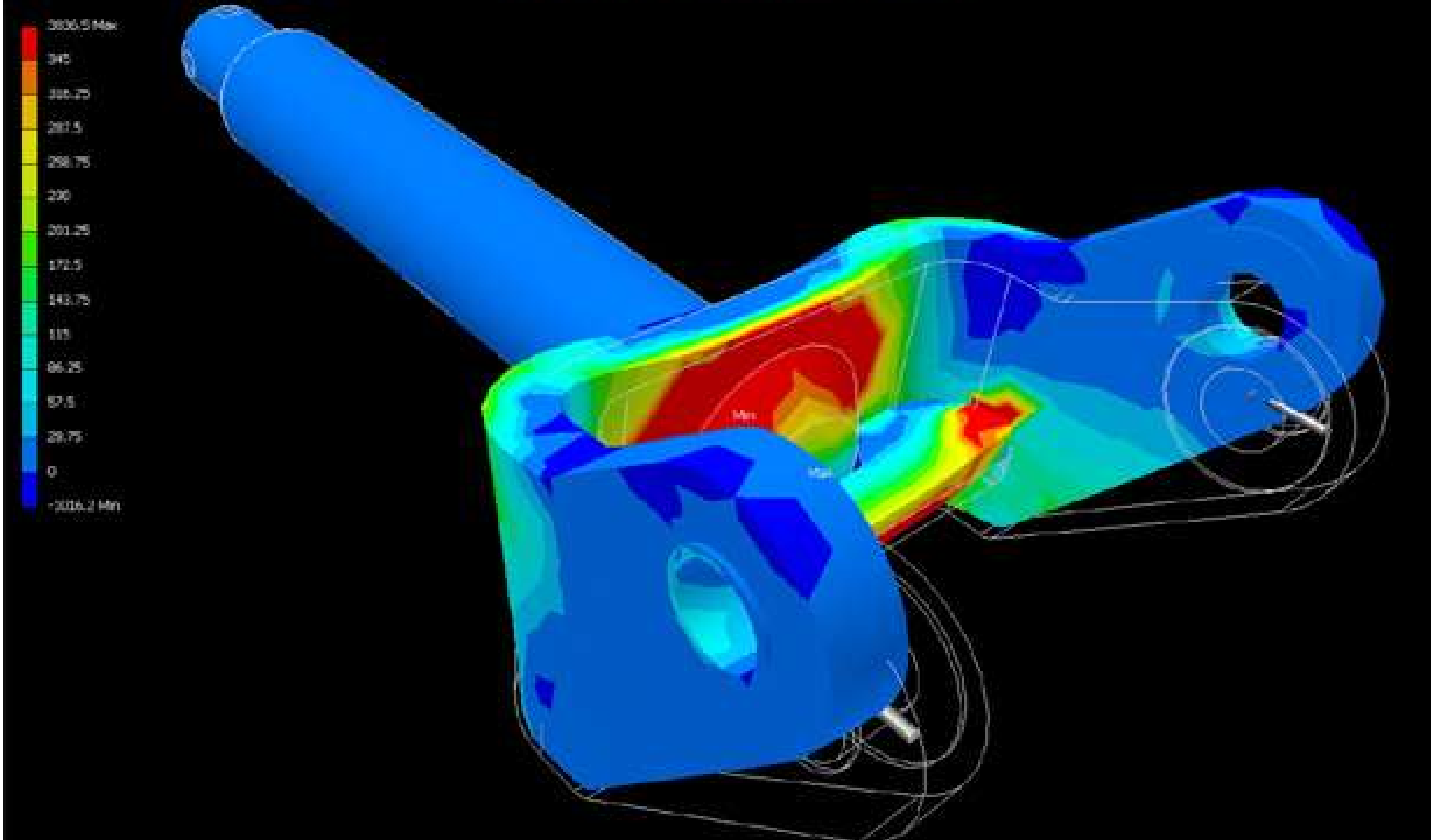
Handbook of Computer-Aided Technologies

Rylie Schubert



First Edition, 2012

ISBN 978-81-323-3570-2

Published by:
University Publications
4735/22 Prakashdeep Bldg,
Ansari Road, Darya Ganj,
Delhi - 110002
Email: info@wtbooks.com

Table of Contents

Chapter- 1

Introduction of Computer-Aided Technologies

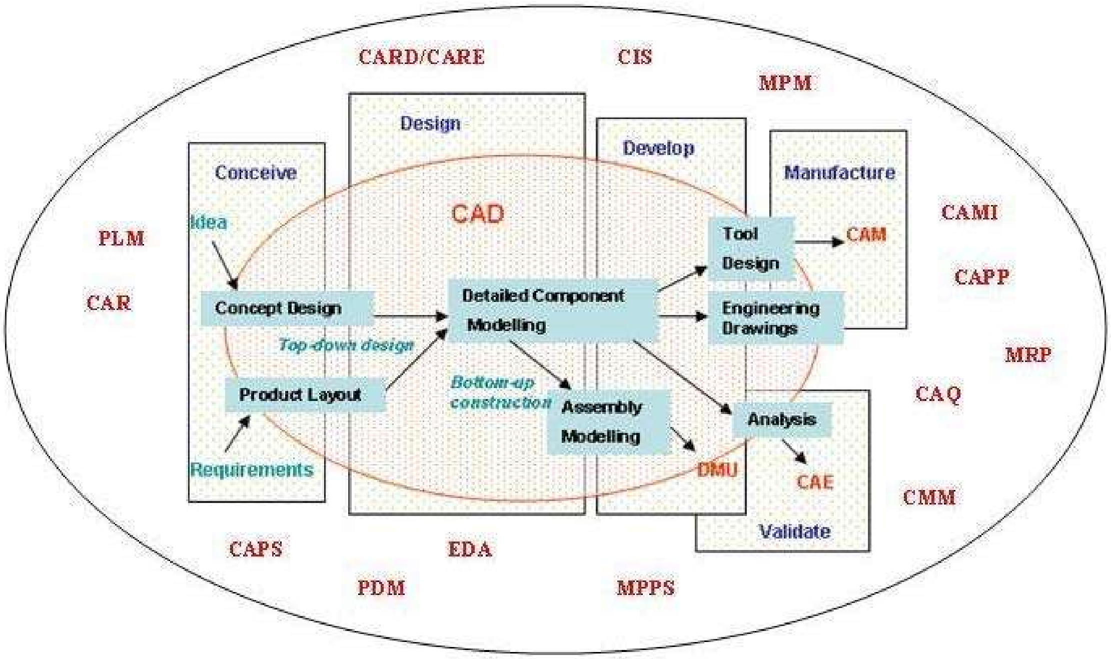
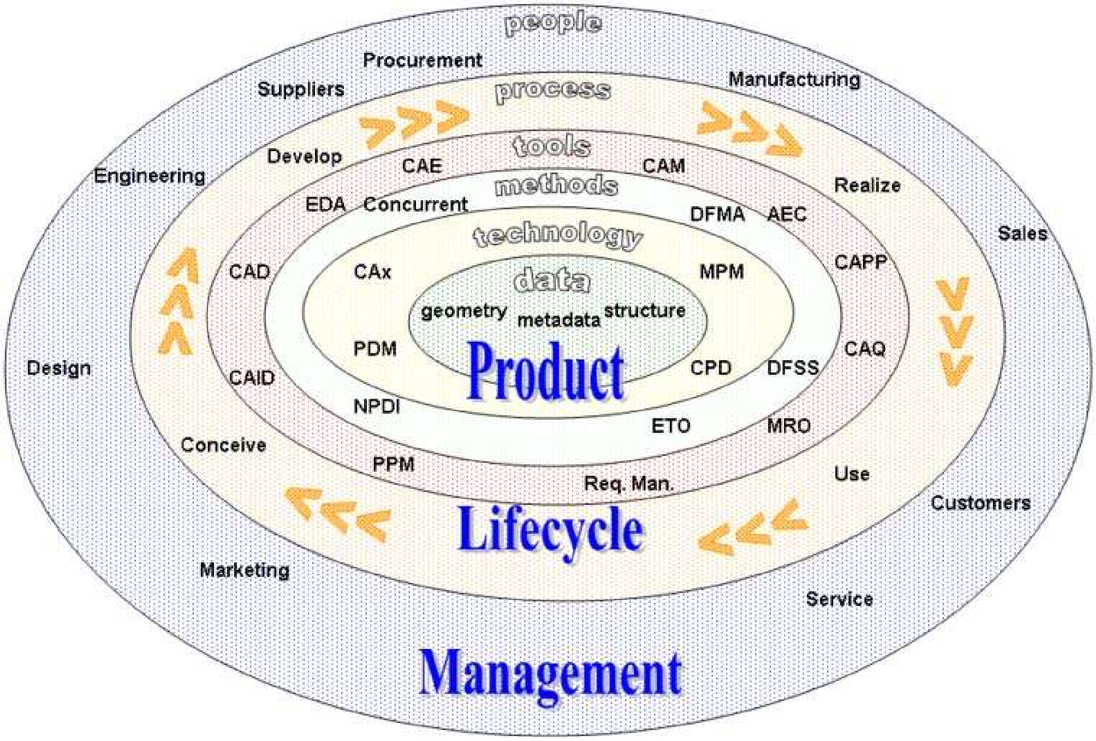


Illustration of the interaction of the various computer-aided technologies.



CAx tools in the context of product lifecycle management

Computer-aided technologies (**CAx**) is a broad term describing the use of computer technology to aid in the design, analysis, and manufacture of products.

Advanced CAx tools merge many different aspects of the product lifecycle management (PLM), including design, analysis using finite element analysis (FEA), manufacturing, production planning, product testing using virtual lab models and visualization, product documentation, product support, etc. CAx encompasses a broad range of tools, both those commercially available and those which are proprietary to the engineering firm.

The term **CAD/CAM** (computer-aided design and computer-aided manufacturing) is also often used in the context of a software tool covering a number of engineering functions.

Specimen of Computer-aided Technology: -

Computer-aided design

Computer-aided design (**CAD**), also known as **computer-aided design and drafting** (**CADD**), is the use of computer technology for the process of design and design-documentation. Computer Aided Drafting describes the process of drafting with a computer. CADD software, or environments, provide the user with input-tools for the purpose of streamlining design processes; drafting, documentation, and manufacturing processes. CADD output is often in the form of electronic files for print or machining operations. The development of CADD-based software is in direct correlation with the processes it seeks to economize; industry-based software (construction, manufacturing, etc.) typically uses vector-based (linear) environments whereas graphic-based software utilizes raster-based (pixelated) environments.

CADD environments often involve more than just shapes. As in the manual drafting of technical and engineering drawings, the output of CAD must convey information, such as materials, processes, dimensions, and tolerances, according to application-specific conventions.

CAD may be used to design curves and figures in two-dimensional (2D) space; or curves, surfaces, and solids in three-dimensional (3D) objects.

CAD is an important industrial art extensively used in many applications, including automotive, shipbuilding, and aerospace industries, industrial and architectural design, prosthetics, and many more. CAD is also widely used to produce computer animation for special effects in movies, advertising and technical manuals. The modern ubiquity and power of computers means that even perfume bottles and shampoo dispensers are designed using techniques unheard of by engineers of the 1960s. Because of its enormous economic importance, CAD has been a major driving force for research in computational geometry, computer graphics (both hardware and software), and discrete differential geometry.

The design of geometric models for object shapes, in particular, is often called *computer-aided geometric design* (*CAGD*).

Overview

Current computer-aided design software packages range from 2D vector-based drafting systems to 3D solid and surface modellers. Modern CAD packages can also frequently allow rotations in three dimensions, allowing viewing of a designed object from any desired angle, even from the inside looking out. Some CAD software is capable of dynamic mathematic modeling, in which case it may be marketed as **CADD** — *computer-aided design and drafting*.

CAD is used in the design of tools and machinery and in the drafting and design of all types of buildings, from small residential types (houses) to the largest commercial and industrial structures (hospitals and factories).

CAD is mainly used for detailed engineering of 3D models and/or 2D drawings of physical components, but it is also used throughout the engineering process from conceptual design and layout of products, through strength and dynamic analysis of assemblies to definition of manufacturing methods of components. It can also be used to design objects.

CAD has become an especially important technology within the scope of computer-aided technologies, with benefits such as lower product development costs and a greatly shortened design cycle. CAD enables designers to lay out and develop work on screen, print it out and save it for future editing, saving time on their drawings.

Uses

Computer-aided design is one of the many tools used by engineers and designers and is used in many ways depending on the profession of the user and the type of software in question.

CAD is one part of the whole Digital Product Development (DPD) activity within the Product Lifecycle Management (PLM) process, and as such is used together with other tools, which are either integrated modules or stand-alone products, such as:

- Computer-aided engineering (CAE) and Finite element analysis (FEA)
- Computer-aided manufacturing (CAM) including instructions to Computer Numerical Control (CNC) machines
- Photo realistic rendering
- Document management and revision control using Product Data Management (PDM).

CAD is also used for the accurate creation of photo simulations that are often required in the preparation of Environmental Impact Reports, in which computer-aided designs of intended buildings are superimposed into photographs of existing environments to represent what that locale will be like were the proposed facilities allowed to be built. Potential blockage of view corridors and shadow studies are also frequently analyzed through the use of CAD.

Types

There are several different types of CAD. Each of these different types of CAD systems require the operator to think differently about how he or she will use them and he or she must design their virtual components in a different manner for each.

There are many producers of the lower-end 2D systems, including a number of free and open source programs. These provide an approach to the drawing process without all the fuss over scale and placement on the drawing sheet that accompanied hand drafting, since these can be adjusted as required during the creation of the final draft.

3D wireframe is basically an extension of 2D drafting. Each line has to be manually inserted into the drawing. The final product has no mass properties associated with it and cannot have features directly added to it, such as holes. The operator approaches these in a similar fashion to the 2D systems, although many 3D systems allow using the wireframe model to make the final engineering drawing views.

3D "dumb" solids (programs incorporating this technology include AutoCAD) are created in a way analogous to manipulations of real world objects. Basic three-dimensional geometric forms (prisms, cylinders, spheres, and so on) have solid volumes added or subtracted from them, as if assembling or cutting real-world objects. Two-dimensional projected views can easily be generated from the models. Basic 3D solids don't usually include tools to easily allow motion of components, set limits to their motion, or identify interference between components.

3D parametric solid modeling require the operator to use what is referred to as "design intent". The objects and features created are adjustable. Any future modifications will be simple, difficult, or nearly impossible, depending on how the original part was created. One must think of this as being a "perfect world" representation of the component. If a feature was intended to be located from the center of the part, the operator needs to locate it from the center of the model, not, perhaps, from a more convenient edge or an arbitrary point, as he could when using "dumb" solids. Parametric solids require the operator to consider the consequences of his actions carefully.

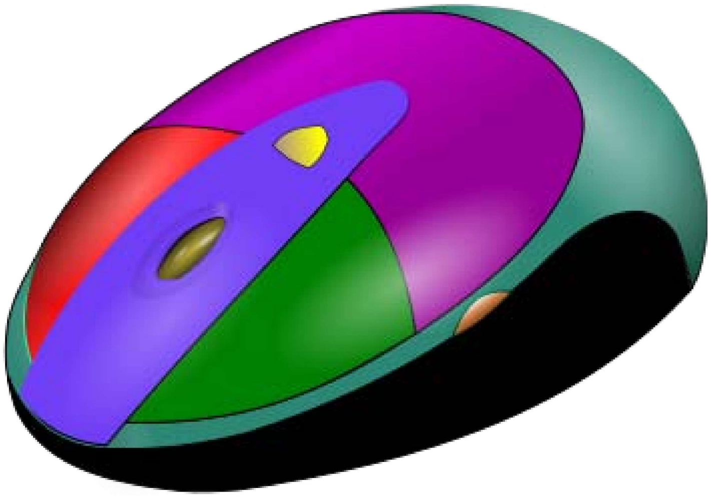
Some software packages provide the ability to edit parametric and non-parametric geometry without the need to understand or undo the design intent history of the geometry by use of direct modeling functionality. This ability may also include the additional ability to infer the correct relationships between selected geometry (e.g., tangency, concentricity) which makes the editing process less time and labor intensive while still freeing the engineer from the burden of understanding the model’s design intent history. These kind of non history based systems are called Explicit Modellers or Direct CAD Modelers. The first Explicit Modeling system was introduced to the world at the end of 80's by Hewlett-Packard under the name SolidDesigner.

Draft views are able to be generated easily from the models. Assemblies usually incorporate tools to represent the motions of components, set their limits, and identify interference. The tool kits available for these systems are ever increasing; including 3D piping and injection mold designing packages.

Mid range software are integrating parametric solids more easily to the end user: integrating more intuitive functions (SketchUp), using the best of both 3D dumb solids and parametric characteristics (VectorWorks), making very real-view scenes in relative few steps (Cinema4D) or offering all-in-one (form•Z).

Top end systems offer the capabilities to incorporate more organic, aesthetics and ergonomic features into designs (Catia, GenerativeComponents). Freeform surface modelling is often combined with solids to allow the designer to create products that fit the human form and visual requirements as well as they interface with the machine.

Technology



A CAD model of a mouse.

Originally software for Computer-Aided Design systems was developed with computer languages such as Fortran, but with the advancement of object-oriented programming methods this has radically changed. Typical modern parametric feature based modeler and freeform surface systems are built around a number of key C modules with their own APIs. A CAD system can be seen as built up from the interaction of a graphical user interface (GUI) with NURBS geometry and/or boundary representation (B-rep) data via a geometric modeling kernel. A geometry constraint engine may also be employed to manage the associative relationships between geometry, such as wireframe geometry in a sketch or components in an assembly.

Unexpected capabilities of these associative relationships have led to a new form of prototyping called digital prototyping. In contrast to physical prototypes, which entail manufacturing time in the design.

Today, CAD systems exist for all the major platforms (Windows, Linux, UNIX and Mac OS X); some packages even support multiple platforms.

Right now, no special hardware is required for most CAD software. However, some CAD systems can do graphically and computationally expensive tasks, so good graphics card, high speed (and possibly multiple) CPUs and large amounts of RAM are recommended.

The human-machine interface is generally via a computer mouse but can also be via a pen and digitizing graphics tablet. Manipulation of the view of the model on the screen is also sometimes done with the use of a spacemouse/SpaceBall. Some systems also support stereoscopic glasses for viewing the 3D model.

Effects

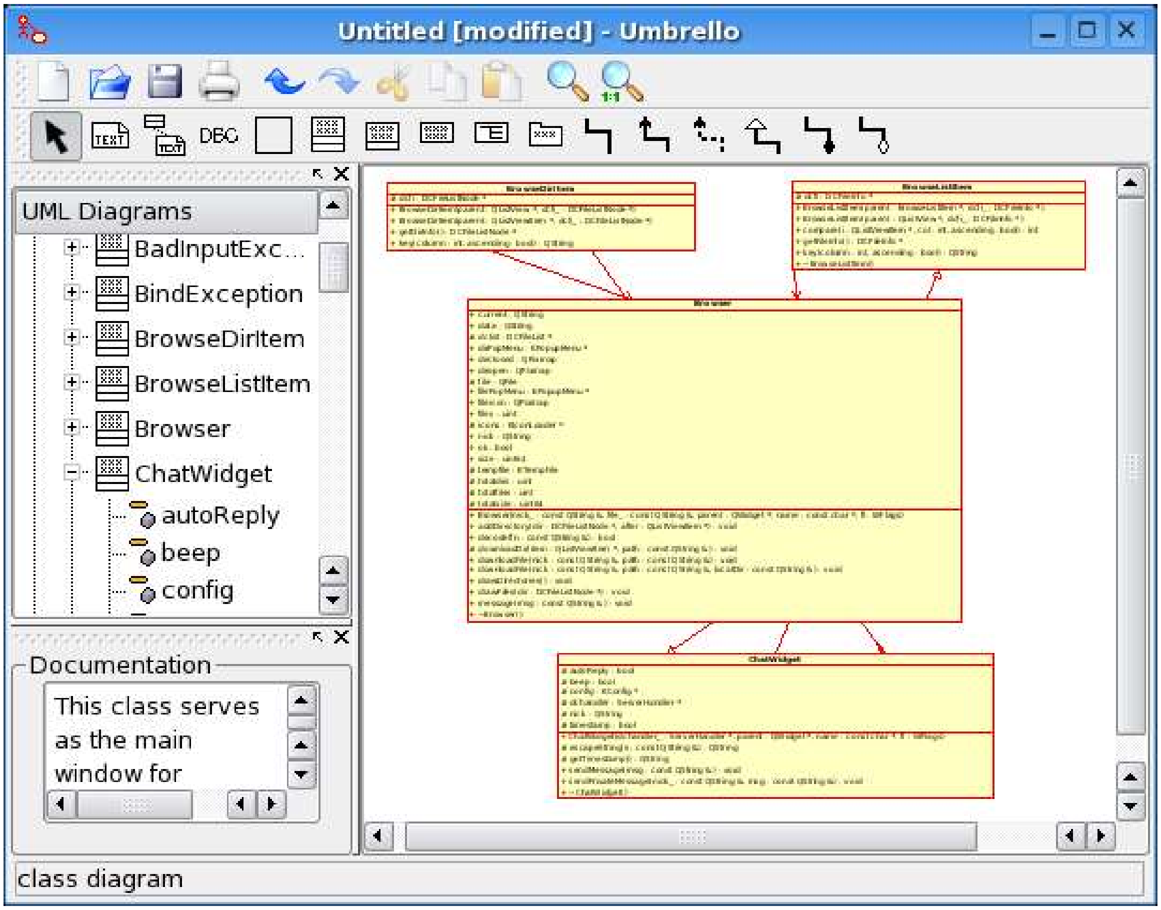
Beginning in the 1980s Computer-Aided Design programs reduced the need of draftsmen significantly, especially in small to mid-sized companies. Their affordability and ability to run on personal computers also allowed engineers to do their own drafting work, eliminating the need for entire departments. In today's world most, if not all, students in universities do not learn drafting techniques because they are not required to do so. The days of hand drawing for final drawings are almost obsolete. Universities such as New Jersey Institute of Technology no longer require the use of protractors and compasses to create drawings, instead there are several classes that focus on the use of CAD software such as Pro Engineer or IDEAS-MS.

Another consequence had been that since the latest advances were often quite expensive, small and even mid-size firms often could not compete against large firms who could use their computational edge for competitive purposes. Today, however, hardware and software costs have come down. Even high-end packages work on less expensive platforms and some even support multiple platforms. The costs associated with CAD implementation now are more heavily weighted to the costs of training in the use of these high level tools, the cost of integrating a CAD/CAM/CAE PLM using enterprise across multi-CAD and multi-platform environments and the costs of modifying design work flows to exploit the full advantage of CAD tools. CAD vendors have effectively lowered these training costs. These methods can be split into three categories:

1. Improved and simplified user interfaces. This includes the availability of “role” specific tailorable user interfaces through which commands are presented to users in a form appropriate to their function and expertise.
2. Enhancements to application software. One such example is improved design-in-context, through the ability to model/edit a design component from within the context of a large, even multi-CAD, active digital mockup.
3. User oriented modeling options. This includes the ability to free the user from the need to understand the design intent history of a complex intelligent model.

Chapter- 2

Computer-Aided Software Engineering



This class serves as the main window for

Example of a CASE tool.

Computer-aided software engineering (**CASE**) is the scientific application of a set of tools and methods to a software system which is meant to result in high-quality, defect-free, and maintainable software products. It also refers to methods for the development of information systems together with automated tools that can be used in the software development process.

Overview

The term "Computer-aided software engineering" (CASE) can refer to the software used for the automated development of systems software, i.e., computer code. The CASE functions include analysis, design, and programming. CASE tools automate methods for designing, documenting, and producing structured computer code in the desired programming language.

Computer Aided Software Engineering (CASE) is the name given to the software used to support the software process activities such as requirement engineering, design, program development and testing. Therefore, CASE tools include design editors, data dictionaries, compilers, debuggers, system building tools, etc.

Computer Aided Software Engineering (CASE) refers to the methods dedicated to an engineering discipline for the development of information system using automated tools.

The case is mainly used for the developement of quality softwares which will perform effectively.

History

The ISDOS project at the University of Michigan initiated a great deal of interest in the whole concept of using computer systems to help analysts in the very difficult process of analysing requirements and developing systems. Several papers by Daniel Teichroew fired a whole generation of enthusiasts with the potential of automated systems development. His PSL/PSA tool was a CASE tool although it predated the term. His insights into the power of meta-meta-models was inspiring, particularly to a former student, Dr. Hasan Sayani, currently Professor, Program Director at University of Maryland University College.

