of Manufacturing Information between CAD/CAM Systems and CNC Controllers*. Presented at Proceedings of the 16th International Conference on Flexible Automation and Intelligent Manufacturing Conference, Limerick, Ireland.

[9] Newman, S. (2004). Integrated manufacture for the 21st century Development of the STEP-NC standard and its implications for manufacturing processes worldwide. *Metalworking Production, 148,* 13-16.

[10] Peng, T. K., & Trappery, A. (1998). A step toward STEP-compatible engineering data management:the data models of product structure and engineering changes. *Robotics and computer-integrated manufacturing, 14*, 89-100.

[11] Pratt, M., Anderson, B., & Rangerc, T. (2005). Towards the standardized exchange of parameterize feature-based CAD models. *Computer-Aided Design, 37*, 1251-1265.

[12] Yildiz, Y., Korkut, İ., & Şeker, U. (2006). Development of a Feature Based CAM System for Rotational Parts. *G.U. Journal of Science, 19*, 35-40.

[13] Başak, H., & Gülesin, M. (2004). A Feature based parametric design program and expert system for design. *Mathematical and Computational Applications, 9*(3), 359-370.

[14] Burkett, W. (2001). Product data markup language: a new paradigm. *Computer-Aided Design, 33*, 489-500.

[15] Gielingh, W. (2008). An assessment of the current state of product data technologies. *Computer-Aided Design, 40*, 750-755.

[16] Markson, H. Achieving CAD Interoperability in Global Product Design Environments. Retrieved March 1, 2008, from: http://www2.spaceclaim.com/LearnMoreNow/WhitePapers.aspx.

[17] Álvares, A., & Ferreira, J. C. (2008). *Web Cad By Features: Collaborative design of featurebased parts through the internet*. ABCM Symposium Series in Mechatronics, 3, 701-710.

[18] Capers, J. (1991). *Applied Software Measurement: Assuring Productivity and Quality*. New York: McGraw-Hill.

[19] Capers, J. (1994). *Assessment and Control of Software Risks*. Englewood Cliffs, N.J.: Yourdon Press.

[20] Kitson, D. H. & Masters, S. (1993). *An Analysis of SEI Software Process Assessment Results, 1987-1991*. In Proceedings of the Fifteenth International Conference on Software Engineering (Washington, DC: IEEE Computer Society Press), 68-77.

[21] Jones, C., (1986). ed. *Tutorial: Programming Productivity: Issues for the Eighties*, 2nd Ed. (Los Angeles: IEEE Computer Society Press).

[22] Barry, B. W., & Papaccio, P. N. (1988). Understanding and Controlling Software Costs. *IEEE Transactions on Software Engineering, 14*, (10), 1462-1477.

[23] Gilb, T. (1988). *Principles of Software Engineering Management*. Wokingham, England: Addison-Wesley.

[24] Card, D. N. (1987). A Software Technology Evaluation Program. *Information And Software Technology, 29*(6), 291-300.

[25] Gilb, T. & Graham, D. (1993). *Software Inspection*. Wokingham, England: Addison-Wesley.

[26] Russell, G. W. (1991). Experience with Inspection in Ultralarge-Scale Developments. *IEEE Software, 8*(1), 25-31.

[27] Panchal, J. H. (2007). *An Adaptable Service-based Framework for Distributed Product Realization*. s.l. : Springer Series in Advanced Manufacturing, ISSN 1860-5168. ISBN 978-1-84628- 801-2 e-ISBN 978-1-84628-802-9.

[28] Fenves, S., Sriram, R., Choi, Y., Elm, J., & Robert, J. (2003, December). *Advanced Engineering Environments for Small Manufacturing Enterprises*: Volume I.. TECHNICAL REPORT, CMU/SEI-2003-TR-013, ESC-TR-2003-013.

[29] Xu, X. (2009). *Integrating Advanced Computer-Aided Design, Manufacturing, and Numerical Control: Principles and Implementations.* Auckland : Yurchak Printing Inc.

[30] Xu, X., Wang, H., Mao, J., Newman, S., Kramer, T., Proctor, F., & Michaloski, J. (2005). STEP-compliant NC research: the search for intelligent CAD/CAPP/CAM/CNC integration. *International Journal of Production Research, 43*(17), 3703-3743.

[31] Gielingh, W. (2008). An assessment of the current state of product data technologies. *Computer-Aided Design, 40*, 750-755.

[32] Peng, T. K., & Trappery, A. J. C. (1998). A step toward STEP-compatible engineering data management: the data models of product structure and engineering changes. *Robotics and computer-integrated manufacturing, 14*, 89-100.

[33] Pratt, M. J., Anderson, B. D., & Rangerc, T. (2005). Towards the standardized exchange of parameterize feature-based CAD models. *Computer-Aided Design, 37*, 1251-1265.