Another major thread emerged as a logical extension to the DBMS directory. By extending the range of meta-data held, the attributes of an application could be held within a dictionary and used at runtime. This "active dictionary" became the precursor to the more modern "model driven execution" (MDE) capability. However, the active dictionary did not provide a graphical representation of any of the meta-data. It was the linking of the concept of a dictionary holding analysts' meta-data, as derived from the use of an integrated set of techniques, together with the graphical representation of such data that gave rise to the earlier versions of I-CASE.

The term CASE was originally coined by software company Nastec Corporation of Southfield, Michigan in 1982 with their original integrated graphics and text editor GraphiText, which also was the first microcomputer-based system to use hyperlinks to cross-reference text strings in documents—an early forerunner of today's web page link. GraphiText's successor product, DesignAid, was the first microprocessor-based tool to logically and semantically evaluate software and system design diagrams and build a data dictionary.

Under the direction of Albert F. Case, Jr. vice president for product management and consulting, and Vaughn Frick, director of product management, the DesignAid product suite was expanded to support analysis of a wide range of structured analysis and design methodologies, notably Ed Yourdon and Tom DeMarco, Chris Gane & Trish Sarson, Ward-Mellor (real-time) SA/SD and Warnier-Orr (data driven).

The next entrant into the market was Excelerator from Index Technology in Cambridge, Mass. While DesignAid ran on Convergent Technologies and later Burroughs Ngen networked microcomputers, Index launched Excelerator on the IBM PC/AT platform. While, at the time of launch, and for several years, the IBM platform did not support networking or a centralized database as did the Convergent Technologies or Burroughs machines, the allure of IBM was strong, and Excelerator came to prominence. Hot on the heels of Excelerator were a rash of offerings from companies such as Knowledgeware (James Martin, Fran Tarkenton and Don Addington), Texas Instrument's IEF and Accenture's FOUNDATION toolset (METHOD/1, DESIGN/1, INSTALL/1, FCP).

CASE tools were at their peak in the early 1990s. At the time IBM had proposed AD/Cycle, which was an alliance of software vendors centered around IBM's Software repository using IBM DB2 in mainframe and OS/2:

> *The application development tools can be from several sources: from IBM, from vendors, and from the customers themselves. IBM has entered into relationships with Bachman Information Systems, Index Technology Corporation, and Knowledgeware, Inc. wherein selected products from these vendors will be marketed through an IBM complementary marketing program to provide offerings that will help to achieve complete life-cycle coverage.*

With the decline of the mainframe, AD/Cycle and the Big CASE tools died off, opening the market for the mainstream CASE tools of today. Nearly all of the leaders of the CASE market of the early 1990s ended up being purchased by Computer Associates, including IEW, IEF, ADW, Cayenne, and Learmonth & Burchett Management Systems (LBMS).

Supporting software

Alfonso Fuggetta classified CASE into 3 categories:

1. *Tools* support only specific tasks in the software process.
2. *Workbenches* support only one or a few activities.
3. *Environments* support (a large part of) the software process.

Workbenches and environments are generally built as collections of tools. Tools can therefore be either stand alone products or components of workbenches and environments.

Tools

CASE tools are a class of software that automate many of the activities involved in various life cycle phases. For example, when establishing the functional requirements of a proposed application, prototyping tools can be used to develop graphic models of application screens to assist end users to visualize how an application will look after development. Subsequently, system designers can use automated design tools to transform the prototyped functional requirements into detailed design documents. Programmers can then use automated code generators to convert the design documents into code. Automated tools can be used collectively, as mentioned, or individually. For example, prototyping tools could be used to define application requirements that get passed to design technicians who convert the requirements into detailed designs in a traditional manner using flowcharts and narrative documents, without the assistance of automated design software.

Existing CASE tools can be classified along 4 different dimensions :

1. Life-Cycle Support
2. Integration Dimension
3. Construction Dimension
4. Knowledge Based CASE dimension

Let us take the meaning of these dimensions along with their examples one by one :

Life-Cycle Based CASE Tools

This dimension classifies CASE Tools on the basis of the activities they support in the information systems life cycle. They can be classified as Upper or Lower CASE tools.

- Upper CASE Tools: support strategic, planning and construction of conceptual level product and ignore the design aspect. They support traditional diagrammatic languages such as ER diagrams, Data flow diagram, Structure charts, Decision Trees, Decision tables, etc.
- Lower CASE Tools: concentrate on the back end activities of the software life cycle and hence support activities like physical design, debugging, construction, testing, integration of software components, maintenance, reengineering and reverse engineering activities.

Integration dimension

Three main CASE Integration dimensions have been proposed :

1. CASE Framework
2. ICASE Tools
3. Integrated Project Support Environment(IPSE)

Workbenches

Workbenches integrate several CASE tools into one application to support specific software-process activities. Hence they achieve:

- a homogeneous and consistent interface (presentation integration).
- easy invocation of tools and tool chains (control integration).
- access to a common data set managed in a centralized way (data integration).

CASE workbenches can be further classified into following 8 classes:

1. Business planning and modeling
2. Analysis and design
3. User-interface development
4. Programming
5. Verification and validation
6. Maintenance and reverse engineering
7. Configuration management
8. Project management

Environments

An environment is a collection of CASE tools and workbenches that supports the software process. CASE environments are classified based on the focus/basis of integration

1. Toolkits
2. Language-centered
3. Integrated
4. Fourth generation
5. Process-centered

Toolkits

Toolkits are loosely integrated collections of products easily extended by aggregating different tools and workbenches. Typically, the support provided by a toolkit is limited to programming, configuration management and project management. And the toolkit itself is environments extended from basic sets of operating system tools, for example, the Unix Programmer's Work Bench and the VMS VAX Set. In addition, toolkits' loose integration requires user to activate tools by explicit invocation or simple control mechanisms. The resulting files are unstructured and could be in different format, therefore the access of file from different tools may require explicit file format conversion. However, since the only constraint for adding a new component is the formats of the files, toolkits can be easily and incrementally extended.

Language-centered

The environment itself is written in the programming language for which it was developed, thus enabling users to reuse, customize and extend the environment. Integration of code in different languages is a major issue for language-centered environments. Lack of process and data integration is also a problem. The strengths of these environments include good level of presentation and control integration. Interlisp, Smalltalk, Rational, and KEE are examples of language-centered environments.

Integrated

These environments achieve presentation integration by providing uniform, consistent, and coherent tool and workbench interfaces. Data integration is achieved through the *repository* concept: they have a specialized database managing all information produced and accessed in the environment. Examples of integrated environment are IBM AD/Cycle and DEC Cohesion.

Fourth-generation

Fourth-generation environments were the first integrated environments. They are sets of tools and workbenches supporting the development of a specific class of program: electronic data processing and business-oriented applications. In general, they include programming tools, simple configuration management tools, document handling facilities and, sometimes, a code generator to produce code in lower level languages. Informix 4GL, and Focus fall into this category.

Process-centered

Environments in this category focus on process integration with other integration dimensions as starting points. A process-centered environment operates by interpreting a process model created by specialized tools. They usually consist of tools handling two functions:

- Process-model execution
- Process-model production

Examples are East, Enterprise II, Process Wise, Process Weaver, and Arcadia.

Applications

All aspects of the software development life cycle can be supported by software tools, and so the use of tools from across the spectrum can, arguably, be described as CASE;

from project management software through tools for business and functional analysis, system design, code storage, compilers, translation tools, test software, and so on.

However, tools that are concerned with analysis and design, and with using design information to create parts (or all) of the software product, are most frequently thought of as CASE tools. CASE applied, for instance, to a database software product, might normally involve:

- Modeling business / real-world processes and data flow
- Development of data models in the form of entity-relationship diagrams
- Development of process and function descriptions

Risks and associated controls

Common CASE risks and associated controls include:

- *Inadequate Standardization* : Linking CASE tools from different vendors (design tool from Company X, programming tool from Company Y) may be difficult if the products do not use standardized code structures and data classifications. File formats can be converted, but usually not economically. Controls include using tools from the same vendor, or using tools based on standard protocols and insisting on demonstrated compatibility. Additionally, if organizations obtain tools for only a portion of the development process, they should consider acquiring them from a vendor that has a full line of products to ensure future compatibility if they add more tools.

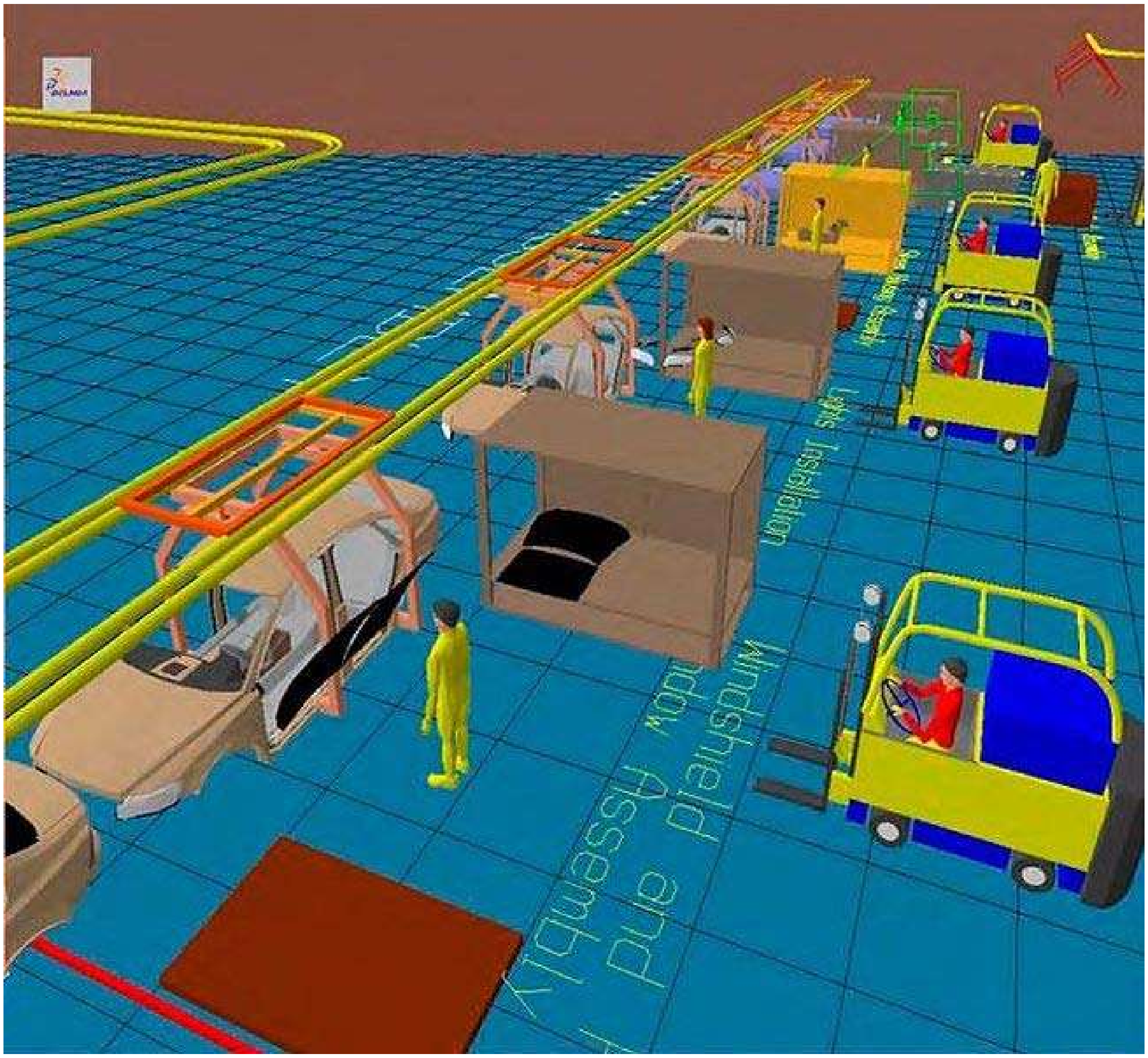
- *Unrealistic Expectations* : Organizations often implement CASE technologies to reduce development costs. Implementing CASE strategies usually involves high start-up costs. Generally, management must be willing to accept a long-term payback period. Controls include requiring senior managers to define their purpose and strategies for implementing CASE technologies.

- *Slow Implementation* : Implementing CASE technologies can involve a significant change from traditional development environments. Typically, organizations should not use CASE tools the first time on critical projects or projects with short deadlines because of the lengthy training process. Additionally, organizations should consider using the tools on smaller, less complex projects and gradually implementing the tools to allow more training time.

- *Weak Repository Controls* : Failure to adequately control access to CASE repositories may result in security breaches or damage to the work documents, system designs, or code modules stored in the repository. Controls include protecting the repositories with appropriate access, version, and backup controls.

Chapter- 3

Computer-Integrated Manufacturing



Manufacturing Systems Integration Program, NIST 2008.

Computer-integrated manufacturing (**CIM**) is the manufacturing approach of using computers to control the entire production process. This integration allows individual processes to exchange information with each other and initiate actions. Through the integration of computers, manufacturing can be faster and less error-prone, although the main advantage is the ability to create automated manufacturing processes. Typically CIM relies on closed-loop control processes, based on real-time input from sensors. It is also known as *flexible design and manufacturing*.

Overview

The term "computer-integrated manufacturing" is both a method of manufacturing and the name of a computer-automated system in which individual engineering, production, marketing, and support functions of a manufacturing enterprise are organized. In a CIM system functional areas such as design, analysis, planning, purchasing, cost accounting, inventory control, and distribution are linked through the computer with factory floor functions such as materials handling and management, providing direct control and monitoring of all the operations.

As a method of manufacturing, three components distinguish CIM from other manufacturing methodologies:

- Means for data storage, retrieval, manipulation and presentation;
- Mechanisms for sensing state and modifying processes;
- Algorithms for uniting the data processing component with the sensor/modification component.

CIM is an example of the implementation of information and communication technologies (ICTs) in manufacturing.

CIM implies that there are at least two computers exchanging information, e.g. the controller of an arm robot and a micro-controller of a CNC machine.

Some factors involved when considering a CIM implementation are the production volume, the experience of the company or personnel to make the integration, the level of the integration into the product itself and the integration of the production processes. CIM is most useful where a high level of ICT is used in the company or facility, such as CAD/CAM systems, the availability of process planning and its data.

History

The idea of "digital manufacturing" was prominent the 1980s, when computer-integrated manufacturing was developed and promoted by machine tool manufacturers and the Computer and Automated Systems Association and Society of Manufacturing Engineers (CASA/SME).

> "*CIM is the integration of total manufacturing enterprise by using integrated systems and data communication coupled with new managerial philosophies that improve organizational and personnel efficiency*." ERHUM

Computer-integrated manufacturing topics

Key challenges

There are three major challenges to development of a smoothly operating computer-integrated manufacturing system:

- Integration of components from different suppliers: When different machines, such as CNC, conveyors and robots, are using different communications protocols. In the case of AGVs, even differing lengths of time for charging the batteries may cause problems.

- Data integrity: The higher the degree of automation, the more critical is the integrity of the data used to control the machines. While the CIM system saves on labor of operating the machines, it requires extra human labor in ensuring that there are proper safeguards for the data signals that are used to control the machines.

- Process control: Computers may be used to *assist* the human operators of the manufacturing facility, but there must always be a competent engineer on hand to handle circumstances which could not be foreseen by the designers of the control software.

Subsystems in computer-integrated manufacturing

A computer-integrated manufacturing system is not the same as a *"lights-out" factory*, which would run completely independent of human intervention, although it is a big step in that direction. Part of the system involves flexible manufacturing, where the factory can be quickly modified to produce different products, or where the volume of products can be changed quickly with the aid of computers. Some or all of the following subsystems may be found in a CIM operation:

Computer-aided techniques:

- CAD (computer-aided design)
- CAE (computer-aided engineering)
- CAM (computer-aided manufacturing)
- CAPP (computer-aided process planning)
- CAQ (computer-aided quality assurance)
- PPC (production planning and control)
- ERP (enterprise resource planning)
- A business system integrated by a common database.

Devices and equipment required:

- CNC, Computer numerical controlled machine tools
- DNC, Direct numerical control machine tools
- PLCs, Programmable logic controllers
- Robotics
- Computers
- Software
- Controllers
- Networks
- Interfacing
- Monitoring equipment

Technologies:

- FMS, (flexible manufacturing system)
- ASRS, automated storage and retrieval system
- AGV, automated guided vehicle
- Robotics
- Automated conveyance systems

Others:

- Lean manufacturing

CIMOSA

CIMOSA (Computer Integrated Manufacturing Open System Architecture), is a 1990s European proposal for an open system architecture for CIM developed by the AMICE Consortium as a series of ESPRIT projects. The goal of CIMOSA was "to help companies to manage change and integrate their facilities and operations to face world wide competition. It provides a consistent architectural framework for both enterprise modeling and enterprise integration as required in CIM environments".

CIMOSA provides a solution for business integration with four types of products:

- The CIMOSA Enterprise Modeling Framework, which provides a reference architecture for enterprise architecture
- CIMOSA IIS, a standard for physical and application integration.
- CIMOSA Systems Life Cycle, is a life cycle model for CIM development and deployment.
- Inputs to standardization, basics for international standard development.

CIMOSA according to Vernadat (1996), coined the term business process and introduced the process-based approach for integrated enterprise modeling based on a cross-boundaries approach, which opposed to traditional function or activity-based approaches.

With CIMOSA also the concept of an "Open System Architecture" (OSA) for CIM was introduced, which was designed to be vendor-independent, and constructed with standardised CIM modules. Here to the OSA is "described in terms of their function, information, resource, and organizational aspects. This should be designed with structured engineering methods and made operational in a modular and evolutionary architecture for operational use".

Application

There are multiple areas of usage:

- In mechanical engineering
- In electronic design automation (printed circuit board (PCB) and integrated circuit design data for manufacturing)

Chapter- 4

Numerical Control

A CNC Turning Center.

Siemens CNC panel.

Numerical control (**NC**) refers to the automation of machine tools that are operated by abstractly programmed commands encoded on a storage medium, as opposed to manually controlled via handwheels or levers, or mechanically automated via cams alone. The first NC machines were built in the 1940s and '50s, based on existing tools that were modified with motors that moved the controls to follow points fed into the system on punched tape. These early servomechanisms were rapidly augmented with analog and digital computers, creating the modern **computer numerical controlled** (**CNC**) machine tools that have revolutionized the manufacturing process.

In modern CNC systems, end-to-end component design is highly automated using computer-aided design (CAD) and computer-aided manufacturing (CAM) programs. The programs produce a computer file that is interpreted to extract the commands needed to operate a particular machine via a postprocessor, and then loaded into the CNC machines for production. Since any particular component might require the use of a number of different tools-drills, saws, etc.-modern machines often combine multiple tools into a single "cell". In other cases, a number of different machines are used with an external controller and human or robotic operators that move the component from machine to machine. In either case, the complex series of steps needed to produce any part is highly automated and produces a part that closely matches the original CAD design.

History

Earlier forms of automation

Cams

The automation of machine tool control began in the 1800s with cams that "played" a machine tool in the way that cams had long been playing musical boxes or operating elaborate cuckoo clocks. Thomas Blanchard built his gun-stock-copying lathes (1820s-30s), and the work of people such as Christopher Miner Spencer developed the turret lathe into the screw machine (1870s). Cam-based automation had already reached a highly advanced state by World War I (1910s).

However, automation via cams is fundamentally different from numerical control because it cannot be abstractly programmed. Cams can encode information, but getting the information from the abstract level of an engineering drawing into the cam is a manual process that requires sculpting and/or machining and filing.

Various forms of abstractly programmable control had existed during the 1800s: those of the Jacquard loom, player pianos, and mechanical computers pioneered by Charles Babbage and others. These developments had the potential for convergence with the automation of machine tool control starting in that century, but the convergence did not happen until many decades later.

Tracer control

The application of hydraulics to cam-based automation resulted in tracing machines that used a stylus to trace a template, such as the enormous Pratt & Whitney "Keller Machine", which could copy templates several feet across. Another approach was "record and playback", pioneered at General Motors (GM) in the 1950s, which used a storage system to record the movements of a human machinist, and then play them back on demand. Analogous systems are common even today, notably the "teaching lathe" which gives new machinists a hands-on feel for the process. None of these were numerically programmable, however, and required a master machinist at some point in the process, because the "programming" was physical rather than numerical.

Servos and selsyns

One barrier to complete automation was the required tolerances of the machining process, which are routinely on the order of thousandths of an inch. Although connecting some sort of control to a storage device like punched cards was easy, ensuring that the controls were moved to the correct position with the required accuracy was another issue. The movement of the tool resulted in varying forces on the controls that would mean a linear input would not result in linear tool motion. The key development in this area was the introduction of the servomechanism, which produced highly accurate measurement information. Attaching two servos together produced a selsyn, where a remote servo's motions were accurately matched by another. Using a variety of mechanical or electrical systems, the output of the selsyns could be read to ensure proper movement had occurred (in other words, forming a closed-loop control system).

The first serious suggestion that selsyns could be used for machining control was made by Ernst F. W. Alexanderson, a Swedish immigrant to the U.S. working at General Electric (GE). Alexanderson had worked on the problem of torque amplification that allowed the small output of a mechanical computer to drive very large motors, which GE used as part of a larger gun laying system for US Navy ships. Like machining, gun laying requires very high accuracies, much less than a degree, and the forces during the motion of the gun turrets was non-linear. In November 1931 Alexanderson suggested to the Industrial Engineering Department that the same systems could be used to drive the inputs of machine tools, allowing it to follow the outline of a template without the strong physical contact needed by existing tools like the Keller Machine. He stated that it was a "matter of straight engineering development". However, the concept was ahead of its time from a business development perspective, and GE did not take the matter seriously until years later, when others had pioneered the field.

Parsons and the invention of NC

The birth of NC is generally credited to John T. Parsons, a machinist and salesman at his father's machining company, Parsons Corp.

In 1942 he was told that helicopters were going to be the "next big thing" by the former head of Ford Trimotor production, Bill Stout. He called Sikorsky Aircraft to inquire about possible work, and soon got a contract to build the wooden stringers in the rotor blades. After setting up production at a disused furniture factory and ramping up production, one of the blades failed and it was traced to the spar. As at least some of the problem appeared to stem from spot welding a metal collar on the stringer to the metal spar, so Parsons suggested a new method of attaching the stringers to the spar using adhesives, never before tried on an aircraft design.

But that development led Parsons to wonder about the possibility of using stamped metal stringers instead of wood, which would be much easier to make and stronger too. The stringers for the rotors were built to a design provided by Sikorsky, which was sent to them as a series of 17 points defining the outline. Parsons then had to "fill in" the dots with a french curve to generate an outline they could use as a template to build the jigs for the wooden versions. But how to make a tool able to cut metal with that shape was a much harder problem. Parsons went to visit Wright Field to see Frank Stulen, who was the head of the Rotary Ring Branch at the Propeller lab. During their conversation, Stulen concluded that Parsons didn't really know what he was talking about. Parsons realized this, and hired Stulen on the spot. Stulen started work on 1 April 1946 and hired three new engineers to join him.

Stulen's brother worked at Curtis Wright Propeller, and mentioned that they were using punched card calculators for engineering calculations. Stulen decided to adopt the idea to run stress calculations on the rotors, the first detailed automated calculations on helicopter rotors. When Parsons saw what Stulen was doing with the punched card machines, he asked him if they could be used to generate an outline with 200 points instead of the 17 they were given, and offset each point by the radius of the cutting tool on a mill. If you cut at each of those points, it would produce a relatively accurate cutout of the stringer even in hard steel, and it could easily be filed down to a smooth shape. The resulting tool would be useful as a template for stamping metal stringers. Stullen had no problem making such a program, and used it to produce large tables of numbers that would be taken onto the machine floor. Here, one operator read the numbers off the charts to two other operators, one on each of the X- and Y- axes, and they would move the cutting head to that point and make a cut. This was called the "by-the-numbers method".

At that point Parsons conceived of a fully automated tool. With enough points on the outline, no manual working would be needed at all, but with manual operation the time saved by having the part more closely match the outline was offset by the time needed to move the controls. If the machine's inputs were attached directly to the card reader, this delay, and any associated manual errors, would be removed and the number of points could be dramatically increased. Such a machine could repeatedly punch out perfectly accurate templates on command. But at the time he had no funds to develop these ideas.

When one of Parsons's salesmen was on a visit to Wright Field, he was told of the problems the newly-formed US Air Force was having with new jet designs. He asked if Parsons had anything to help them. Parsons showed Lockheed their idea of an automated mill, but they were uninterested. They had already decided to use 5-axis template copiers to produce the stringers, cutting from a metal template, and had ordered the expensive cutting machine already. But as Parsons noted:

Now just picture the situation for a minute. Lockheed had contracted to design a machine to make these wings. This machine had five axes of cutter movement, and each of these was tracer controlled using a template. Nobody was using my method of making templates, so just imagine what chance they were going to have of making an accurate airfoil shape with inaccurate templates.

Parsons worries soon came true, and Lockheed's protests that they could fix the problem eventually rang hollow. In 1949 the Air Force arranged funding for Parsons to build his machines on his own. Early work with Snyder Machine & Tool Corp proved that the system of directly driving the controls from motors failed to have the accuracy needed to set the machine for a perfectly smooth cut. Since the mechanical controls did not respond in a linear fashion, you couldn't simply drive it with a certain amount of power, because the differing forces would mean the same amount of power would not always produce the same amount of motion in the controls. No matter how many points you included, the outline would still be rough.

Enter MIT

This was not an impossible problem to solve, but would require some sort of feedback system, like a selsyn, to directly measure how far the controls had actually turned. Faced with the daunting task of building such a system, in the spring of 1949 Parsons turned to Gordon S. Brown's Servomechanisms Laboratory at MIT, which was a world leader in mechanical computing and feedback systems. During the war the Lab had built a number of complex motor-driven devices like the motorized gun turret systems for the Boeing B-29 Superfortress and the automatic tracking system for the SCR-584 radar. They were naturally suited to technological transfer into a prototype of Parsons's automated "by-the-numbers" machine.

The MIT team was led by William Pease assisted by James McDonough. They quickly concluded that Parsons's design could be greatly improved; if the machine did not simply cut *at* points A and B, but instead moved smoothly *between* the points, then not only would it make a perfectly smooth cut, but could do so with many fewer points - the mill could cut lines directly instead of having to define a large number of cutting points to "simulate" it. A three-way agreement was arranged between Parsons, MIT, and the Air Force, and the project officially ran from July 1949 to June 1950. The contract called for the construction of two "Card-a-matic Milling Machine"s, a prototype and a production system. Both to be handed to Parsons for attachment to one of their mills in order to develop a deliverable system for cutting stringers.

Instead, in 1950 MIT bought a surplus Cincinnati Milling Machine Company "Hydro-Tel" mill of their own and arranged a new contract directly with the Air Force that froze Parsons out of further development. Parsons would later comment that he "never dreamed that anybody as reputable as MIT would deliberately go ahead and take over my project." In spite of the development being handed to MIT, Parsons filed for a patent on "Motor Controlled Apparatus for Positioning Machine Tool" on 5 May 1952, sparking a filing by MIT for a "Numerical Control Servo-System" on 14 August 1952. Parsons received US Patent 2,820,187 on 14 January 1958, and the company sold an exclusive license to Bendix. IBM, Fujitsu and General Electric all took sub-licenses after having already started development of their own devices.

MIT's machine

MIT fit gears to the various handwheel inputs and drove them with roller chains connected to motors, one for each of the machine's three axes (X, Y, and Z). The associated controller consisted of five refrigerator-sized cabinets that, together, were almost as large as the mill they were connected to. Three of the cabinets contained the motor controllers, one controller for each motor, the other two the digital reading system.

Unlike Parsons's original punched card design, the MIT design used standard 7-track punch tape for input. Three of the tracks were used to control the different axes of the machine, while the other four encoded various control information. The tape was read in a cabinet that also housed six relay-based hardware registers, two for each axis. With every read operation the previously read point was copied into the "starting point" register, and the newly read one into the "ending point". The tape was read continually and the number in the register increased until a "stop" instruction was encountered, four holes in a line.

The final cabinet held a clock that sent pulses through the registers, compared them, and generated output pulses that interpolated between the points. For instance, if the points were far apart the output would have pulses with every clock cycle, whereas closely spaced points would only generate pulses after multiple clock cycles. The pulses are sent into a summing register in the motor controllers, counting up by the number of pulses every time they were received. The summing registers were connected to a digital to analog convertor that output increased power to the motors as the count in the registers increased.

The registers were decremented by encoders attached to the motors and the mill itself, which would reduce the count by one for every one degree of rotation. Once the second point was reached the pulses from the clock would stop, and the motors would eventually drive the mill to the encoded position. Each 1 degree rotation of the controls produced a 0.0005 inch movement of the cutting head. The programmer could control the speed of the cut by selecting points that were closer together for slow movements, or further apart for rapid ones.

The system was publicly demonstrated in September 1952, appearing in that month's *Scientific American*. MIT's system was an outstanding success by any technical measure, quickly making any complex cut with extremely high accuracy that could not easily be duplicated by hand. However, the system was terribly complex, including 250 vacuum tubes, 175 relays and numerous moving parts, reducing its reliability in a production setting. It was also very expensive, the total bill presented to the Air Force was $360,000.14, $2,641,727.63 in 2005 dollars. Between 1952 and 1956 the system was used to mill a number of one-off designs for various aviation firms, in order to study their potential economic impact.

Proliferation of NC

The Air Force funding for the project ran out in 1953, but development was picked up by the Giddings and Lewis Machine Tool Co. In 1955 many of the MIT team left to form Concord Controls, a commercial NC company with Giddings' backing, producing the Numericord controller. Numericord was similar to the MIT design, but replaced the punch tape with a magnetic tape reader that General Electric was working on. The tape contained a number of signals of different phases, which directly encoded the angle of the various controls. The tape was played at a constant speed in the controller, which set its half of the selsyn to the encoded angles while the remote side was attached to the machine controls. Designs were still encoded on paper tape, but the tapes were transferred to a reader/writer that converted them into magnetic form. The magtapes could then be used on any of the machines on the floor, where the controllers were greatly reduced in complexity. Developed to produce highly accurate dies for an aircraft skinning press, the Numericord "NC5" went into operation at G&L's plant at Fond du Lac, WI in 1955.

Monarch Machine Tool also developed an NC-controlled lathe, starting in 1952. They demonstrated their machine at the 1955 Chicago Machine Tool Show, along with a number of other vendors with punched card or paper tape machines that were either fully developed or in prototype form. These included Kearney & Trecker's Milwaukee-Matic II that could change its cutting tool under NC control, a common feature on modern machines.

A Boeing report noted that "numerical control has proved it can reduce costs, reduce lead times, improve quality, reduce tooling and increase productivity.” In spite of these developments, and glowing reviews from the few users, uptake of NC was relatively slow. As Parsons later noted:

The NC concept was so strange to manufacturers, and so slow to catch on, that the US Army itself finally had to build 120 NC machines and lease them to various manufacturers to begin popularizing its use.

In 1958 MIT published its report on the economics of NC. They concluded that the tools were competitive with human operators, but simply moved the time from the machining to the creation of the tapes. In *Forces of Production*, Noble claims that this was the whole point as far as the Air Force was concerned; moving the process off of the highly unionized factory floor and into the un-unionized white collar design office. The cultural context of the early 1950s, a second Red Scare with a widespread fear of a bomber gap and of domestic subversion, sheds light on this interpretation. It was strongly feared that the West would lose the defense production race to the Communists, and that syndicalist power was a path toward losing, either by "getting too soft" (less output, greater unit expense) or even by Communist sympathy and subversion within unions (arising from their common theme of empowering the working class).

CNC arrives

Many of the commands for the experimental parts were programmed "by hand" to produce the punch tapes that were used as input. During the development of Whirlwind, MIT's real-time computer, John Runyon coded a number of subroutines to produce these tapes under computer control. Users could enter a list of points and speeds, and the program would generate the punch tape. In one instance, this process reduced the time required to produce the instruction list and mill the part from 8 hours to 15 minutes. This led to a proposal to the Air Force to produce a generalized "programming" language for numerical control, which was accepted in June 1956.

Starting in September Ross and Pople outlined a language for machine control that was based on points and lines, developing this over several years into the APT programming language. In 1957 the Aircraft Industries Association (AIA) and Air Material Command at the Wright-Patterson Air Force Base joined with MIT to standardize this work and produce a fully computer-controlled NC system. On 25 February 1959 the combined team held a press conference showing the results, including a 3D machined aluminum ash tray that was handed out in the press kit.

Meanwhile, Patrick Hanratty was making similar developments at GE as part of their partnership with G&L on the Numericord. His language, PRONTO, beat APT into commercial use when it was released in 1958. Hanratty then went on to develop MICR magnetic ink characters that were used in cheque processing, before moving to General Motors to work on the groundbreaking DAC-1 CAD system.

APT was soon extended to include "real" curves in 2D-APT-II. With its release, MIT reduced its focus on CNC as it moved into CAD experiments. APT development was picked up with the AIA in San Diego, and in 1962, to Illinois Institute of Technology Research. Work on making APT an international standard started in 1963 under USASI X3.4.7, but many manufacturers of CNC machines had their own one-off additions (like PRONTO), so standardization was not completed until 1968, when there were 25 optional add-ins to the basic system.

Just as APT was being released in the early 1960s, a second generation of lower-cost transistorized computers was hitting the market that were able to process much larger volumes of information in production settings. This so lowered the cost of implementing a NC system that by the mid 1960s, APT runs accounted for a third of all computer time at large aviation firms.

CAD meets CNC

While the Servomechanisms Lab was in the process of developing their first mill, in 1953 MIT's Mechanical Engineering Department dropped the requirement that undergraduates take courses in drawing. The instructors formerly teaching these programs were merged into the Design Division, where an informal discussion of computerized design started. Meanwhile the Electronic Systems Laboratory, the newly rechristened Servomechanisms Laboratory, had been discussing whether or not design would ever start with paper diagrams in the future.

In January 1959, an informal meeting was held involving individuals from both the Electronic Systems Laboratory and the Mechanical Engineering Department's Design Division. Formal meetings followed in April and May, which resulted in the "Computer-Aided Design Project". In December 1959, the Air Force issued a one year contract to

ESL for $223,000 to fund the Project, including $20,800 earmarked for 104 hours of computer time at $200 per hour. This proved to be far too little for the ambitious program they had in mind, although their engineering calculation system, AED, was released in March 1965.

In 1959 General Motors started an experimental project to digitize, store and print the many design sketches being generated in the various GM design departments. When the basic concept demonstrated that it could work, they started the DAC-1 project with IBM to develop a production version. One part of the DAC project was the direct conversion of paper diagrams into 3D models, which were then converted into APT commands and cut on milling machines. In November 1963 a trunk lid design moved from 2D paper sketch to 3D clay prototype for the first time. With the exception of the initial sketch, the design-to-production loop had been closed.

Meanwhile MIT's offsite Lincoln Labs was building computers to test new transistorized designs. The ultimate goal was essentially a transistorized Whirlwind known as TX-2, but in order to test various circuit designs a smaller version known as TX-0 was built first. When construction of TX-2 started, time in TX-0 freed up and this led to a number of experiments involving interactive input and use of the machine's CRT display for graphics. Further development of these concepts led to Ivan Sutherland's groundbreaking Sketchpad program on the TX-2.

Sutherland moved to the University of Utah after his Sketchpad work, but it inspired other MIT graduates to attempt the first true CAD system. It was Electronic Drafting Machine (EDM), sold to Control Data and known as "Digigraphics", that Lockheed used to build production parts for the C-5 Galaxy, the first example of an end-to-end CAD/CNC production system.

By 1970 there were a wide variety of CAD firms including Intergraph, Applicon, Computervision, Auto-trol Technology, UGS Corp. and others, as well as large vendors like CDC and IBM.

Proliferation of CNC

The price of computer cycles fell drastically during the 1960s with the widespread introduction of useful minicomputers. Eventually it became less expensive to handle the motor control and feedback with a computer program than it was with dedicated servo systems. Small computers were dedicated to a single mill, placing the entire process in a small box. PDP-8's and Data General Nova computers were common in these roles. The introduction of the microprocessor in the 1970s further reduced the cost of implementation, and today almost all CNC machines use some form of microprocessor to handle all operations.

The introduction of lower-cost CNC machines radically changed the manufacturing industry. Curves are as easy to cut as straight lines, complex 3-D structures are relatively easy to produce, and the number of machining steps that required human action have been dramatically reduced. With the increased automation of manufacturing processes with CNC machining, considerable improvements in consistency and quality have been achieved with no strain on the operator. CNC automation reduced the frequency of errors and provided CNC operators with time to perform additional tasks. CNC automation also allows for more flexibility in the way parts are held in the manufacturing process and the time required to change the machine to produce different components.

During the early 1970s the Western economies were mired in slow economic growth and rising employment costs, and NC machines started to become more attractive. The major U.S. vendors were slow to respond to the demand for machines suitable for lower-cost NC systems, and into this void stepped the Germans. In 1979, sales of German machines surpassed the U.S. designs for the first time. This cycle quickly repeated itself, and by 1980 Japan had taken a leadership position, U.S. sales dropping all the time. Once sitting in the #1 position in terms of sales on a top-ten chart consisting entirely of U.S. companies in 1971, by 1987 Cincinnati Milacron was in 8th place on a chart heavily dominated by Japanese firms.

Many researchers have commented that the U.S. focus on high-end applications left them in an uncompetitive situation when the economic downturn in the early 1970s led to greatly increased demand for low-cost NC systems. Unlike the U.S. companies, who had focused on the highly profitable aerospace market, German and Japanese manufacturers targeted lower-profit segments from the start and were able to enter the low-cost markets much more easily.

As computing and networking evolved, so did direct numerical control (DNC). Its long-term coexistence with less networked variants of NC and CNC is explained by the fact that individual firms tend to stick with whatever is profitable, and their time and money for trying out alternatives is limited. This explains why machine tool models and tape storage media persist in grandfathered fashion even as the state of the art advances.

DIY, Hobby, and Personal CNC

Recent developments in small scale CNC have been enabled, in large part, by the Enhanced Machine Controller project from the National Institute of Standards and Technology (NIST), an agency of the Commerce Department of the United States government. EMC is a public domain program operating under the Linux operating system and working on PC based hardware. After the NIST project ended, development continued, leading to EMC2 which is licensed under the GNU General Public License and Lesser GNU General Public License (GPL and LGPL). Derivations of the original EMC software have also led to several proprietary PC based programs notably TurboCNC, and Mach3, as well as embedded systems based on proprietary hardware. The availability of these PC based control programs has led to the development of DIY CNC, allowing hobbyists to build their own using open source hardware designs. The same basic architecture has allowed manufacturers, such as Sherline and Taig, to produce turnkey lightweight desktop milling machines for hobbyists.

Eventually the homebrew architecture was fully commercialized and used to create larger machinery suitable for commercial and industrial applications. This class of equipment has been referred to as Personal CNC. Parallel to the evolution of personal computers, Personal CNC has its roots in EMC and PC based control, but has evolved to the point where it can replace larger conventional equipment in many instances. As with the Personal Computer, Personal CNC is characterized by equipment whose size, capabilities, and original sales price make it useful for individuals, and which is intended to be operated directly by an end user, often without professional training in CNC technology.

Today

Although modern data storage techniques have moved on from punch tape in almost every other role, tapes are still relatively common in CNC systems. This is because it was often easier to add a punch tape reader to a microprocessor controller than it was to re-write large libraries of tapes into a new format. One change that was implemented fairly widely was the switch from paper to mylar tapes, which are much more mechanically robust. Floppy disks, USB flash drives and local area networking have replaced the tapes to some degree, especially in larger environments that are highly integrated.

The proliferation of CNC led to the need for new CNC standards that were not encumbered by licensing or particular design concepts, like APT. A number of different "standards" proliferated for a time, often based around vector graphics markup languages supported by plotters. One such standard has since become very common, the "G-code" that was originally used on Gerber Scientific plotters and then adapted for CNC use. The file format became so widely used that it has been embodied in an EIA standard. In turn, while G-code is the predominant language used by CNC machines today, there is a push to supplant it with STEP-NC, a system that was deliberately designed for CNC, rather than grown from an existing plotter standard.

While G-code is the most common method of programming, some machine-tool/control manufacturers also have invented their own proprietary "conversational" methods of programming, trying to make it easier to program simple parts and make set-up and modifications at the machine easier (such as Mazak's Mazatrol and Hurco). These have met with varying success.

A more recent advancement in CNC interpreters is support of logical commands, known as parametric programming (also known as macro programming). Parametric programs include both device commands as well as a control language similar to BASIC. The programmer can make if/then/else statements, loops, subprogram calls, perform various arithmetic, and manipulate variables to create a large degree of freedom within one program. An entire product line of different sizes can be programmed using logic and simple math to create and scale an entire range of parts, or create a stock part that can be scaled to any size a customer demands.

Description

Modern CNC mills differ little in concept from the original model built at MIT in 1952. Mills typically consist of a table that moves in the X and Y axes, and a tool spindle that moves in the Z (depth). The position of the tool is driven by motors through a series of step-down gears in order to provide highly accurate movements, or in modern designs, direct-drive stepper motors. Closed-loop control is not mandatory today, as open-loop control works as long as the forces are kept small enough.

As the controller hardware evolved, the mills themselves also evolved. One change has been to enclose the entire mechanism in a large box as a safety measure, often with additional safety interlocks to ensure the operator is far enough from the working piece for safe operation. Most new CNC systems built today are completely electronically controlled.

CNC-like systems are now used for any process that can be described as a series of movements and operations. These include laser cutting, welding, friction stir welding, ultrasonic welding, flame and plasma cutting, bending, spinning, pinning, gluing, fabric cutting, sewing, tape and fiber placement, routing, picking and placing (PnP), and sawing.

Tools with CNC variants

- Drills
- EDMs
- Lathes
- Milling machines
- Wood routers
- Sheet metal works (Turret Punch)
- Wire bending machines
- Hot-wire foam cutters
- Plasma cuttings
- Water jet cutters
- Laser cutting
- Oxy-fuel
- Surface grinders
- Cylindrical grinders
- 3D Printing
- Induction hardening machines

Chapter- 5

Enterprise Resource Planning

An **Enterprise Resource Planning** (**ERP**) system is an integrated computer-based application used to manage internal and external resources, including tangible assets, financial resources, materials, and human resources. Its purpose is to facilitate the flow of information between all business functions inside the boundaries of the organization and manage the connections to outside stakeholders. Built on a centralized database and normally utilizing a common computing platform, ERP systems consolidate all business operations into a uniform and enterprise-wide system environment.

An ERP system can either reside on a centralized server or be distributed across modular hardware and software units that provide "services" and communicate on a local area network. The distributed design allows a business to assemble modules from different vendors without the need for the placement of multiple copies of complex and expensive computer systems in areas which will not use their full capacity.

Origin of the term

The initialism ERP was first employed by research and analysis firm Gartner Group in 1990 as an extension of MRP (Material Requirements Planning; later manufacturing resource planning) and CIM (Computer Integrated Manufacturing), and whilst not supplanting these terms, it has come to represent a larger whole. It came into use, as the creators of MRP software started to develop software applications beyond the manufacturing arena. However, this does not mean that ERP packages have typically been developed from a manufacturing core. Many of the major players started their development of an integrated package from other directions, such as accounting, maintenance and human resources management. ERP systems now attempt to cover all core functions of an enterprise, regardless of the organization's business or charter. These systems can now be found in non-manufacturing businesses, non-profit organizations and governments.

To be considered as an ERP system, a software package should have the following traits:

- An integrated system that operates in (next to) real time, without relying on periodic batch updates.
- A common database, that is accessed by all applications, preventing redundant data and multiple data definitions.
- A consistent look and feel throughout each module (sales, manufacturing, accounting etc.).
- The ability to access the system without specialist integration by the Information System (or IT) department.

Components / Modules

- Transactional Backbone
    - Financials
    - Distribution
    - Human Resources
    - Product lifecycle management

- Advanced Applications
    - Customer Relationship Management (CRM)
    - Supply chain management software
        - Purchasing
        - Manufacturing
        - Distribution
    - Warehouse Management System.

- Management Portal/Dashboard
    - Decision Support System

These modules can exist in a system or can be utilized in an ad-hoc fashion.

Commercial applications

Manufacturing

Engineering, bills of material, work orders, scheduling, capacity, workflow management, quality control, cost management, manufacturing process, manufacturing projects, manufacturing flow

Supply chain management

Order to cash, inventory, order entry, purchasing, product configurator, supply chain planning, supplier scheduling, inspection of goods, claim processing, commission calculation

Financials

General ledger, cash management, accounts payable, accounts receivable, fixed assets

Project management

Costing, billing, time and expense, performance units, activity management

Human resources
Human resources, payroll, training, time and attendance, rostering, benefits
Customer relationship management
Sales and marketing, commissions, service, customer contact, call-center support
Data services
Various "self-service" interfaces for customers, suppliers and/or employees
Access control
Management of user privileges for various processes

History

The term "Enterprise resource planning" originally derived from manufacturing resource planning (MRP II) that followed material requirements planning (MRP). MRP evolved into ERP when "routings" became a major part of the software architecture and a company's capacity planning activity also became a part of the standard software activity. ERP systems typically handle the manufacturing, logistics, distribution, inventory, shipping, invoicing, and accounting for a company. ERP software can aid in the control of many business activities, including sales, marketing, delivery, billing, production, inventory management, quality management, and human resource management.

ERP systems saw a large boost in sales in the 1990s as companies faced the Y2K problem in their legacy systems. Many companies took this opportunity to replace such information systems with ERP systems. This rapid growth in sales was followed by a slump in 1999, at which time most companies had already implemented their Y2K solution.

ERP systems are often incorrectly called *back office systems*, indicating that customers and the general public are not directly involved. This is contrasted with *front office systems* like customer relationship management (CRM) systems that deal directly with the customers, or the eBusiness systems such as eCommerce, eGovernment, eTelecom, and eFinance, or supplier relationship management (SRM) systems.

ERP systems are cross-functional and enterprise-wide. All functional departments that are involved in operations or production are integrated in one system. In addition to areas such as manufacturing, warehousing, logistics, and information technology, this typically includes accounting, human resources, marketing and strategic management.

ERP II, a term coined in the early 2000s, is often used to describe what would be the next generation of ERP software. This new generation of software is web-based and allows both employees and external resources (such as suppliers and customers) real-time access to the system's data.

EAS — Enterprise Application Suite is a new name for formerly developed ERP systems which include (almost) all segments of business using ordinary Internet browsers as thin clients.

Though traditionally ERP packages have been on-premise installations, ERP systems are now also available as Software as a Service.

Best practices are incorporated into most ERP vendor's software packages. When implementing an ERP system, organizations can choose between customizing the software or modifying their business processes to the "best practice" function delivered in the "out-of-the-box" version of the software.

Prior to ERP, software was developed to fit individual processes of an individual business. Due to the complexities of most ERP systems and the negative consequences of a failed ERP implementation, most vendors have included "Best Practices" into their software. These "Best Practices" are what the Vendor deems as the most efficient way to carry out a particular business process in an Integrated Enterprise-Wide system. A study conducted by Ludwigshafen University of Applied Science surveyed 192 companies and concluded that companies which implemented industry best practices decreased mission-critical project tasks such as configuration, documentation, testing and training. In addition, the use of best practices reduced over risk by 71% when compared to other software implementations.

The use of best practices can make complying with requirements such as IFRS, Sarbanes-Oxley, or Basel II easier. They can also help where the process is a commodity such as electronic funds transfer. This is because the procedure of capturing and reporting legislative or commodity content can be readily codified within the ERP software, and then replicated with confidence across multiple businesses who have the same business requirement.

Implementation

Businesses have a wide scope of applications and processes throughout their functional units, producing ERP software systems that are typically complex and usually impose significant changes on staff work practices. Implementing ERP software is typically too complex for in-house developers, lacking the required skills, so it is desirable and advisable to hire outside consultants who are professionally trained to implement these systems. This is typically the most cost-effective way. There are three types of services that may be employed - Consulting, Customization, and Support. The length of time to implement an ERP system depends on the size of the business, the number of modules, the extent of customization, the scope of the change, and the willingness of the customer to take ownership for the project. ERP systems are modular, so they don't all need be implemented at once. Implementation can be divided into various stages, or phase-ins. The typical project is about 14 months and requires around 150 consultants. A small project (e.g. a company of less than 100 staff) can be planned and delivered within 3–9 months; however, a large, multi-site or multi-country implementation can take years. The length of the implementations is closely tied to the amount of customization desired.

To implement ERP systems, companies often seek the help of an ERP vendor or a third-party consulting company. Consulting firms typically provide three areas of professional services: consulting, customization, and support. The client organization can also employ independent program management, business analysis, change management, and UAT specialists to ensure their business requirements remain a priority during implementation.