[34] Kemmerer, S. J. (1999). *STEP: The Grand Experience*. Special Publication 939, 187 pages, CODEN: NSPUE2.

[35] Qin, S. F, & Wright, D. K. (2004). Incremental Simulation Modelling for Internet Collaborative Design. *Journal of Engineering Manufacture, 218*, 1009-1015.

[36] Nassehi, A., Newman, S. T., & Allen, R. D. (2006). The application of multi-agent systems for STEP-NC computer aided process planning of prismatic components. *International Journal of Machine Tools & Manufacture, 46*, 559-574.

[37] Lopez-Ortega, M. O., & Ramirez, M. (2005). A STEP-based manufacturing information system to share flexible manufacturing resources data. *Journal of Intelligent Manufacturing, 16*, 287-301.

[38] Lee, S. H. & Jeong, Y. S. (2006). A system integration framework through development of ISO 10303-based product model for steel bridges. *Automation in Construction, 15*, 212-228.

[39] Canciglieri, O. J., Favaretto, F., & Young, R. I. M. (2005). Information sharing using features technology to support multiple viewpoint design for manufacture. *Produto & Produção, 8*, 75-86.

[40] Pisarciuc, C. (2007, October). *Information management for manufacturing systems design*. International Conference on Economic Engineering and Manufacturing Systems, Brasov.

[41] Nguyen Van, T., Ferru, F., Guellec, P., & Yannou, B. (2007, July). *Engineering Data Management for extended enterprise - Context of the European VIVACE Project.* International Conference on Product Lifecycle Management, Bangalore, Kartanaka, India.

[42] Newman, S. T., & Nassehi, A. (2007). Universal Manufacturing Platform for CNC Machining. *Annals of the CIRP, 56*.

[43] Slansky, D. (2005). *Interoperability and Openness across PLM: Have We Finally Arrived?* ARC Strategies.

[44] Xu, X. W., Wang, L., & Rong, Y. (2006). ‚STEP-NC and Function Blocks for Interoperable Manufacturing. *IEEE Transactions On Automation Science And Engineering, 3*(3).

[45] Burkett, W. C. (2001). Product data markup language: a new paradigm. *Computer-Aided Design, 33*, 489-500.

[46] Başak, H., & Gülesin, M. (2004). A Feature based parametric design program and expert system for design. *Mathematical and Computational Applications, 9*(3), 359-370.

[47] Pratt, M., Anderson, B., & Ranger, T. (2005). Towards the standardized exchange of parameterized feature-based CAD models., *Computer-Aided Design*, *37(*12), 1251-1265.

[48] Lockett, H. (2005). *A knowledge based manufacturing based manufacturing advisor for CAD* (PhD Thesis, Cranfield University).

[49] Sun, Q., & Gramoll, K. (2002). Internet-based Distributed Collaborative Engineering Analysis. *Concurrent Engineering Research and Applications*, *10*(4), 341-348.

[50] Bianconi, F., Conii, P., & Angelo, L. (2006). Interoperability among CAD/CAM/CAE systems: a review of current research trends. Geometric Modeling and Imaging - New Trends.

[51] Sarigecili, M. I., Baysal M. M., Roy, U. (2007). *An Evauation mechanism for defining gaps and overlaps of product information exchange standards*. Proceedings of International Mechanical Engineering Congress and Exposition, Seattle, Washington, USA.

[52] Markson, H. (2008). *Achieving CAD Interoperability in Global Product Design Environments*. Retrieved March 1. 2008, from http://www2.spaceclaim.com/ LearnMoreNow/WhitePapers.aspx.

[53] Gerbino, S. (2003, June). *Tools For The Interoperability Among Cad Systems.* International Conference On Tools And Methods Evolution In Engineering Design, Cassino, Napoli, Salerno.

[54] Moalla, N., Chettaoui, H., Ouzrout, Y., Noel, F., & Bouras, A. (2009). *Model-Driven Architecture to enhance interoperability between product applications.* International Conference on Product Lifecycle Management, Séoul, Korea, Republic Of.

[55] Yildiz, Y., Korkut, İ., & Şeker, U. (2006). Development of a Feature Based CAM System for Rotational Parts. *G.U. Journal of Science, 19*, 35-40.

[56] Liu, X., & Raorane, S. (2007). *A Web-based Intelligent Collaborative System for Engineering Design.* ISSN 1860-5168, s.l. : springer, 2007, Springer Series in Advanced Manufacturing, p. 37. ISBN 978-1-84628-801-2 e-ISBN 978-1-84628-802-9.

[57] J. Kim, & Han, S. (2004). Manipulating Geometry in a STEP DB from Commercial CAD Systems. *Concurrent Engineering : R&A, 12*(1), 49-57.

[58] Álvares, A. J., Carlos, J., & Ferreira, E. (2008). *WEB CAD BY FEATURES: Collaborative design of featurebased parts through the internet.* ABCM Symposium Series in Mechatronics, 3, 701-710.