Data Migration

Data migration is one of the most important activities in determining the success of an ERP implementation. Since many decisions must be made before migration, a significant amount of planning must occur. Unfortunately, data migration is the last activity before the production phase of an ERP implementation, and therefore receives minimal attention due to time constraints. The following are steps of a data migration strategy that can help with the success of an ERP implementation:

1. Identify the data to be migrated
2. Determine the timing of data migration
3. Generate the data templates
4. Freeze the tools for data migration
5. Decide on migration-related setups
6. Decide on data archiving

Process preparation

ERP vendors have designed their systems around standard business processes, based upon best business practices. Different vendor(s) have different types of processes but they are all of a standard, modular nature. Firms that want to implement ERP systems are consequently forced to adapt their organizations to standardized processes as opposed to adapting the ERP package to the existing processes. Neglecting to map current business processes prior to starting ERP implementation is a main reason for failure of ERP projects. It is therefore crucial that organizations perform a thorough business process analysis before selecting an ERP vendor and setting off on the implementation track. This analysis should map out all present operational processes, enabling selection of an ERP vendor whose standard modules are most closely aligned with the established organization. Redesign can then be implemented to achieve further process congruence. Research indicates that the risk of business process mismatch is decreased by:

- linking each current organizational process to the organization's strategy;
- analyzing the effectiveness of each process in light of its current related business capability;
- understanding the automated solutions currently implemented.

ERP implementation is considerably more difficult (and politically charged) in organizations structured into nearly independent business units, each responsible for their own profit and loss, because they will each have different processes, business rules, data semantics, authorization hierarchies and decision centers. Solutions include requirements coordination negotiated by local change management professionals or, if this is not possible, federated implementation using loosely integrated instances (e.g. linked via Master data management) specifically configured and/or customized to meet local needs.

A disadvantage usually attributed to ERP is that business process redesign to fit the standardized ERP modules can lead to a loss of competitive advantage. While documented cases exist where this has indeed materialized, other cases show that, following thorough process preparation, ERP systems can actually increase sustainable competitive advantage.

Configuration

Configuring an ERP system is largely a matter of balancing the way you want the system to work with the way the system lets you work. Begin by deciding which modules to install, then adjust the system using configuration tables to achieve the best possible fit in working with your company's processes.

Modules — Most systems are modular simply for the flexibility of implementing some functions but not others. Some common modules, such as finance and accounting are adopted by nearly all companies implementing enterprise systems; others however such as human resource management are not needed by some companies and therefore not adopted. A service company for example will not likely need a module for manufacturing. Other times companies will not adopt a module because they already have their own proprietary system they believe to be superior. Generally speaking, the greater the number of modules selected, the greater the integration benefits, but also the increase in costs, risks and changes involved.

Configuration Tables – A configuration table enables a company to tailor a particular aspect of the system to the way it chooses to do business. For example, an organization can select the type of inventory accounting – FIFO or LIFO – it will employ, or whether it wants to recognize revenue by geographical unit, product line, or distribution channel.

So what happens when the options the system allows just aren't good enough? At this point a company has two choices, both of which are not ideal. It can re-write some of the enterprise system's code, or it can continue to use an existing system and build interfaces between it and the new enterprise system. Both options will add time and cost to the implementation process. Additionally they can dilute the system's integration benefits. The more customized the system becomes the less possible seamless communication between suppliers and customers.kli

Connectivity to Plant Floor Information

ERP systems connect to real-time data and transaction data (data accumulated into collections to deliver sets of information) in a variety of ways. These systems are typically configured by System Integrators, able to bring their unique knowledge on process, equipment and vendor solutions.

Direct Integration – ERP systems connectivity (communications to plant floor equipment) as part of their product offering. This requires the ERP system developers to offer specific support for the variety of plant floor equipment that they want to interface with. ERP Vendors must be expert in their own products, and connectivity to other vendor products, often those offered by competitors.

Relational Database (RDB) Integration – ERP systems connect to plant floor data sources through a Relational Database Staging Table. Plant floor systems will deposit the necessary information into a Relational Data Base. The ERP system will remove and use the information from the RDB Table. The benefit of RDB Staging is that ERP vendors do not need to get involved in the complexities of plant floor equipment integration. Connectivity becomes the responsibility of the System Integrator.

EATM (Enterprise Transaction Modules) – These devices have the ability to communicate directly with plant floor equipment and will transact data with the ERP system in methods best supported by the ERP system. Again, this can be through a staging table, Web Services, or through system specific business system APIs. The benefit of an EATM is that it offers a complete, off the shelf solution, minimizing long term costs and customization.

Custom Integrated Solutions – Many system integrators designs offer custom crafted solutions, created on a per instance basis to meet site and system requirements. There are a wide variety of communications drivers available for plant floor equipment and there are separate products that have the ability to log data to relational database tables. Standards exist within the industry to support interoperability between software products, the most widely known being OPC, managed by the OPC Foundation. Custom Integrated Solutions typically run on workstation or server class computers. These systems tend to have the highest level of initial integration cost, and can have a higher long term cost in terms on maintenance and reliability. Long term costs can be minimized through careful system testing and thorough documentation.

Consulting services

Many organizations do not have sufficient internal skills to implement an ERP project. This results in many organizations offering consulting services for ERP implementation. Typically, a consulting team is responsible for the entire ERP implementation including:

1. selecting
2. planning
3. training
4. testing
5. implementation
6. delivery

of any customized modules. Examples of customization includes creating processes and reports for compliance; additional product training; creation of process triggers and workflow; specialist advice to improve how the ERP is used in the business; system optimization; and assistance writing reports, complex data extracts or implementing Business Intelligence.

For most mid-sized companies, the cost of the implementation will range from around the list price of the ERP user licenses to up to twice this amount (depending on the level of customization required). Large companies, and especially those with multiple sites or countries, will often spend considerably more on the implementation than the cost of the user licenses—three to five times more is not uncommon for a multi-site implementation.

Unlike most single-purpose applications, ERP packages have historically included full source code and shipped with vendor-supported team IDEs for customizing and extending the delivered code. During the early years of ERP the guarantee of mature tools and support for extensive customization was an important sales argument when a potential customer was considering developing their own unique solution in-house, or assembling a cross-functional solution by integrating multiple "best of breed" applications.

"Core system" customization vs configuration

Increasingly, ERP vendors have tried to reduce the need for customization by providing built-in "configuration" tools to address most customers' needs for changing how the out-of-the-box core system works. Key differences between customization and configuration include:

- Customization is always optional, whereas some degree of configuration (e.g., setting up cost/profit centre structures, organisational trees, purchase approval rules, etc.) may be needed before the software will work at all.
- Configuration is available to all customers, whereas customization allows an individual customer to implement proprietary "market-beating" processes.
- Configuration changes tend to be recorded as entries in vendor-supplied data tables, whereas customization usually requires some element of programming and/or changes to table structures or views.
- The effect of configuration changes on the performance of the system is relatively predictable and is largely the responsibility of the ERP vendor. The effect of customization is unpredictable and may require time-consuming stress testing by the implementation team.
- Configuration changes are almost always guaranteed to survive upgrades to new software versions. Some customizations (e.g. code that uses pre-defined "hooks" that are called before/after displaying data screens) will survive upgrades, though they will still need to be retested. More extensive customizations (e.g. those involving changes to fundamental data structures) will be overwritten during upgrades and must be reimplemented manually.

By this analysis, customizing an ERP package can be unexpectedly expensive and complicated, and tends to delay delivery of the obvious benefits of an integrated system.

Nevertheless, customizing an ERP suite gives the scope to implement secret recipes for excellence in specific areas while ensuring that industry best practices are achieved in less sensitive areas.

Extensions

In this context, "Extensions" refers to ways that an ERP environment can be "extended" (supplemented) with third-party programs. It is technically easy to expose most ERP transactions to outside programs that do other things, e.g.:

- archiving, reporting and republishing (these are easiest to achieve, because they mainly address static data);
- performing transactional data captures, e.g. using scanners, tills or RFIDs (also relatively easy because they touch existing data).

However, because ERP applications typically contain sophisticated rules that control how data can be created or changed, such functions can be very difficult to implement.

Advantages

In the absence of an ERP system, a large manufacturer may find itself with many software applications that cannot communicate or interface effectively with one another. Tasks that need to interface with one another may involve:

- ERP systems connect the necessary software in order for accurate forecasting to be done. This allows inventory levels to be kept at maximum efficiency and the company to be more profitable.
- Integration among different functional areas to ensure proper communication, productivity and efficiency
- Design engineering (how to best make the product)
- Order tracking, from acceptance through fulfillment
- The revenue cycle, from invoice through cash receipt
- Managing inter-dependencies of complex processes bill of materials
- Tracking the three-way match between purchase orders (what was ordered), inventory receipts (what arrived), and costing (what the vendor invoiced)
- The accounting for all of these tasks: tracking the revenue, cost and profit at a granular level.

ERP systems centralize the data in one place. Benefits of this include:

- Eliminates the problem of synchronizing changes between multiple systems - consolidation of finance, marketing and sales, human resource, and manufacturing applications
- Permits control of business processes that cross functional boundaries

- Provides top-down view of the enterprise (no "islands of information"), real time information is available to management anywhere, anytime to make proper decisions.
- Reduces the risk of loss of sensitive data by consolidating multiple permissions and security models into a single structure.
- Shorten production lead time and delivery time
- Facilitating business learning, empowering, and building common visions

Some security features are included within an ERP system to protect against both outsider crime, such as industrial espionage, and insider crime, such as embezzlement. A data-tampering scenario, for example, might involve a disgruntled employee intentionally modifying prices to below-the-break-even point in order to attempt to interfere with the company's profit or other sabotage. ERP systems typically provide functionality for implementing internal controls to prevent actions of this kind. ERP vendors are also moving toward better integration with other kinds of information security tools.

Disadvantages

Problems with ERP systems are mainly due to inadequate investment in ongoing training for the involved IT personnel - including those implementing and testing changes - as well as a lack of corporate policy protecting the integrity of the data in the ERP systems and the ways in which it is used.

Disadvantages

- Customization of the ERP software is limited.
- Re-engineering of business processes to fit the "industry standard" prescribed by the ERP system may lead to a loss of competitive advantage.
- ERP systems can be very expensive. (This has led to a new category of "ERP light" solutions.)
- ERPs are often seen as too rigid and too difficult to adapt to the specific workflow and business process of some companies—this is cited as one of the main causes of their failure.
- Many of the integrated links need high accuracy in other applications to work effectively. A company can achieve minimum standards, then over time "dirty data" will reduce the reliability of some applications.
- Once a system is established, switching costs are very high for any partner (reducing flexibility and strategic control at the corporate level).
- The blurring of company boundaries can cause problems in accountability, lines of responsibility, and employee morale.
- Resistance in sharing sensitive internal information between departments can reduce the effectiveness of the software.
- Some large organizations may have multiple departments with separate, independent resources, missions, chains-of-command, etc., and consolidation into a single enterprise may yield limited benefits.

Chapter- 6

Product Data Management

Product data management (**PDM**) is the business function often within product lifecycle management that is responsible for the creation, management and publication of product data.

Introduction

Product data management **(PDM)** is the use of software or other tools to track and control data related to a particular product. The data tracked usually involves the technical specifications of the product, specifications for manufacture and development, and the types of materials that will be required to produce goods. The use of product data management allows a company to track the various costs associated with the creation and launch of a product. Product data management is part of product life cycle management, and is primarily used by engineers.

Within PDM the focus is on managing and tracking the creation, change and archive of all information related to a product. The information being stored and managed (on one or more file servers) will include engineering data such as Computer-aided design (CAD) models, drawings and their associated documents.

Product data management (PDM) serves as a central knowledge repository for process and product history, and promotes integration and data exchange among all business users who interact with products — including project managers, engineers, sales people, buyers, and quality assurance teams.

The central database will also manage metadata such as owner of a file and release status of the components. The package will: control check-in and check-out of the product data to multi-user; carry out engineering change management and release control on all versions/issues of components in a product; build and manipulate the product structure bill of materials (BOM) for assemblies; and assist in configurations management of product variants.

This enables automatic reports on product costs, etc. Furthermore, PDM enables companies producing complex products to spread product data into the entire PLM launch-process. This significantly enhances the effectiveness of the launch process.

Product data management is focused on capturing and maintaining information on products and/or services through its development and useful life. Typical information managed in the PDM module include

- Part number
- Part description
- Supplier/vendor
- Vendor part number and description
- Unit of measure
- Cost/price
- Schematic or CAD drawing
- Material data sheets

PDM Advantages:

- Track and manage all changes to product related data
- Accelerate return on investment with easy setup;
- Spend less time organizing and tracking design data;
- Improve productivity through reuse of product design data;
- Enhance collaboration.

History of PDM

PDM stems from traditional engineering design activities that created product drawings and schematics on paper and using CAD tools to create parts lists (Bills of Material structures - BOM). The PDM and BOM data is used in enterprise resource planning (ERP) systems to plan and coordinate all transactional operations of a company (sales order management, purchasing, cost accounting, logistics, etc.)

PDM is a subset of a larger concept of product lifecycle management (PLM). PLM encompasses the processes needed to launch new products (NPI), manage changes to existing products (ECN/ECO) and retire products at the end of their life (End of Life) .

Capabilities

PDM systems vary in their functionality, but some of their common capabilities are described below.

- Access control

Access control is a system which enables an authority to control access to areas and resources in a given physical facility or computer-based information system. An access control system, within the field of physical security, is generally seen as the second layer in the security of a physical structure.

Access control is, in reality, an everyday phenomenon. A lock on a car door is essentially a form of access control. A PIN on an ATM system at a bank is another means of access control. Bouncers standing in front of a night club is perhaps a more primitive mode of access control (given the evident lack of information technology involved). The possession of access control is of prime importance when persons seek to secure important, confidential, or sensitive information and equipment.

Item control or electronic key management is an area within (and possibly integrated with) an access control system which concerns the managing of possession and location of small assets or physical (mechanical) keys.

Physical access

Underground entrance to the New York City Subway system

Physical access by a person may be allowed depending on payment, authorization, etc. Also there may be one-way traffic of people. These can be enforced by personnel such as a border guard, a doorman, a ticket checker, etc., or with a device such as a turnstile. There may be fences to avoid circumventing this access control. An alternative of access control in the strict sense (physically controlling access itself) is a system of checking authorized presence, see e.g. Ticket controller (transportation). A variant is exit control, e.g. of a shop (checkout) or a country.

In physical security, the term access control refers to the practice of restricting entrance to a property, a building, or a room to authorized persons. Physical access control can be achieved by a human (a guard, bouncer, or receptionist), through mechanical means such as locks and keys, or through technological means such as access control systems like the Access control vestibule. Within these environments, physical key management may also be employed as a means of further managing and monitoring access to mechanically keyed areas or access to certain small assets.

Physical access control is a matter of who, where, and when. An access control system determines who is allowed to enter or exit, where they are allowed to exit or enter, and when they are allowed to enter or exit. Historically this was partially accomplished through keys and locks. When a door is locked only someone with a key can enter through the door depending on how the lock is configured. Mechanical locks and keys do not allow restriction of the key holder to specific times or dates. Mechanical locks and keys do not provide records of the key used on any specific door and the keys can be easily copied or transferred to an unauthorized person. When a mechanical key is lost or the key holder is no longer authorized to use the protected area, the locks must be re-keyed.

Electronic access control uses computers to solve the limitations of mechanical locks and keys. A wide range of credentials can be used to replace mechanical keys. The electronic access control system grants access based on the credential presented. When access is granted, the door is unlocked for a predetermined time and the transaction is recorded. When access is refused, the door remains locked and the attempted access is recorded. The system will also monitor the door and alarm if the door is forced open or held open too long after being unlocked.

Access control system operation

When a credential is presented to a reader, the reader sends the credential’s information, usually a number, to a control panel, a highly reliable processor. The control panel compares the credential's number to an access control list, grants or denies the presented request, and sends a transaction log to a database. When access is denied based on the access control list, the door remains locked. If there is a match between the credential and the access control list, the control panel operates a relay that in turn unlocks the door. The control panel also ignores a door open signal to prevent an alarm. Often the reader provides feedback, such as a flashing red LED for an access denied and a flashing green LED for an access granted.

The above description illustrates a single factor transaction. Credentials can be passed around, thus subverting the access control list. For example, Alice has access rights to the server room but Bob does not. Alice either gives Bob her credential or Bob takes it; he now has access to the server room. To prevent this, two-factor authentication can be used. In a two factor transaction, the presented credential and a second factor are needed for access to be granted; another factor can be a PIN, a second credential, operator intervention, or a biometric input.

There are three types (factors) of authenticating information:

- something the user knows, eg a password, pass-phrase or PIN
- something the user has, such as smart card
- something the user is, such as fingerprint, verified by biometric measurement

Passwords are a common means of verifying a user's identity before access is given to information systems. In addition, a fourth factor of authentication is now recognized: someone you know, where another person who knows you can provide a human element of authentication in situations where systems have been set up to allow for such scenarios. For example, a user may have their password, but have forgotten their smart card. In such a scenario, if the user is known to designated cohorts, the cohorts may provide their smart card and password in combination with the extant factor of the user in question and thus provide two factors for the user with missing credential, and three factors overall to allow access.

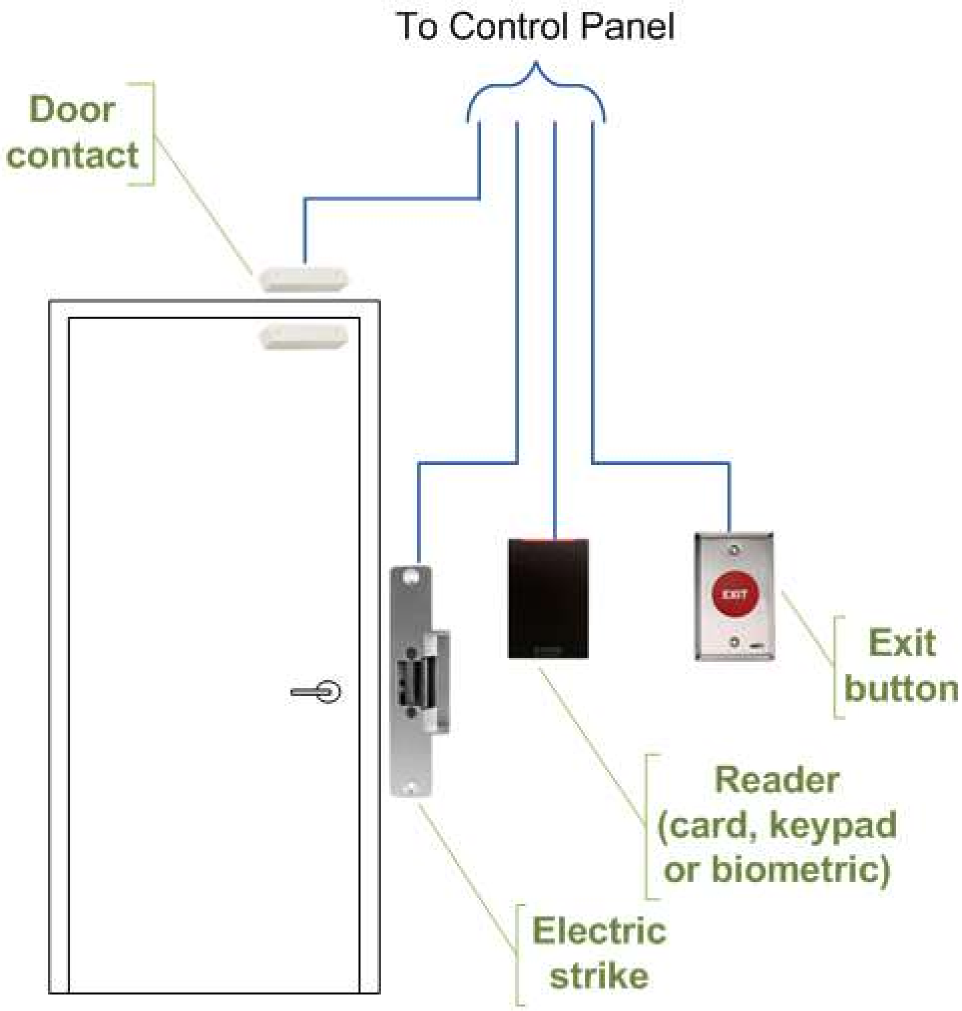
Credential

A credential is a physical/tangible object, a piece of knowledge, or a facet of a person's physical being, that enables an individual access to a given physical facility or computer-based information system. Typically, credentials can be something you know (such as number or PIN), something you have (such as an access badge), something you are (such as a biometric feature) or some combination of these items. The typical credential is an access card, key fob, or other key. There are many card technologies including magnetic stripe, bar code, Wiegand, 125 kHz proximity, 26 bit card-swipe, contact smart cards, and contactless smart cards. Also available are key-fobs which are more compact than ID cards and attach to a key ring. Typical biometric technologies include fingerprint, facial recognition, iris recognition, retinal scan, voice, and hand geometry.

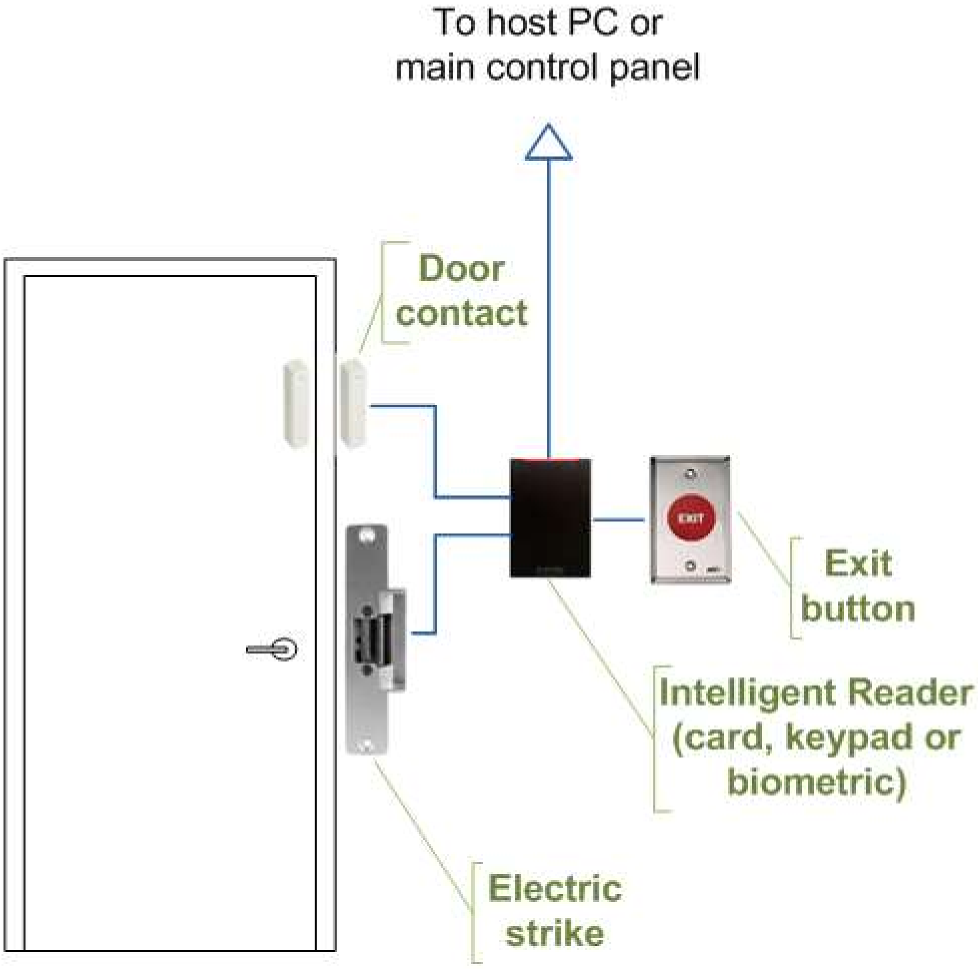
Access control system components

An access control point, which can be a door, turnstile, parking gate, elevator, or other physical barrier where granting access can be electronically controlled. Typically the access point is a door. An electronic access control door can contain several elements. At its most basic there is a stand-alone electric lock. The lock is unlocked by an operator with a switch. To automate this, operator intervention is replaced by a reader. The reader could be a keypad where a code is entered, it could be a card reader, or it could be a biometric reader. Readers do not usually make an access decision but send a card number to an access control panel that verifies the number against an access list. To monitor the door position a magnetic door switch is used. In concept the door switch is not unlike those on refrigerators or car doors. Generally only entry is controlled and exit is uncontrolled. In cases where exit is also controlled a second reader is used on the opposite side of the door. In cases where exit is not controlled, free exit, a device called a request-to-exit (RTE) is used. Request-to-exit devices can be a pushbutton or a motion detector. When the button is pushed or the motion detector detects motion at the door, the door alarm is temporarily ignored while the door is opened. Exiting a door without having to electrically unlock the door is called mechanical free egress. This is an important safety feature. In cases where the lock must be electrically unlocked on exit, the request-to-exit device also unlocks the door.

Access control topology



Typical access control door wiring


To host PC or main control panel
Door contact
EXIT
Exit button
Intelligent Reader (card, keypad or biometric)
Electric strike

Access control door wiring when using intelligent readers

Access control decisions are made by comparing the credential to an access control list. This lookup can be done by a host or server, by an access control panel, or by a reader. The development of access control systems has seen a steady push of the lookup out from a central host to the edge of the system, or the reader. The predominate topology circa 2009 is hub and spoke with a control panel as the hub and the readers as the spokes. The lookup and control functions are by the control panel. The spokes communicate through a serial connection; usually RS485. Some manufactures are pushing the decision making to the edge by placing a controller at the door. The controllers are IP enabled and connect to a host and database using standard networks.

Types of readers

Access control readers may be classified by functions they are able to perform:

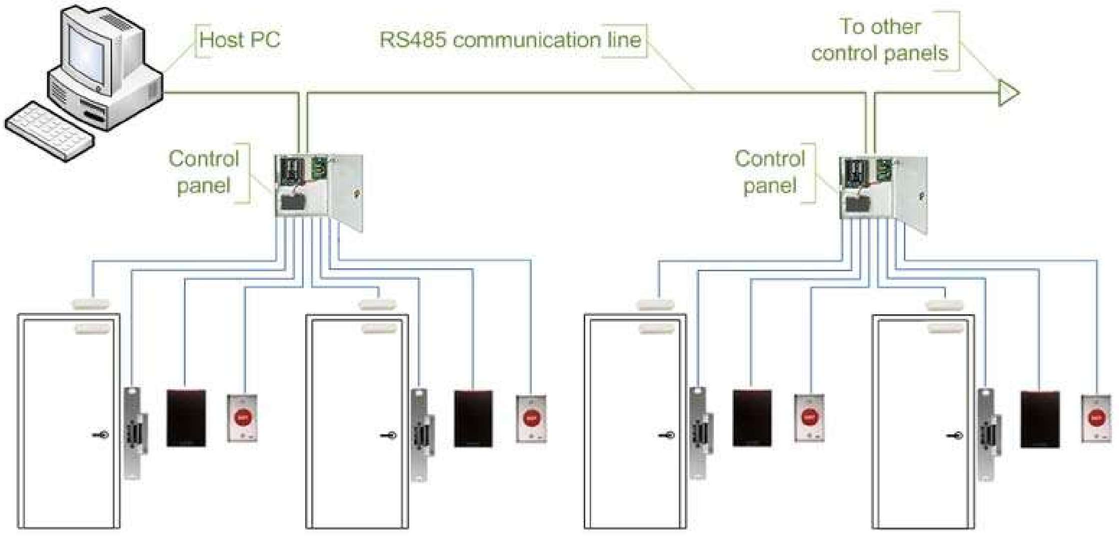
- Basic (non-intelligent) readers: simply read card number or PIN and forward it to a control panel. In case of biometric identification, such readers output ID number of a user. Typically Wiegand protocol is used for transmitting data to the control panel, but other options such as RS-232, RS-485 and Clock/Data are not uncommon. This is the most popular type of access control readers. Examples of such readers are RF Tiny by RFLOGICS, ProxPoint by HID, and P300 by Farpointe Data.

- Semi-intelligent readers: have all inputs and outputs necessary to control door hardware (lock, door contact, exit button), but do not make any access decisions. When a user presents a card or enters PIN, the reader sends information to the main controller and waits for its response. If the connection to the main controller is interrupted, such readers stop working or function in a degraded mode. Usually semi-intelligent readers are connected to a control panel via an RS-485 bus. Examples of such readers are InfoProx Lite IPL200 by CEM Systems and AP-510 by Apollo.
- Intelligent readers: have all inputs and outputs necessary to control door hardware, they also have memory and processing power necessary to make access decisions independently. Same as semi-intelligent readers they are connected to a control panel via an RS-485 bus. The control panel sends configuration updates and retrieves events from the readers. Examples of such readers could be InfoProx IPO200 by CEM Systems and AP-500 by Apollo. There is also a new generation of intelligent readers referred to as "IP readers". Systems with IP readers usually do not have traditional control panels and readers communicate directly to PC that acts as a host. Examples of such readers are PowerNet IP Reader by Isonas Security Systems, ID08 by Solus has the built in webservice to make it user friendly, Edge ER40 reader by HID Global, LogLock and UNiLOCK by ASPiSYS Ltd, and BioEntry Plus reader by Suprema Inc.

Some readers may have additional features such as LCD and function buttons for data collection purposes (i.e. clock-in/clock-out events for attendance reports), camera/speaker/microphone for intercom, and smart card read/write support.

Access control readers may also be classified by the type of identification technology.

Access control system topologies



Access control system using serial controllers

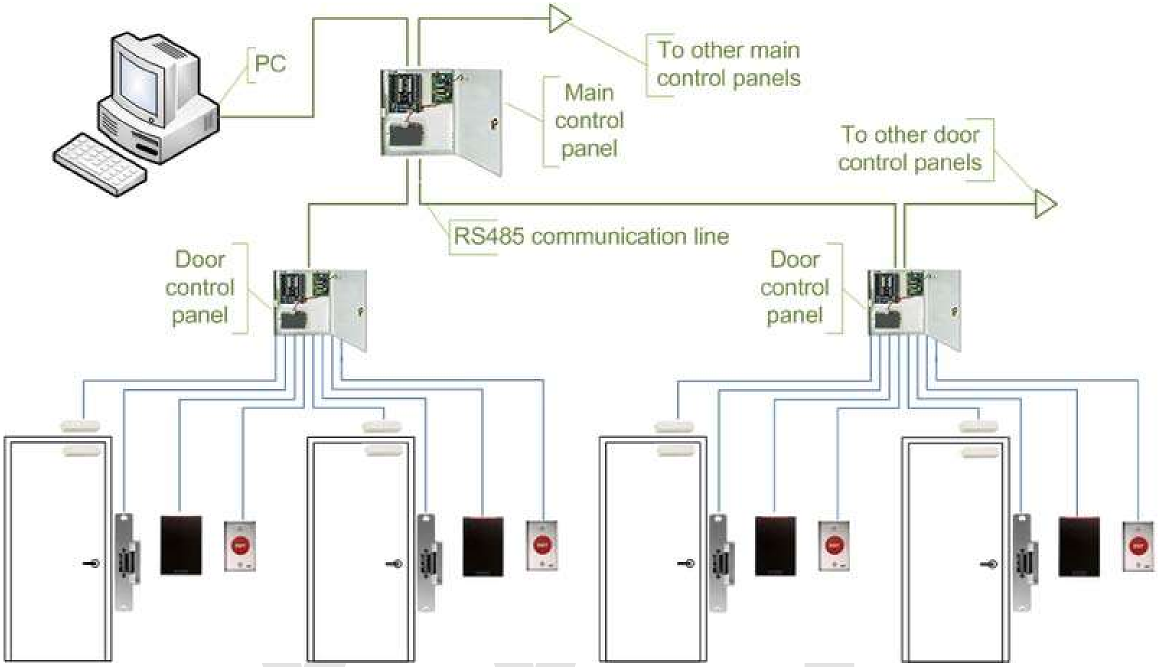
1. Serial controllers. Controllers are connected to a host PC via a serial RS-485 communication line (or via 20mA current loop in some older systems). External RS-232/485 converters or internal RS-485 cards have to be installed as standard PCs do not have RS-485 communication ports. In larger systems multi-port serial IO boards are used, Digi International being one of most popular options. Advantages:

- RS-485 standard allows long cable runs, up to 4000 feet (1200 m)
- Relatively short response time. The maximum number of devices on an RS-485 line is limited to 32, which means that the host can frequently request status updates from each device and display events almost in real time.
- High reliability and security as the communication line is not shared with any other systems.

Disadvantages:

- RS-485 does not allows Star-type wiring unless splitters are used
- RS-485 is not well suited for transferring large amounts of data (i.e. configuration and users). The highest possible throughput is 115.2 kbit/s, but in most system it is downgraded to 56.2 kbit/s or less to increase reliability.
- RS-485 does not allow host PC to communicate with several controllers connected to the same port simultaneously. Therefore in large systems transfers of configuration and users to controllers may take a very long time and interfere with normal operations.
- Controllers cannot initiate communication in case of an alarm. The host PC acts as a master on the RS-485 communication line and controllers have to wait till they are polled.
- Special serial switches are required in order to build a redundant host PC setup.

- Separate RS-485 lines have to be installed instead of using an already existing network infrastructure.
- Cable that meets RS-485 standards is significantly more expensive than the regular Category 5 UTP network cable.
- Operation of the system is highly dependent on the host PC. In case the host PC fails, events from controllers are not retrieved and functions that required interaction between controllers (i.e. anti-passback) stop working.



Access control system using serial main and sub-controllers

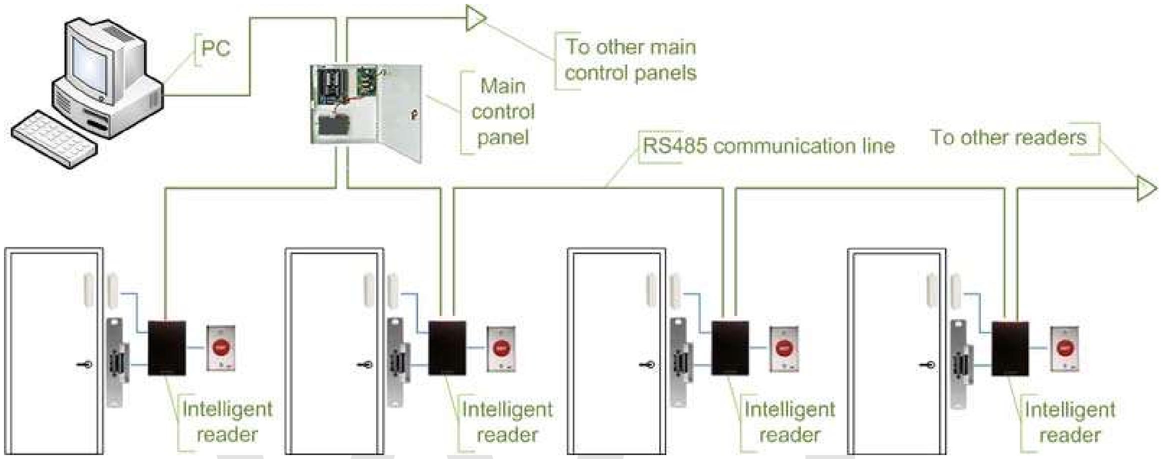
2. Serial main and sub-controllers. All door hardware is connected to sub-controllers (a.k.a. door controllers or door interfaces). Sub-controllers usually do not make access decisions, and forward all requests to the main controllers. Main controllers usually support from 16 to 32 sub-controllers. Advantages:

- Work load on the host PC is significantly reduced, because it only needs to communicate with a few main controllers.
- The overall cost of the system is lower, as sub-controllers are usually simple and inexpensive devices.
- All other advantages listed in the first paragraph apply.

Disadvantages:

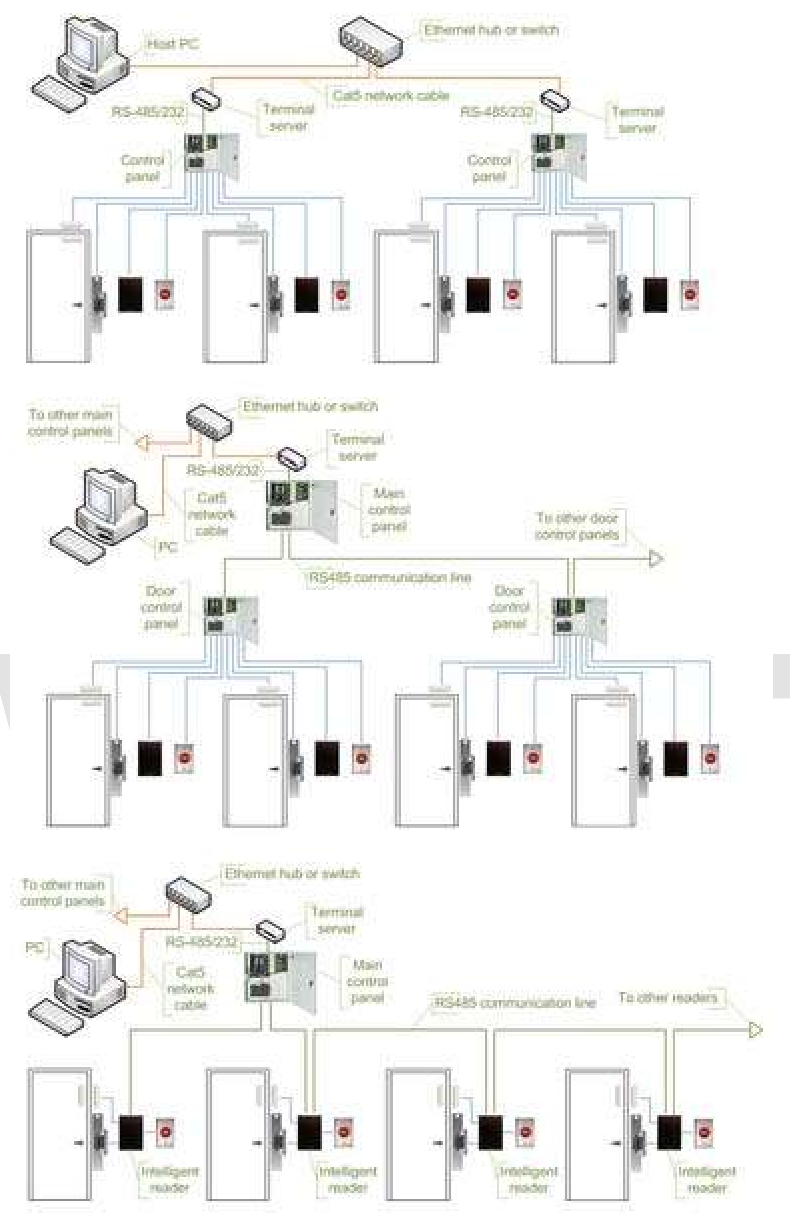
- Operation of the system is highly dependent on main controllers. In case one of the main controllers fails, events from its sub-controllers are not retrieved and functions that require interaction between sub controllers (i.e. anti-passback) stop working.

- Some models of sub-controllers (usually lower cost) have no memory and processing power to make access decisions independently. If the main controller fails, sub-controllers change to degraded mode in which doors are either completely locked or unlocked and no events are recorded. Such sub-controllers should be avoided or used only in areas that do not require high security.
- Main controllers tend to be expensive, therefore such topology is not very well suited for systems with multiple remote locations that have only a few doors.
- All other RS-485-related disadvantages listed in the first paragraph apply.



Access control system using serial main controller and intelligent readers

3. Serial main controllers & intelligent readers. All door hardware is connected directly to intelligent or semi-intelligent readers. Readers usually do not make access decisions, and forward all requests to the main controller. Only if the connection to the main controller is unavailable, the readers use their internal database to make access decisions and record events. Semi-intelligent reader that have no database and cannot function without the main controller should be used only in areas that do not require high security. Main controllers usually support from 16 to 64 readers. All advantages and disadvantages are the same as the ones listed in the second paragraph.



Access control systems using serial controllers and terminal servers

4. Serial controllers with terminal servers. In spite of the rapid development and increasing use of computer networks, access control manufacturers remained conservative and did not rush to introduce network-enabled products. When pressed for solutions with network connectivity, many chose the option requiring less efforts: addition of a terminal server, a device that converts serial data for transmission via LAN or WAN. Terminal servers manufactured by Lantronix and Tibbo Technology are popular in the security industry. Advantages:

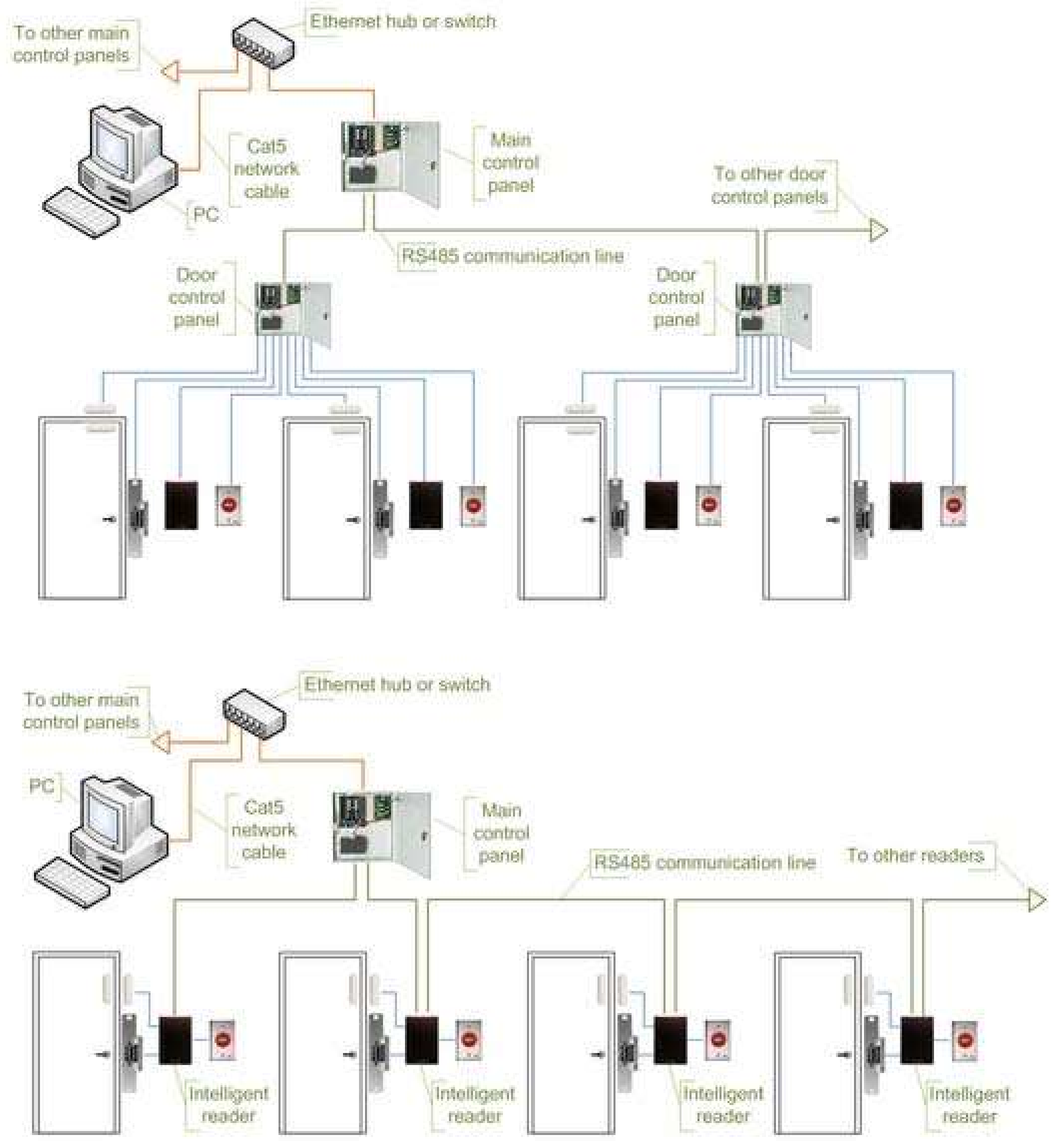
- Allows utilizing existing network infrastructure for connecting separate segments of the system.

- Provides convenient solution in cases when installation of an RS-485 line would be difficult or impossible.

Disadvantages:

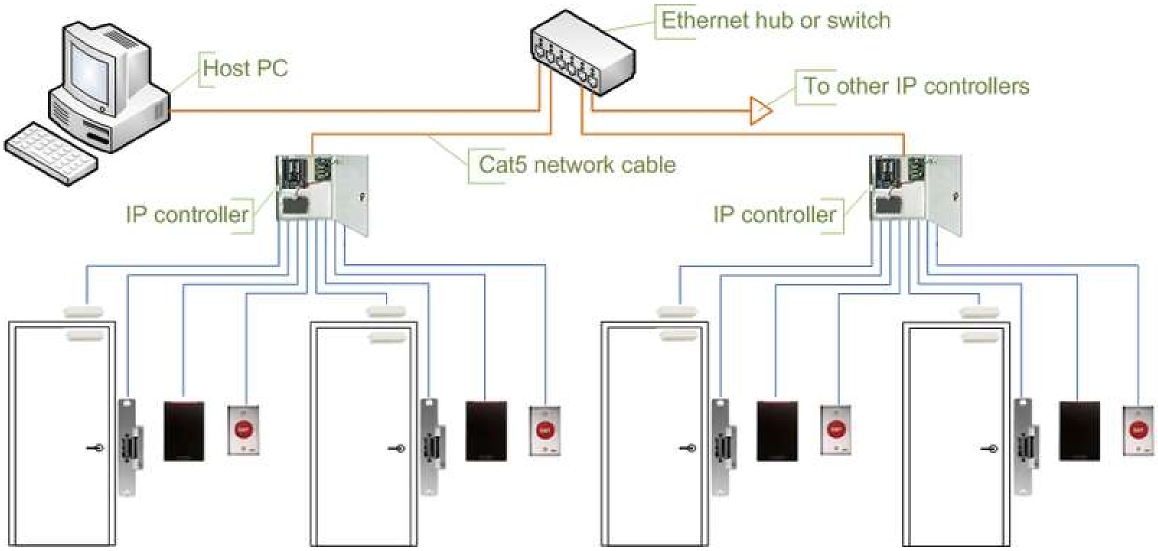
- Increases complexity of the system.
- Creates additional work for installers: usually terminal servers have to be configured independently, not through the interface of the access control software.
- Serial communication link between the controller and the terminal server acts as a bottleneck: even though the data between the host PC and the terminal server travels at the 10/100/1000Mbit/s network speed it then slows down to the serial speed of 112.5 kbit/s or less. There are also additional delays introduced in the process of conversion between serial and network data.

All RS-485-related advantages and disadvantages also apply.



Access control system using network-enabled main controllers

5. Network-enabled main controllers. The topology is nearly the same as described in the second and third paragraphs. The same advantages and disadvantages apply, but the on-board network interface offers a couple valuable improvements. Transmission of configuration and users to the main controllers is faster and may be done in parallel. This makes the system more responsive and does not interrupt normal operations. No special hardware is required in order to achieve redundant host PC setup: in case the primary host PC fails, the secondary host PC may start polling network controllers. The disadvantages introduced by terminal servers (listed in the fourth paragraph) are also eliminated.



Access control system using IP controllers

6. IP controllers. Controllers are connected to a host PC via Ethernet LAN or WAN. Advantages:

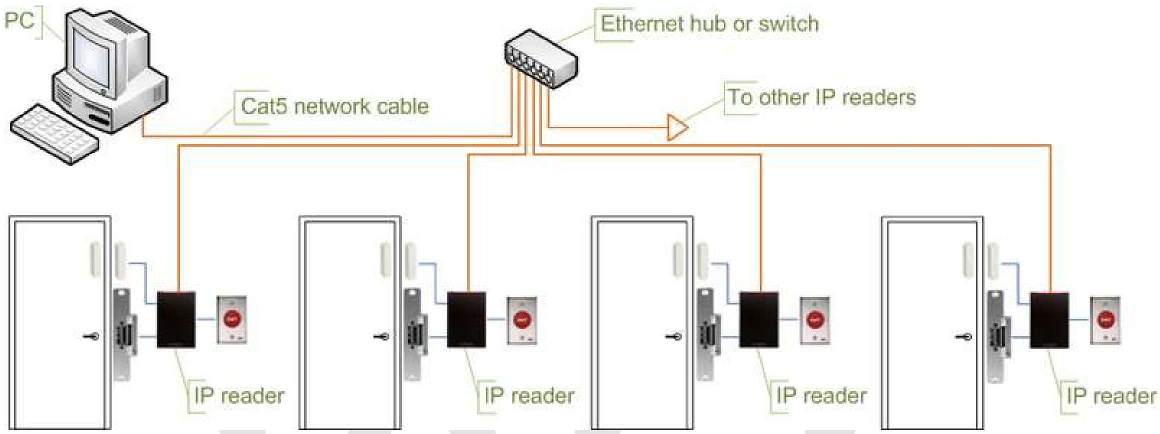
- An existing network infrastructure is fully utilized, there is no need to install new communication lines.
- There are no limitations regarding the number of controllers (32 per line in case of RS-485).
- Special RS-485 installation, termination, grounding and troubleshooting knowledge is not required.
- Communication with controllers may be done at the full network speed, which is important if transferring a lot of data (databases with thousands of users, possibly including biometric records).
- In case of an alarm controllers may initiate connection to the host PC. This ability is important in large systems because it allows to reduce network traffic caused by unnecessary polling.
- Simplifies installation of systems consisting of multiple sites separated by large distances. Basic Internet link is sufficient to establish connections to remote locations.
- Wide selection of standard network equipment is available to provide connectivity in different situations (fiber, wireless, VPN, dual path, PoE)

Disadvantages:

- The system becomes susceptible to network related problems, such as delays in case of heavy traffic and network equipment failures.
- Access controllers and workstations may become accessible to hackers if the network of the organization is not well protected. This threat may be eliminated by physically separating the access control network from the network of the organization. Also it should be noted that most IP controllers utilize either Linux

platform or proprietary operating systems, which makes them more difficult to hack. Industry standard data encryption is also used.

- Maximum distance from a hub or a switch to the controller (if using a copper cable) is 100 meters (330 ft).
- Operation of the system is dependent on the host PC. In case the host PC fails, events from controllers are not retrieved and functions that required interaction between controllers (i.e. anti-passback) stop working. Some controllers, however, have peer-to-peer communication option in order to reduce dependency on the host PC.



Access control system using IP readers

7. IP readers. Readers are connected to a host PC via Ethernet LAN or WAN.
Advantages:

- Most IP readers are PoE capable. This feature makes it very easy to provide battery backed power to the entire system, including the locks and various types of detectors (if used).
- IP readers eliminate the need for controller enclosures.
- There is no wasted capacity when using IP readers (i.e. a 4-door controller would have 25% unused capacity if it was controlling only 3 doors).
- IP reader systems scale easily: there is no need to install new main or sub-controllers.
- Failure of one IP reader does not affect any other readers in the system.

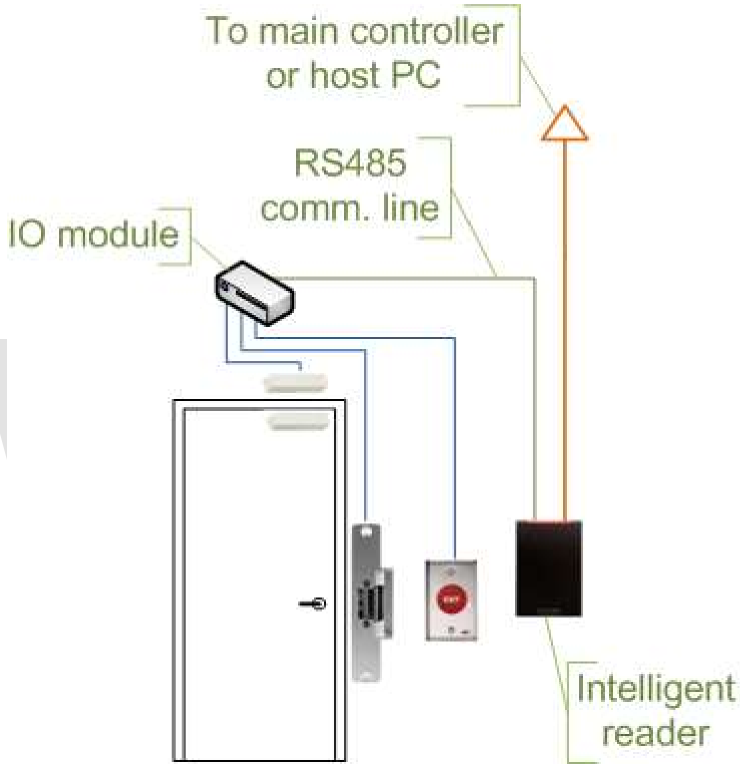
Disadvantages:

- In order to be used in high-security areas IP readers require special input/output modules to eliminate the possibility of intrusion by accessing lock and/or exit button wiring. Not all IP reader manufacturers have such modules available.
- Being more sophisticated than basic readers IP readers are also more expensive and sensitive, therefore they should not be installed outdoors in areas with harsh weather conditions or high possibility of vandalism, unless specifically designed for exterior installation. A few manufacturers make such models.

- In the past, the variety of IP readers in terms of identification technologies and read range was much lower than that of the basic readers. However, with the advent of long range multi-technology readers such as those manufactured by Nedap, Sirit, and a few others, this is no longer so.

The advantages and disadvantages of IP controllers apply to the IP readers as well.

Security risks



Access control door wiring when using intelligent readers and IO module

The most common security risk of intrusion of an access control system is simply following a legitimate user through a door. Often the legitimate user will hold the door for the intruder. This risk can be minimized through security awareness training of the user population or more active means such as turnstiles. In very high security applications this risk is minimized by using a sally port, sometimes called a security vestibule or mantrap where operator intervention is required presumably to assure valid identification.

The second most common risk is from levering the door open. This is surprisingly simple and effective on most doors. The lever could be as small as a screw driver or big as a crow bar. Fully implemented access control systems include forced door monitoring alarms. These vary in effectiveness usually failing from high false positive alarms, poor database configuration, or lack of active intrusion monitoring.

Similar to levering is crashing through cheap partition walls. In shared tenant spaces the demisal wall is a vulnerability. Along the same lines is breaking sidelights.

Spoofing locking hardware is fairly simple and more elegant than levering. A strong magnet can operate the solenoid controlling bolts in electric locking hardware. Motor locks, more prevalent in Europe than in the US, are also susceptible to this attack using a donut shaped magnet. It is also possible to manipulate the power to the lock either by removing or adding current.

Access cards themselves have proven vulnerable to sophisticated attacks. Enterprising hackers have built portable readers that capture the card number from a user's proximity card. The hacker simply walks by the user, reads the card, and then presents the number to a reader securing the door. This is possible because card numbers are sent in the clear, no encryption being used.

Finally, most electric locking hardware still have mechanical keys as a failover. Mechanical key locks are vulnerable to bumping.

The need-to-know principle

The need to know principle can be enforced with user access controls and authorization procedures and its objective is to ensure that only authorized individuals gain access to information or systems necessary to undertake their duties.

Computer security

In computer security, access control includes authentication, authorization and audit. It also includes measures such as physical devices, including biometric scans and metal locks, hidden paths, digital signatures, encryption, social barriers, and monitoring by humans and automated systems.

In any access control model, the entities that can perform actions in the system are called *subjects*, and the entities representing resources to which access may need to be controlled are called *objects*. Subjects and objects should both be considered as software entities and as human users. Although some systems equate subjects with *user IDs*, so that all processes started by a user by default have the same authority, this level of control is not fine-grained enough to satisfy the Principle of least privilege, and arguably is responsible for the prevalence of malware in such systems.

In some models, for example the object-capability model, any software entity can potentially act as both a subject and object.

Access control models used by current systems tend to fall into one of two classes: those based on capabilities and those based on access control lists (ACLs). In a capability-based model, holding an unforgeable reference or *capability* to an object provides access to the object (roughly analogous to how possession of your house key grants you access to your house); access is conveyed to another party by transmitting such a capability over a secure channel. In an ACL-based model, a subject's access to an object depends on whether its identity is on a list associated with the object (roughly analogous to how a bouncer at a private party would check your ID to see if your name is on the guest list); access is conveyed by editing the list. (Different ACL systems have a variety of different conventions regarding who or what is responsible for editing the list and how it is edited.)

Both capability-based and ACL-based models have mechanisms to allow access rights to be granted to all members of a *group* of subjects (often the group is itself modeled as a subject).

Access control systems provide the essential services of *identification and authentication* (*I&A*), *authorization*, and *accountability* where:

- identification and authentication determine who can log on to a system, and the association of users with the software subjects that they are able to control as a result of logging in;
- authorization determines what a subject can do;
- accountability identifies what a subject (or all subjects associated with a user) did.

Identification and authentication (I&A)

Identification and authentication (I&A) is the process of verifying that an identity is bound to the entity that makes an assertion or claim of identity. The I&A process assumes that there was an initial validation of the identity, commonly called identity proofing. Various methods of identity proofing are available ranging from in person validation using government issued identification to anonymous methods that allow the claimant to remain anonymous, but known to the system if they return. The method used for identity proofing and validation should provide an assurance level commensurate with the intended use of the identity within the system. Subsequently, the entity asserts an identity together with an authenticator as a means for validation. The only requirements for the identifier is that it must be unique within its security domain.

Authenticators are commonly based on at least one of the following four factors:

- *Something you know*, such as a password or a personal identification number (PIN). This assumes that only the owner of the account knows the password or PIN needed to access the account.
- *Something you have*, such as a smart card or security token. This assumes that only the owner of the account has the necessary smart card or token needed to unlock the account.
- *Something you are*, such as fingerprint, voice, retina, or iris characteristics.
- *Where you are*, for example inside or outside a company firewall, or proximity of login location to a personal GPS device.

Authorization

Authorization applies to subjects. Authorization determines what a subject can do on the system.

Most modern operating systems define sets of permissions that are variations or extensions of three basic types of access:

- Read (R): The subject can
  - Read file contents
  - List directory contents
- Write (W): The subject can change the contents of a file or directory with the following tasks:
  - Add
  - Create
  - Delete
  - Rename
- Execute (X): If the file is a program, the subject can cause the program to be run. (In Unix systems, the 'execute' permission doubles as a 'traverse directory' permission when granted for a directory.)

These rights and permissions are implemented differently in systems based on *discretionary access control* (*DAC*) and *mandatory access control* (*MAC*).

Accountability

Accountability uses such system components as *audit trails* (records) and *logs* to associate a subject with its actions. The information recorded should be sufficient to map the subject to a controlling user. Audit trails and logs are important for

- Detecting security violations
- Re-creating security incidents

If no one is regularly reviewing your logs and they are not maintained in a secure and consistent manner, they may not be admissible as evidence.

Many systems can generate automated reports based on certain predefined criteria or thresholds, known as *clipping levels*. For example, a clipping level may be set to generate a report for the following:

- More than three failed logon attempts in a given period
- Any attempt to use a disabled user account

These reports help a system administrator or security administrator to more easily identify possible break-in attempts.

Access control techniques

Access control techniques are sometimes categorized as either discretionary or non-discretionary. The three most widely recognized models are Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role Based Access Control (RBAC). MAC and RBAC are both non-discretionary.

Attribute-based access control

In attribute-based access control (ABAC), access is granted not based on the rights of the subject associated with a user after authentication, but based on attributes of the user. The user has to prove so called claims about his attributes to the access control engine. An attribute-based access control policy specifies which claims need to satisfied in order to grant access to an object. For instance the claim could be "older than 18" . Any user that can prove this claim is granted access. Users can be anonymous as authentication and identification are not strictly required. One does however require means for proving claims anonymously. This can for instance be achieved using anonymous credentials or XACML (extensible access control markup language).

Discretionary access control

Discretionary access control (DAC) is an access policy determined by the owner of an object. The owner decides who is allowed to access the object and what privileges they have.

Two important concepts in DAC are

- File and data ownership: Every object in the system has an *owner*. In most DAC systems, each object's initial owner is the subject that caused it to be created. The access policy for an object is determined by its owner.
- Access rights and permissions: These are the controls that an owner can assign to other subjects for specific resources.

Access controls may be discretionary in ACL-based or capability-based access control systems. (In capability-based systems, there is usually no explicit concept of 'owner', but the creator of an object has a similar degree of control over its access policy.)

Mandatory access control

Mandatory access control (MAC) is an access policy determined by the system, not the owner. MAC is used in multilevel systems that process highly sensitive data, such as classified government and military information. A multilevel system is a single computer system that handles multiple classification levels between subjects and objects.

- Sensitivity labels: In a MAC-based system, all subjects and objects must have labels assigned to them. A subject's sensitivity label specifies its level of trust. An object's sensitivity label specifies the level of trust required for access. In order to access a given object, the subject must have a sensitivity level equal to or higher than the requested object.

- Data import and export: Controlling the import of information from other systems and export to other systems (including printers) is a critical function of MAC-based systems, which must ensure that sensitivity labels are properly maintained and implemented so that sensitive information is appropriately protected at all times.

Two methods are commonly used for applying mandatory access control:

- Rule-based (or label-based) access control: This type of control further defines specific conditions for access to a requested object. All MAC-based systems implement a simple form of rule-based access control to determine whether access should be granted or denied by matching:
  - An object's sensitivity label
  - A subject's sensitivity label
- Lattice-based access control: These can be used for complex access control decisions involving multiple objects and/or subjects. A lattice model is a mathematical structure that defines greatest lower-bound and least upper-bound values for a pair of elements, such as a subject and an object.

Few systems implement MAC; XTS-400 is an example of one that does. The computer system at the company in the movie *Tron* is an example of MAC in popular culture.

Role-based access control

Role-based access control (RBAC) is an access policy determined by the system, not the owner. RBAC is used in commercial applications and also in military systems, where multi-level security requirements may also exist. RBAC differs from DAC in that DAC allows users to control access to their resources, while in RBAC, access is controlled at the system level, outside of the user's control. Although RBAC is non-discretionary, it can be distinguished from MAC primarily in the way permissions are handled. MAC controls read and write permissions based on a user's clearance level and additional labels. RBAC controls collections of permissions that may include complex operations such as an e-commerce transaction, or may be as simple as read or write. A role in RBAC can be viewed as a set of permissions.

Three primary rules are defined for RBAC:

1. Role assignment: A subject can execute a transaction only if the subject has selected or been assigned a role.
2. Role authorization: A subject's active role must be authorized for the subject. With rule 1 above, this rule ensures that users can take on only roles for which they are authorized.

3. Transaction authorization: A subject can execute a transaction only if the transaction is authorized for the subject's active role. With rules 1 and 2, this rule ensures that users can execute only transactions for which they are authorized.

Additional constraints may be applied as well, and roles can be combined in a hierarchy where higher-level roles subsume permissions owned by sub-roles.

Most IT vendors offer RBAC in one or more products.

Telecommunication

In telecommunication, the term *access control* is defined in U.S. Federal Standard 1037C with the following meanings:

1. A service feature or technique used to permit or deny use of the components of a communication system.
2. A technique used to define or restrict the rights of individuals or application programs to obtain data from, or place data onto, a storage device.
3. The definition or restriction of the rights of individuals or application programs to obtain data from, or place data into, a storage device.
4. The process of limiting access to the resources of an AIS to authorized users, programs, processes, or other systems.
5. That function performed by the resource controller that allocates system resources to satisfy user requests.

Notice that this definition depends on several other technical terms from Federal Standard 1037C.

Access control to each element in the product definition data base can be specified. Read only access can be given to personnel not directly involved with the design, development and planning process. Creation and maintenance access can be given to the individuals responsible for product and process design. As Product Data Management systems evolve towards Collaborative Product Commerce (CPC) systems which are used across multiple enterprises in a supply chain, access control becomes more critical and requires control to limit access to specific projects, products or parts for a specific supplier or customer

- Component / Material Classification

Components and materials can be classified and organized and attributes assigned. This supports standardization by identifying similar components/materials, eliminating redundancy, and establishing a preferred parts list. Establishing classes and subclasses with attributes allows a designer to search and select a needed material, component or assembly with minimal effort thereby avoiding having to re-specifying an existing or similar component or material

- Product Structure

Since the relationship of a product's parts is a logical one maintained by the information system rather than a fixed physical relationship as represented on a drawing, it is possible to readily maintain more than one relationship. This will allow different views of part relationships in assemblies to correspond to the various departmental needs (e.g., engineering and manufacturing product structures), while maintaining rigor and consistency of the product's definition through this single data base. Thus, this one logical data base can support product and process design requirements as well as maintain part relationships to serve as a manufacturing bill of materials for MRP II/ERP. In other words, PDM provides the ability to hold not just the physical relationships between parts in an assembly but also other kinds of structures; for instance, manufacturing, financial, maintenance or document relationships. So, it is possible for specialist team members to see the product structured from their point of view. Product data can be accessed via this complete Bill of Materials. This access includes assemblies, parts and related documents.An integrated approach to developing, organizing and maintaining part and product definition data facilitates the design process, makes design data more readily usable and enhances integration with process requirements

- Engineering Changes

Engineering changes can be facilitated with this configuration management and administrative control embedded within the system. CAE/CAD tools will enable engineering changes to be more thoroughly developed and analyzed to better define change impact. Once a design has been created, it can be checked-out electronically to a workstation for engineering changes. When the changes have been made, it can be returned to the central database and placed in a queue or an email notification sent for approval by designated parties. In this manner, a Change Control Board (CCB) can even "convene" and provide individual member's input electronically. In addition to supporting engineering analysis, information related to procurement, inventory, manufacturing and cost is available for members of the CCB to evaluate, designate the effectivity of the change and determine the disposition of existing items.

- Process Management and Workflow

PDM systems support process management by defining process steps related to the development, distribution and use of product data. The process is defined in the form of specified process steps and release or promotion levels that the data must achieve. The manner in which the process is defined varies with every PDM system. Within a project, responsibilities are defined for the process steps - who needs to approve the data or work on the data before it moves to the next release or promotion level. While, the current process is defined in a company's configuration management or engineering change procedures and in its new product development process, often changes have to be made to take advantage of the communication and coordination capabilities of the PDM system. This new data is moved to the next person's "in basket" within PDM or an email notification is sent.

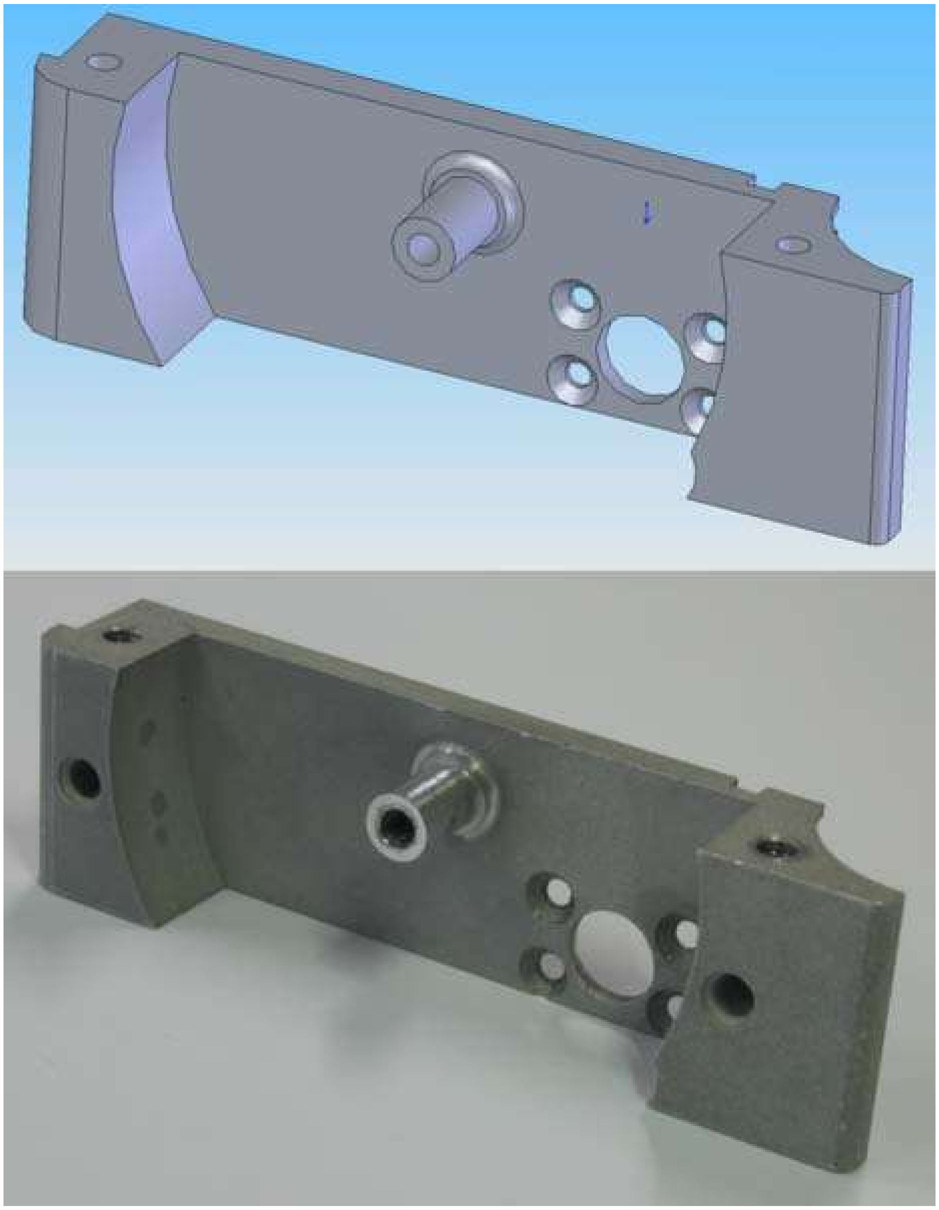
- Collaboration

Collaboration can be supported in several ways. First, a PDM system may be the gateway that a team uses to access the information under discussion avoiding the need to copy and distribute a series of paper documents. Second, the PDM system may provide a synchronous or asynchronous collaboration environment for team members to access, present, review and produce feedback on product and process information. Further, this collaboration tool may incorporate viewing and mark-up capability, as well as provide the ability to store marked-up files or documents submitted by collaborators. Third, what are now described as collaborative product commerce systems (CPC), provide extended PDM functionality and access control outside the enterprise for customers, suppliers and interested third parties (e.g., regulatory agencies). This speeds the distribution of information, enhances coordination, and speeds the capture of feedback.

- Storage and retrieval of product information
- Product structure modeling and management
    - Bill of materials
    - Product configurations
    - Product variations
    - Product versions
- Project tracking
- Resource planning

Chapter- 7

Computer-Aided Manufacturing



CAD model and CNC machined part

Computer-aided manufacturing (**CAM**) is the use of computer software to control machine tools and related machinery in the manufacturing of workpieces. This is not the only definition for CAM, but it is the most common; CAM may also refer to the use of a computer to assist in all operations of a manufacturing plant, including planning, management, transportation and storage. Its primary purpose is to create a faster production process and components and tooling with more precise dimensions and material consistency, which in some cases, uses only the required amount of raw material (thus minimizing waste), while simultaneously reducing energy consumption.

CAM is a subsequent computer-aided process after computer-aided design (CAD) and sometimes computer-aided engineering (CAE), as the model generated in CAD and verified in CAE can be input into CAM software, which then controls the machine tool.

Overview

Chrome-cobalt disc with dental implants manufactured using CAM

Traditionally, CAM has been considered as a numerical control (NC) programming tool, wherein two-dimensional (2-D) or three-dimensional (3-D) models of components generated in CAD software are used to generate G-code to drive computer numerically controlled (CNC) machine tools. Simple designs such as bolt circles or basic contours do not necessitate importing a CAD file.

As with other “Computer-Aided” technologies, CAM does not eliminate the need for skilled professionals such as manufacturing engineers, NC programmers, or machinists. CAM, in fact, leverages both the value of the most skilled manufacturing professionals through advanced productivity tools, while building the skills of new professionals through visualization, simulation and optimization tools.

History

The first commercial applications of CAM were in large companies in the automotive and aerospace industries for example UNISURF in 1971 at Renault for car body design and tooling.

Historically, CAM software was seen to have several shortcomings that necessitated an overly high level of involvement by skilled CNC machinists. Fallows created the first CAM software but this had severe shortcomings and was promptly taken back into the developing stage. CAM software would output code for the least capable machine, as each machine tool control added on to the standard G-code set for increased flexibility. In some cases, such as improperly set up CAM software or specific tools, the CNC machine required manual editing before the program will run properly. None of these issues were so insurmountable that a thoughtful engineer or skilled machine operator could not overcome for prototyping or small production runs; G-Code is a simple language. In high production or high precision shops, a different set of problems were encountered where an experienced CNC machinist must both hand-code programs and run CAM software.

Integration of CAD with other components of CAD/CAM/CAE Product lifecycle management (PLM) environment requires an effective CAD data exchange. Usually it had been necessary to force the CAD operator to export the data in one of the common data formats, such as IGES or STL, that are supported by a wide variety of software. The output from the CAM software is usually a simple text file of G-code, sometimes many thousands of commands long, that is then transferred to a machine tool using a direct numerical control (DNC) program.

CAM packages could not, and still cannot, reason as a machinist can. They could not optimize toolpaths to the extent required of mass production. Users would select the type of tool, machining process and paths to be used. While an engineer may have a working knowledge of g-code programming, small optimization and wear issues compound over time. Mass-produced items that require machining are often initially created through casting or some other non-machine method. This enables hand-written, short, and highly optimized g-code that could not be produced in a CAM package.

At least in the United States, there is a shortage of young, skilled machinists entering the workforce able to perform at the extremes of manufacturing; high precision and mass production. As CAM software and machines become more complicated, the skills required of a machinist or machine operator advance to approach that of a computer programmer and engineer rather than eliminating the CNC machinist from the workforce.

Typical areas of concern:

- High Speed Machining, including streamlining of tool paths
- Multi-function Machining
- 5 Axis Machining
- Feature recognition and machining
- Automation of Machining processes
- Ease of Use

Overcoming historical shortcomings

Over time, the historical shortcomings of CAM are being attenuated, both by providers of niche solutions and by providers of high-end solutions. This is occurring primarily in three arenas:

1. Ease of use
2. Manufacturing complexity
3. Integration with PLM and the extended enterprise

Ease in use

For the user who is just getting started as a CAM user, out-of-the-box capabilities providing Process Wizards, templates, libraries, machine tool kits, automated feature based machining and job function specific tailorable user interfaces build user confidence and speed the learning curve.

User confidence is further built on 3D visualization through a closer integration with the 3D CAD environment, including error-avoiding simulations and optimizations.

Manufacturing complexity

The manufacturing environment is increasingly complex. The need for CAM and PLM tools by the manufacturing engineer, NC programmer or machinist is similar to the need for computer assistance by the pilot of modern aircraft systems. The modern machinery cannot be properly used without this assistance.

Today's CAM systems support the full range of machine tools including: turning, 5 axis machining and wire EDM. Today’s CAM user can easily generate streamlined tool paths, optimized tool axis tilt for higher feed rates and optimized Z axis depth cuts as well as driving non-cutting operations such as the specification of probing motions.

Integration with PLM and the extended enterprise

Today’s competitive and successful companies have used PLM to integrate manufacturing with enterprise operations from concept through field support of the finished product.

To ensure ease of use appropriate to user objectives, modern CAM solutions are scalable from a stand-alone CAM system to a fully integrated multi-CAD 3D solution-set. These solutions are created to meet the full needs of manufacturing personnel including part planning, shop documentation, resource management and data management and exchange.

Machining process

Most machining progresses through four stages, each of which is implemented by a variety of basic and sophisticated strategies, depending on the material and the software available. The stages are:

Roughing

This process begins with raw stock, known as billet, and cuts it very roughly to shape of the final model. In milling, the result often gives the appearance of terraces, because the strategy has taken advantage of the ability to cut the model horizontally. Common strategies are zig-zag clearing, offset clearing, plunge roughing, rest-roughing.

Semi-finishing

This process begins with a roughed part that unevenly approximates the model and cuts to within a fixed offset distance from the model. The semi-finishing pass must leave a small amount of material so the tool can cut accurately while finishing, but not so little that the tool and material deflect instead of shearing. Common strategies are raster passes, waterline passes, constant step-over passes, pencil milling.

Finishing

Finishing involves a slow pass across the material in very fine steps to produce the finished part. In finishing, the step between one pass and another is minimal. Feed rates are low and spindle speeds are raised to produce an accurate surface.

Contour milling

In milling applications on hardware with five or more axes, a separate finishing process called contouring can be performed. Instead of stepping down in fine-grained increments to approximate a surface, the workpiece is rotated to make the cutting surfaces of the tool tangent to the ideal part features. This produces an excellent surface finish with high dimensional accuracy.

Software

The 10 largest CAM software products and companies, by end-user payments in year 2008 are, sorted alphabetically:

- CATIA from Dassault Systèmes
- Cimatron from Cimatron group
- Edgecam from Planit, formerly Pathtrace
- Mastercam from CNC Software
- NX from Siemens PLM Software
- Powermill from Delcam
- Pro/E from PTC
- Space-E/CAM from NDES
- Tebis from Tebis AG
- WorkNC from Sescoi

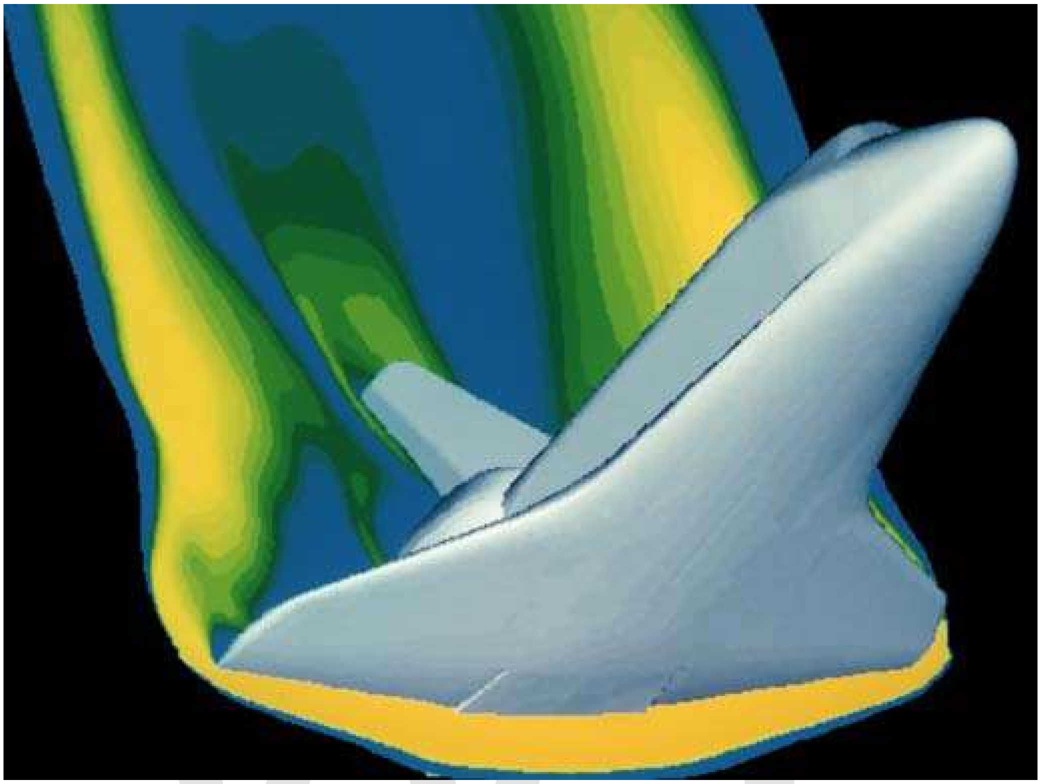
Other CAM products and companies are AlibreCAM, Alphacam, BobCAD, CAMWorks, Dolphin, ESPRIT, GCAM, GIBcam, GibbsCAM, GoElan, MazaCAM, MetaCAM, OneCNC, RhinoCAM, SolidCAM, SprutCAM, SUM3D, SurfCAM, T-FLEX, TopSolid, VisualMILL and VoluMill.

Chapter- 8

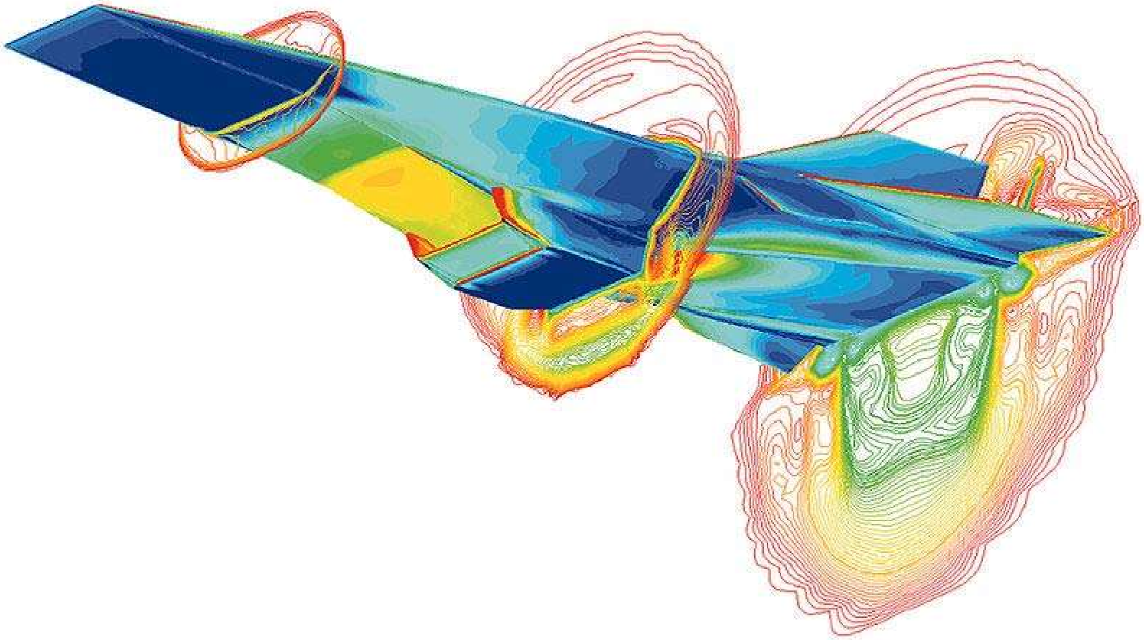
Computational Fluid Dynamics

Computational fluid dynamics (CFD) is a branch of fluid mechanics that uses numerical methods and algorithms to solve and analyze problems that involve fluid flows. Computers are used to perform the calculations required to simulate the interaction of liquids and gases with surfaces defined by boundary conditions. With high-speed supercomputers, better solutions can be achieved. Ongoing research, however, yield software that improves the accuracy and speed of complex simulation scenarios such as transonic or turbulent flows. Initial validation of such software is performed using a wind tunnel with the final validation coming in flight tests.

Background and history



A computer simulation of high velocity air flow around the Space Shuttle during re-entry.



A simulation of the Hyper-X scramjet vehicle in operation at Mach-7

The fundamental basis of almost all CFD problems are the Navier–Stokes equations, which define any single-phase fluid flow. These equations can be simplified by removing terms describing viscosity to yield the Euler equations. Further simplification, by removing terms describing vorticity yields the full potential equations. Finally, these equations can be linearized to yield the linearized potential equations.

Historically, methods were first developed to solve the Linearized Potential equations. Two-dimensional methods, using conformal transformations of the flow about a cylinder to the flow about an airfoil were developed in the 1930s. The computer power available paced development of three-dimensional methods. The first paper on a practical three-dimensional method to solve the linearized potential equations was published by John Hess and A.M.O. Smith of Douglas Aircraft in 1967. This method discretized the surface of the geometry with panels, giving rise to this class of programs being called Panel Methods. Their method itself was simplified, in that it did not include lifting flows and hence was mainly applied to ship hulls and aircraft fuselages. The first lifting Panel Code (A230) was described in a paper written by Paul Rubbert and Gary Saaris of Boeing Aircraft in 1968. In time, more advanced three-dimensional Panel Codes were developed at Boeing (PANAIR, A502), Lockheed (Quadpan), Douglas (HESS), McDonnell Aircraft (MACAERO), NASA (PMARC) and Analytical Methods (WBAERO, USAERO and VSAERO). Some (PANAIR, HESS and MACAERO) were higher order codes, using higher order distributions of surface singularities, while others (Quadpan, PMARC, USAERO and VSAERO) used single singularities on each surface panel. The advantage of the lower order codes was that they ran much faster on the computers of the time. Today, VSAERO has grown to be a multi-order code and is the most widely used program of this class. It has been used in the development of many submarines, surface ships, automobiles, helicopters , aircraft, and more recently wind turbines. Its sister code, USAERO is an unsteady panel method that has also been used for modeling such things as high speed trains and racing yachts. The NASA PMARC code from an early version of VSAERO and a derivative of PMARC, named CMARC, is also commercially available.

In the two-dimensional realm, a number of Panel Codes have been developed for airfoil analysis and design. The codes typically have a boundary layer analysis included, so that viscous effects can be modeled. Professor Richard Eppler of the University of Stuttgart developed the PROFIL code, partly with NASA funding, which became available in the early 1980s. This was soon followed by MIT Professor Mark Drela's XFOIL code. Both PROFIL and XFOIL incorporate two-dimensional panel codes, with coupled boundary layer codes for airfoil analysis work. PROFIL uses a conformal transformation method for inverse airfoil design, while XFOIL has both a conformal transformation and an inverse panel method for airfoil design. Both codes are used.

An intermediate step between Panel Codes and Full Potential codes were codes that used the Transonic Small Disturbance equations. In particular, the three-dimensional WIBCO code, developed by Charlie Boppe of Grumman Aircraft in the early 1980s has seen heavy use.

Developers turned to Full Potential codes, as panel methods could not calculate the non-linear flow present at transonic speeds. The first description of a means of using the Full Potential equations was published by Earll Murman and Julian Cole of Boeing in 1970. Frances Bauer, Paul Garabedian and David Korn of the Courant Institute at New York University (NYU) wrote a series of two-dimensional Full Potential airfoil codes that were widely used, the most important being named Program H. A further growth of Program H was developed by Bob Melnik and his group at Grumman Aerospace as Grumfoil. Antony Jameson, originally at Grumman Aircraft and the Courant Institute of NYU, worked with David Caughey to develop the important three-dimensional Full Potential code FLO22 in 1975. Many Full Potential codes emerged after this, culminating in Boeing's Tranair (A633) code, which still sees heavy use.

The next step was the Euler equations, which promised to provide more accurate solutions of transonic flows. The methodology used by Jameson in his three-dimensional FLO57 code (1981) was used by others to produce such programs as Lockheed's TEAM program and IAI/Analytical Methods' MGAERO program. MGAERO is unique in being a structured cartesian mesh code, while most other such codes use structured body-fitted grids (with the exception of NASA's highly successful CART3D code, Lockheed's SPLITFLOW code and Georgia Tech's NASCART-GT). Antony Jameson also developed the three-dimensional AIRPLANE code (1985) which made use of unstructured tetrahedral grids.

In the two-dimensional realm, Mark Drela and Michael Giles, then graduate students at MIT, developed the ISES Euler program (actually a suite of programs) for airfoil design and analysis. This code first became available in 1986 and has been further developed to design, analyze and optimize single or multi-element airfoils, as the MSES program. MSES sees wide use throughout the world. A derivative of MSES, for the design and analysis of airfoils in a cascade, is MISES, developed by Harold "Guppy" Youngren while he was a graduate student at MIT.

The Navier–Stokes equations were the ultimate target of developers. Two-dimensional codes, such as NASA Ames' ARC2D code first emerged. A number of three-dimensional codes were developed (OVERFLOW, CFL3D are two successful NASA contributions), leading to numerous commercial packages.

Methodology

In all of these approaches the same basic procedure is followed.

- During preprocessing
    - The geometry (physical bounds) of the problem is defined.
    - The volume occupied by the fluid is divided into discrete cells (the mesh). The mesh may be uniform or non uniform.
    - The physical modeling is defined – for example, the equations of motions + enthalpy + radiation + species conservation

    - Boundary conditions are defined. This involves specifying the fluid behaviour and properties at the boundaries of the problem. For transient problems, the initial conditions are also defined.
- The simulation is started and the equations are solved iteratively as a steady-state or transient.
- Finally a postprocessor is used for the analysis and visualization of the resulting solution.

Discretization methods

The stability of the chosen discretization is generally established numerically rather than analytically as with simple linear problems. Special care must also be taken to ensure that the discretization handles discontinuous solutions gracefully. The Euler equations and Navier–Stokes equations both admit shocks, and contact surfaces.

Some of the discretization methods being used are:

Finite Volume Method

The finite volume method (FVM) is a common approach used in CFD codes. The governing equations are solved over discrete control volumes. Finite volume methods recast the governing partial differential equations (typically the Navier-Stokes equations) in a conservative form, and then discretize the new equation. This guarantees the conservation of fluxes through a particular control volume. Though the overall solution will be conservative in nature, there is no guarantee that it is the actual solution. The finite volume equation yields governing equations in the form,

$$\frac{\partial}{\partial t}\iiint Q\,dV + \iint F\,d\mathbf{A} = 0,$$

where Q is the vector of conserved variables, F is the vector of fluxes, V is the volume of the control volume element, and $\mathbf{A}$is the surface area of the control volume element.

Finite Element Method

The finite element method (FEM) is used in structural analysis of solids, but is also applicable to fluids. However, the FEM formulation requires special care to ensure a conservative solution. The FEM formulation has been adapted for use with fluid dynamics governing equations. Although FEM must be carefully formulated to be conservative, it is much more stable than the finite volume approach However, FEM can require more memory than FVM.

In this method, a weighted residual equation is formed:

$$R_i = \iiint W_i Q \, dV^e$$

where R_i is the equation residual at an element vertex i, Q is the conservation equation expressed on an element basis, W_i is the weight factor, and V^e is the volume of the element.

Finite Difference Method

The finite difference method (FDM) has historical importance and is simple to program. It is currently only used in few specialized codes. Modern finite difference codes make use of an embedded boundary for handling complex geometries, making these codes highly efficient and accurate. Other ways to handle geometries include use of overlapping grids, where the solution is interpolated across each grid.

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} + \frac{\partial H}{\partial z} = 0$$

where Q is the vector of conserved variables, and F, G, and H are the fluxes in the x, y, and z directions respectively.

Boundary Element Method

In the boundary element method, the boundary occupied by the fluid is divided into a surface mesh.

High Resolution Discretization Schemes

High-resolution schemes are used where shocks or discontinuities are present. Capturing sharp changes in the solution requires the use of second or higher-order numerical schemes that do not introduce spurious oscillations. This usually necessitates the application of flux limiters to ensure that the solution is total variation diminishing.

Turbulence models

In studying turbulent flows, the objective is to obtain a theory or a model that can yield quantities of interest, such as velocities. For turbulent flow, the range of length scales and complexity of phenomena make most approaches impossible. The primary approach in this case is to create numerical models to calculate the properties of interest. A selection of some commonly-used computational models for turbulent flows are presented in this section.

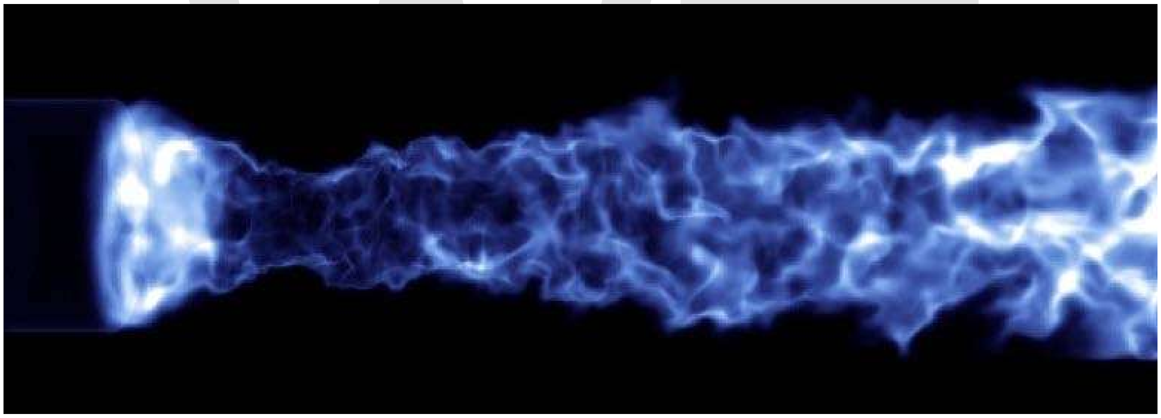
The chief difficulty in modeling turbulent flows comes from the wide range of length and time scales associated with turbulent flow. As a result, turbulence models can be classified based on the range of these length and time scales that are modeled and the range of length and time scales that are resolved. The more turbulent scales that are resolved, the finer the resolution of the simulation, and therefore the higher the computational cost. If a majority or all of the turbulent scales are modeled, the computational cost is very low, but the tradeoff comes in the form of decreased accuracy.

In addition to the wide range of length and time scales and the associated computational cost, the governing equations of fluid dynamics contain a non-linear convection term and a non-linear and non-local pressure gradient term. These nonlinear equations must be solved numerically with the appropriate boundary and initial conditions.

Direct numerical simulation

Direct numerical simulation (DNS) resolves the entire range of turbulent length scales. This marginalizes the effect of models, but is extremely expensive. The computational cost is proportional to Re^3. DNS is intractable for flows with complex geometries or flow configurations.

Large eddy simulation



Volume rendering of a non-premixed swirl flame as simulated by LES.

Large eddy simulation (LES) is a technique in which the smallest scales of the flow are removed through a filtering operation, and their effect modeled using subgrid scale models. This allows the largest and most important scales of the turbulence to be resolved, while greatly reducing the computational cost incurred by the smallest scales. This method requires greater computational resources than RANS methods, but is far cheaper than DNS.

Detached eddy simulation

Detached eddy simulations (DES) is a modification of a RANS model in which the model switches to a subgrid scale formulation in regions fine enough for LES calculations. Regions near solid boundaries and where the turbulent length scale is less than the maximum grid dimension are assigned the RANS mode of solution. As the turbulent length scale exceeds the grid dimension, the regions are solved using the LES mode.

Therefore the grid resolution for DES is not as demanding as pure LES, thereby considerably cutting down the cost of the computation. Though DES was initially formulated for the Spalart-Allmaras model (Spalart et al., 1997), it can be implemented with other RANS models (Strelets, 2001), by appropriately modifying the length scale which is explicitly or implicitly involved in the RANS model. So while Spalart-Allmaras model based DES acts as LES with a wall model, DES based on other models (like two equation models) behave as a hybrid RANS-LES model. Grid generation is more complicated than for a simple RANS or LES case due to the RANS-LES switch. DES is a non-zonal approach and provides a single smooth velocity field across the RANS and the LES regions of the solutions.

Reynolds-averaged Navier–Stokes

Reynolds-averaged Navier-Stokes (RANS) equations are the oldest approach to turbulence modeling. An ensemble version of the governing equations is solved, which introduces new *apparent stresses* known as Reynolds stresses. This adds a second order tensor of unknowns for which various models can provide different levels of closure. It is a common misconception that the RANS equations do not apply to flows with a time-varying mean flow because these equations are 'time-averaged'. In fact, statistically unsteady (or non-stationary) flows can equally be treated. This is sometimes referred to as URANS. There is nothing inherent in Reynolds averaging to preclude this, but the turbulence models used to close the equations are valid only as long as the time over which these changes in the mean occur is large compared to the time scales of the turbulent motion containing most of the energy.

RANS models can be divided into two broad approaches:

Boussinesq hypothesis

> This method involves using an algebraic equation for the Reynolds stresses which include determining the turbulent viscosity, and depending on the level of sophistication of the model, solving transport equations for determining the turbulent kinetic energy and dissipation. Models include k-ε (Spalding), Mixing Length Model (Prandtl) and Zero Equation (Chen). The models available in this approach are often referred to by the number of transport equations associated with the method. For example, the Mixing Length model is a "Zero Equation" model because no transport equations are solved; the $k - \varepsilon$ is a "Two Equation" model because two transport equations (one for k and one for ε) are solved.

Reynolds stress model (RSM)

> This approach attempts to actually solve transport equations for the Reynolds stresses. This means introduction of several transport equations for all the Reynolds stresses and hence this approach is much more costly in CPU effort.

Coherent vortex simulation

The coherent vortex simulation approach decomposes the turbulent flow field into a coherent part, consisting of organized vortical motion, and the incoherent part, which is the random background flow. This decomposition is done using wavelet filtering. The approach has much in common with LES, since it uses decomposition and resolves only the filtered portion, but different in that it does not use a linear, low-pass filter. Instead, the filtering operation is based on wavelets, and the filter can be adapted as the flow field evolves. Farge and Schneider tested the CVS method with two flow configurations and showed that the coherent portion of the flow exhibited the $-\frac{5}{3}$energy spectrum exhibited by the total flow, and corresponded to coherent structures (vortex tubes), while the incoherent parts of the flow composed homogeneous background noise, which exhibited no organized structures. Goldstein and Oleg applied the CVS model to large eddy simulation, but did not assume that the wavelet filter completely eliminated all coherent motions from the subfilter scales. By employing both LES and CVS filtering, they showed that the SFS dissipation was dominated by the SFS flow field's coherent portion.

PDF methods

Probability density function (PDF) methods for turbulence, first introduced by Lundgren, are based on tracking the one-point probability density function of the velocity, $f_V(\boldsymbol{v};\boldsymbol{x},t)d\boldsymbol{v}$, which gives the probability of the velocity at point $\boldsymbol{x}$being between $\boldsymbol{v}$and $\boldsymbol{v}+d\boldsymbol{v}$. This approach is analogous to the kinetic theory of gases, in which the macroscopic properties of a gas are described by a large number of particles. PDF methods are unique in that they can be applied in the framework of a number of different turbulence models; the main differences occur in the form of the PDF transport equation. For example, in the context of large eddy simulation, the PDF becomes the filtered PDF. PDF methods can also be used to describe chemical reactions, and are particularly useful for simulating chemically reacting flows because the chemical source term is unclosed and does not require a model. The PDF is commonly tracked by using Lagrangian particle methods; when combined with large eddy simulation, this leads to a Langevin equation for subfiler particle evolution.

Vortex method

The Vortex method is a grid-free technique for the simulation of turbulent flows. It uses vortices as the computational elements, mimicking the physical structures in turbulence. Vortex methods were developed as a grid-free methodology that would not be limited by the fundamental smoothing effects associated with grid-based methods. To be practical, however, vortex methods require means for rapidly computing velocities from the vortex elements – in other words they require the solution to a particular form of the N-body problem (in which the motion of N objects is tied to their mutual influences). A breakthrough came in the late 1980s with the development of the Fast Multipole Method (FMM), an algorithm by V. Rokhlin (Yale) and L. Greengard (Courant Institute) that has been heralded as one of the top ten advances in numerical science of the 20th century. This breakthrough paved the way to practical computation of the velocities from the vortex elements and is the basis of successful algorithms.

Software based on the Vortex method offer a new means for solving tough fluid dynamics problems with minimal user intervention. All that is required is specification of problem geometry and setting of boundary and initial conditions. Among the significant advantages of this modern technology;

- It is practically grid-free, thus eliminating numerous iterations associated with RANS and LES.
- All problems are treated identically. No modeling or calibration inputs are required.
- Time-series simulations, which are crucial for correct analysis of acoustics, are possible.
- The small scale and large scale are accurately simulated at the same time.

Vorticity Confinement method

The Vorticity confinement (VC) method is an Eulerian technique used in the simulation of turbulent wakes. It uses a solitary-wave like approach to produce stable solution with no numerical spreading. VC can capture the small scale features to over as few as 2 grid cells. Within these features, a nonlinear difference equation is solved as opposed to the finite difference equation. VC is similar to shock capturing methods, where conservation laws are satisfied, so that the essential integral quantities are accurately computed.

Two phase flow

The modeling of two-phase flow is still under development. Different methods have been proposed. The Volume of fluid method has received a lot of attention lately, for problems that do not have dispersed particles, but the Level set method and front tracking are also valuable approaches. Most of these methods are either good in maintaining a sharp interface or at conserving mass. This is crucial since the evaluation of the density, viscosity and surface tension is based on the values averaged over the interface. Lagrangian multiphase models, which are used for dispersed media, are based on solving the Lagrangian equation of motion for the dispersed phase.

Solution algorithms

Discretization in space produces a system of ordinary differential equations for unsteady problems and algebraic equations for steady problems. Implicit or semi-implicit methods are generally used to integrate the ordinary differential equations, producing a system of (usually) nonlinear algebraic equations. Applying a Newton or Picard iteration produces a system of linear equations which is nonsymmetric in the presence of advection and indefinite in the presence of incompressibility. Such systems, particularly in 3D, are frequently too large for direct solvers, so iterative methods are used, either stationary methods such as successive overrelaxation or Krylov subspace methods. Krylov methods such as GMRES, typically used with preconditioning, operate by minimizing the residual over successive subspaces generated by the preconditioned operator.

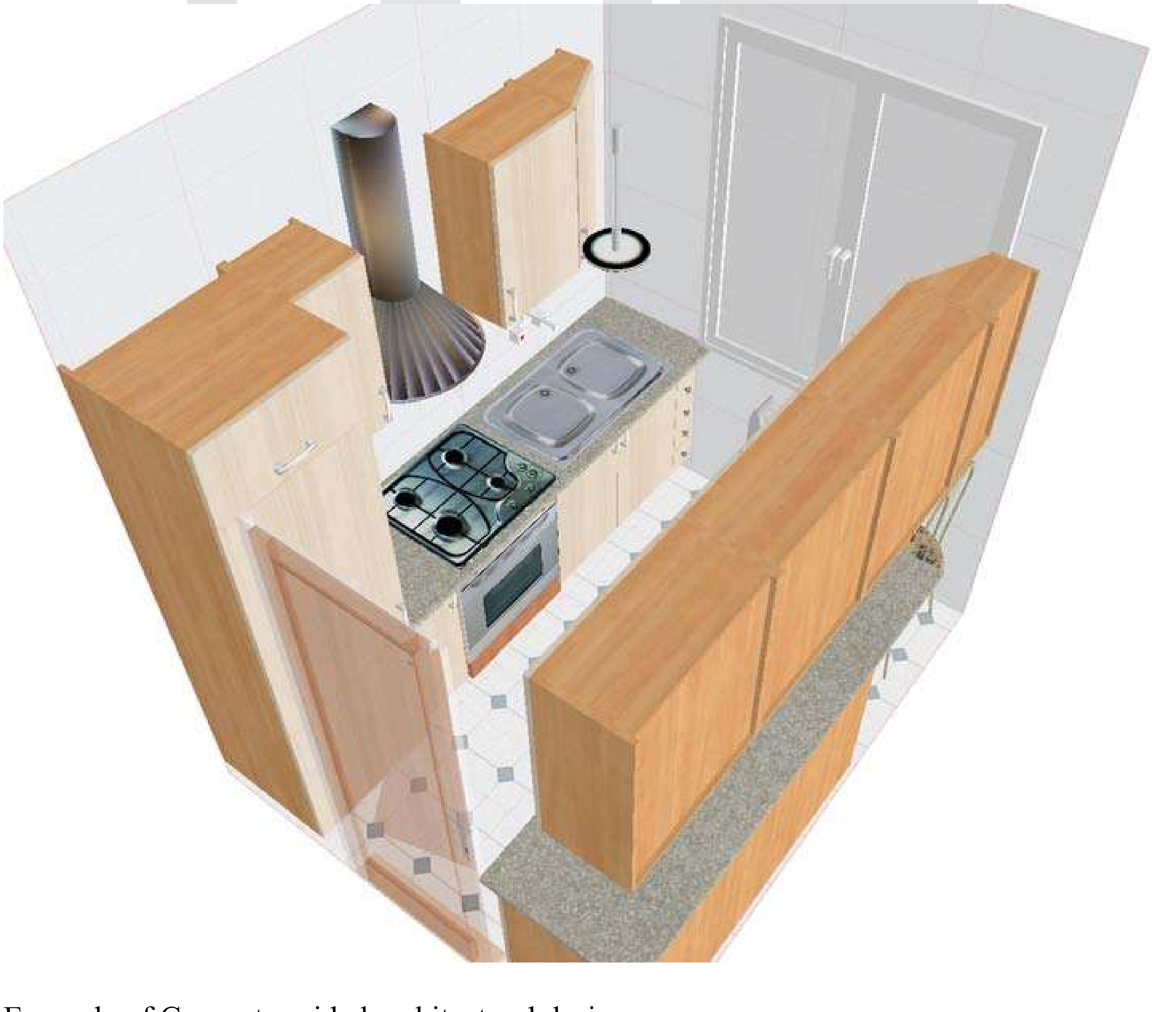
Multigrid is especially popular , both as a solver and as a preconditioner, due to its asymptotically optimal performance on many problems. Traditional solvers and preconditioners are effective at reducing high-frequency components of the residual, but low-frequency components typically require many iterations to reduce. By operating on multiple scales, multigrid reduces all components of the residual by similar factors, leading to a mesh-independent number of iterations.

For indefinite systems, preconditioners such as incomplete LU factorization, additive Schwarz, and multigrid perform poorly or fail entirely, so the problem structure must be used for effective preconditioning. The traditional methods commonly used in CFD are the SIMPLE and Uzawa algorithms which exhibit mesh-dependent convergence rates, but recent advances based on block LU factorization combined with multigrid for the resulting definite systems have led to preconditioners that deliver mesh-independent convergence rates.

Chapter- 9

Other Computer-Aided Technologies

Computer-aided architectural design



Example of Computer-aided architectural design

Computer-aided architectural design (CAAD) software programs are the repository of accurate and comprehensive records of buildings and are used by architects and architectural companies.

The first program was installed back in the 1960's, to help architects save time instead of drawing their blueprints. Computer-aided design also known as CAD was originally the type of program that architects used, but since CAD couldn't offer all the tools that architects needed to complete a project, CAAD developed as a distinct class of software.

Overview

All CAD and CAAD systems employ a database with geometric and other properties of objects; they all have some kind of graphic user interface to manipulate a visual representation rather than the database; and they are all more or less concerned with assembling designs from standard and non-standard pieces. Currently, the main distinction which causes one to speak of CAAD rather than CAD lies in the domain knowledge (architecture-specific objects, techniques, data, and process support) embedded in the system. A CAAD system differs from other CAD systems in two respects:

- It has an explicit object database of building parts and construction knowledge.
- It explicitly supports the creation of architectural objects.

In a more general sense, CAAD also refers to the use of any computational technique in the field of architectural design other than by means of architecture-specific software. For example, software which is specifically developed for the computer animation industry (e.g. Maya and 3DStudio Max), is also used in architectural design. The exact distinction of what properly belongs to CAAD is not always clear. Specialized software, for example for calculating structures by means of the finite element method, is used in architectural design and in that sense may fall under CAAD. On the other hand, such software is seldom used to create new designs.

In 1974 Caad became a current word and was a common topic of commercial modernization.

Three dimensional objects

CAAD has two types of structures in its program. The first system is surface structure which provides a graphics medium to represent three dimensional objects using two dimensional representations. Also algorithms that allow the generation of patterns and their analysis using programmed criteria, and data banks that store information about the problem at hand and the standards and regulations that applies to it. The second system is deep structure which means that the operations performed by the computer have natural limitations. Computer hardware and machine languages that are supported by these make it easy to perform arithmetical operations quickly and accurately. Also almost an illogical number of layers of symbolic processing can be built enabling the functionalities that are found at the surface.

Advantages

Another advantage to CAAD is the two way mapping of activities and functionalities.NOS. The two instances of mapping are indicated to be between the surface structures (TM1) and the deep structures (TM2). These mappings are abstractions that are introduced in order to discuss the process of design and deployment of CAAD systems. In designing the systems the system developers usually consider TM1. Here a one-to-one mapping is the typical statement, which is to develop a computer based functionality that maps as closely as possible into a corresponding manual design activity, for example, drafting of stairs, checking spatial conflict between building systems, and generating perspectives from orthogonal views. The architectural design processes tend to integrate models isolated so far. Many different kinds of expert knowledge, tools, visualization techniques, and media are to be combined. The design process covers the complete life cycle of the building. The areas that are covered are construction, operations, reorganization, as well as destruction. Considering the shared use of digital design tools and the exchange of information and knowledge between designers and across different projects, we speak of a design continuum.

An architect's work involves mostly visually represented data. Problems are often outlined and dealt with in a graphical approach. Only this form of expression serves as a basis for work and discussion. Therefore, the designer should have a maximum visual control over the processes taking place with in the design continuum. Further questions occur about navigation, associative information access, programming and communication with in very large data sets.

Computer-aided process planning

Computer-aided process planning (**CAPP**) is the use of computer technology to aid in the process planning of a part or product, in manufacturing. CAPP is the link between CAD and CAM in that it provides for the planning of the process to be used in producing a designed part.

Computer-aided process planning

Introduction

Process planning is concerned with determining the sequence of individual manufacturing operations needed to produce a given part or product. The resulting operation sequence is documented on a form typically referred to as a route sheet containing a listing of the production operations and associated machine tools for a workpart or assembly. Process planning in manufacturing also refers to the planning of use of blanks, spare parts, packaging material, user instructions (manuals) etc.

The term "Computer-Aided Production Planning" is used in different context on different parts of the production process; to some extent CAPP overlaps with the term "PIC" (Production and Inventory control

Process planning translates design information into the process steps and instructions to efficiently and effectively manufacture products. As the design process is supported by many computer-aided tools, computer-aided process planning (CAPP) has evolved to simplify and improve process planning and achieve more effective use of manufacturing resources..

Process planning encompasses the activities and functions to prepare a detailed set of plans and instructions to produce a part. The planning begins with engineering drawings, specifications, parts or material lists and a forecast of demand. The results of the planning are:

- Routings which specify operations, operation sequences, work centers, standards, tooling and fixtures. This routing becomes a major input to the manufacturing resource planning system to define operations for production activity control purposes and define required resources for capacity requirements planning purposes.

- Process plans which typically provide more detailed, step-by-step work instructions including dimensions related to individual operations, machining parameters, set-up instructions, and quality assurance checkpoints.

- Fabrication and assembly drawings to support manufacture (as opposed to engineering drawings to define the part).

Keneth Crow stated that "Manual process planning is based on a manufacturing engineer's experience and knowledge of production facilities, equipment, their capabilities, processes, and tooling. Process planning is very time-consuming and the results vary based on the person doing the planning".

According to Engelke , the need for CAPP is greater with an increased number of different types of parts being manufactured, and with a more complex manufacturing process.

Computer-aided process planning initially evolved as a means to electronically store a process plan once it was created, retrieve it, modify it for a new part and print the plan. Other capabilities were table-driven cost and standard estimating systems, for sales representatives to create customer quotations and estimate delivery time.

Future development of CAPP

Generative or dynamic CAPP is the main focus of development, the ability to automatically generate production plans for new products, or dynamicly update production plans on the basis of resource availability. Generative CAPP will probably use iterative methods, where simple production plans are applied to automatic CAD/CAM development to refine the initial production plan.

Traditional CAPP methods that optimise plans in a linear manner have not been able provide the need for flexible planning, so new dynamic systems will explore all possible combinations of production processes, and then generate plans according to available machining resources. For example, K.S. Lee et al. states that "By considering the multi-selection tasks simultaneously, a specially designed genetic algorithm searches through the entire solution space to identify the optimal plan" .

Computer-aided quality assurance

Computer-aided quality assurance (CAQ) is the engineering application of computers and computer controlled machines for the definition and inspection of the quality of products.

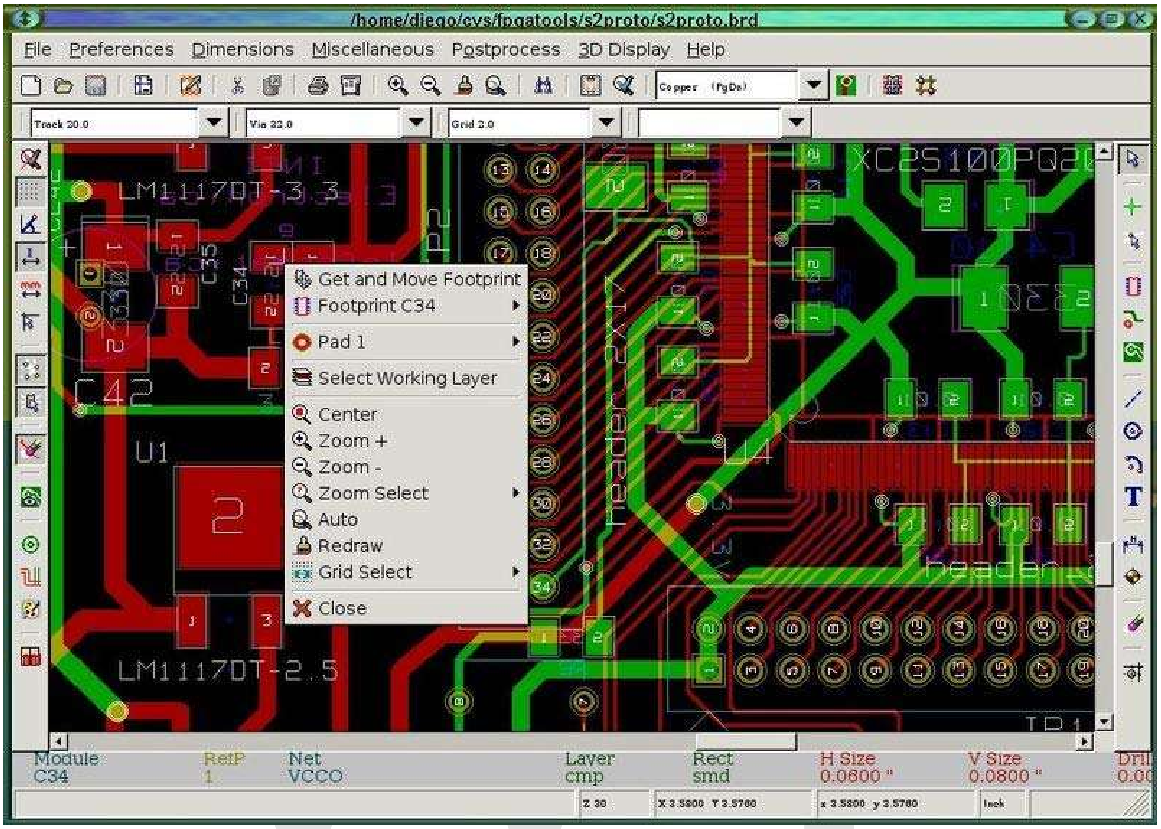
This includes:

- Measuring equipment management
- Goods inward inspection
- Vendor rating
- Attribute chart
- Statistical process control (SPC)
- Documentation

Additional themes:

- Advanced Product Quality Planning (APQP)
- Failure mode and effects analysis (FMEA)
- Dimensional tolerance stack-up analysis using product and manufacturing information (PMI) on CAD models
- Computer aided inspection with coordinate-measuring machines (CMM)
- Comparison of data obtained by mean of 3D scanning technologies of physical parts against CAD models

Electronic design automation



Track 20.0
Via 32.0
Grid 2.0
Copper (PgDn)

Module C34
RefP 1
Net VCCO
Layer cmp
Rect smd
H Size 0.0600 "
V Size 0.0800 "

Z 30
X 3.5800 Y 2.5760
x 3.5800 y 2.5760
Inch

PCB layout program

Electronic design automation (**EDA** or **ECAD**) is a category of software tools for designing electronic systems such as printed circuit boards and integrated circuits. The tools work together in a design flow that chip designers use to design and analyze entire semiconductor chips.

History

Early days

Before EDA, integrated circuits were designed by hand, and manually laid out. Some advanced shops used geometric software to generate the tapes for the Gerber photoplotter, but even those copied digital recordings of mechanically-drawn components. The process was fundamentally graphic, with the translation from electronics to graphics done manually. The best known company from this era was Calma, whose GDSII format survives.

By the mid-70s, developers started to automate the design, and not just the drafting. The first placement and routing (Place and route) tools were developed. The proceedings of the Design Automation Conference cover much of this era.

The next era began about the time of the publication of "Introduction to VLSI Systems" by Carver Mead and Lynn Conway in 1980. This ground breaking text advocated chip design with programming languages that compiled to silicon. The immediate result was a considerable increase in the complexity of the chips that could be designed, with improved access to design verification tools that used logic simulation. Often the chips were easier to lay out and more likely to function correctly, since their designs could be simulated more thoroughly prior to construction. Although the languages and tools have evolved, this general approach of specifying the desired behavior in a textual programming language and letting the tools derive the detailed physical design remains the basis of digital IC design today.

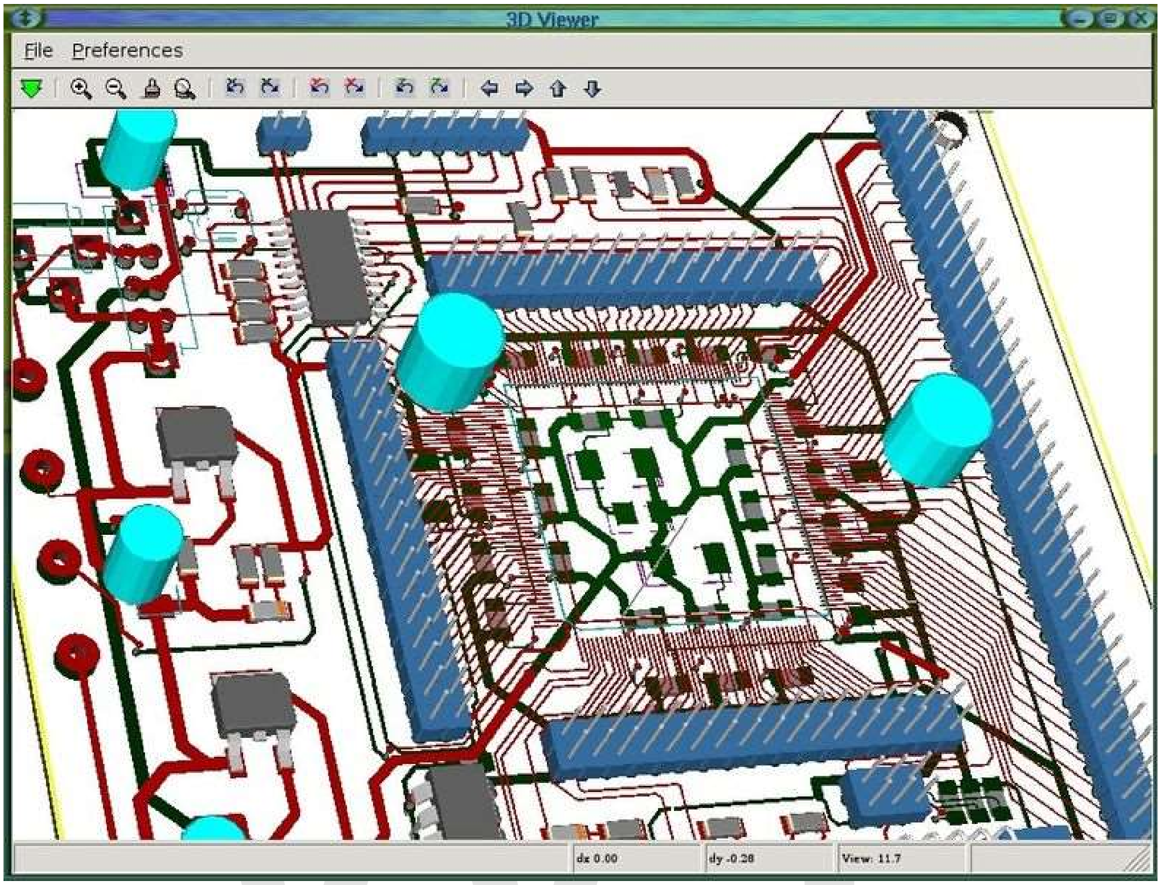
The earliest EDA tools were produced academically. One of the most famous was the "Berkeley VLSI Tools Tarball", a set of UNIX utilities used to design early VLSI systems. Still widely used is the Espresso heuristic logic minimizer and Magic.

Another crucial development was the formation of MOSIS, a consortium of universities and fabricators that developed an inexpensive way to train student chip designers by producing real integrated circuits. The basic concept was to use reliable, low-cost, relatively low-technology IC processes, and pack a large number of projects per wafer, with just a few copies of each projects' chips. Cooperating fabricators either donated the processed wafers, or sold them at cost, seeing the program as helpful to their own long-term growth.

Birth of commercial EDA

1981 marks the beginning of EDA as an industry. For many years, the larger electronic companies, such as Hewlett Packard, Tektronix, and Intel, had pursued EDA internally. In 1981, managers and developers spun out of these companies to concentrate on EDA as a business. Daisy Systems, Mentor Graphics, and Valid Logic Systems were all founded around this time, and collectively referred to as **DMV**. Within a few years there were many companies specializing in EDA, each with a slightly different emphasis. The first trade show for EDA was held at the Design Automation Conference in 1984.

In 1986, Verilog, a popular high-level design language, was first introduced as a hardware description language by Gateway Design Automation. In 1987, the U.S. Department of Defense funded creation of VHDL as a specification language. Simulators quickly followed these introductions, permitting direct simulation of chip designs: executable specifications. In a few more years, back-ends were developed to perform logic synthesis.



3D PCB layout

Current status

Current digital flows are extremely modular. The front ends produce standardized design descriptions that compile into invocations of "cells,", without regard to the cell technology. Cells implement logic or other electronic functions using a particular integrated circuit technology. Fabricators generally provide libraries of components for their production processes, with simulation models that fit standard simulation tools. Analog EDA tools are far less modular, since many more functions are required, they interact more strongly, and the components are (in general) less ideal.

EDA for electronics has rapidly increased in importance with the continuous scaling of semiconductor technology. Some users are foundry operators, who operate the semiconductor fabrication facilities, or "fabs", and design-service companies who use EDA software to evaluate an incoming design for manufacturing readiness. EDA tools are also used for programming design functionality into FPGAs.

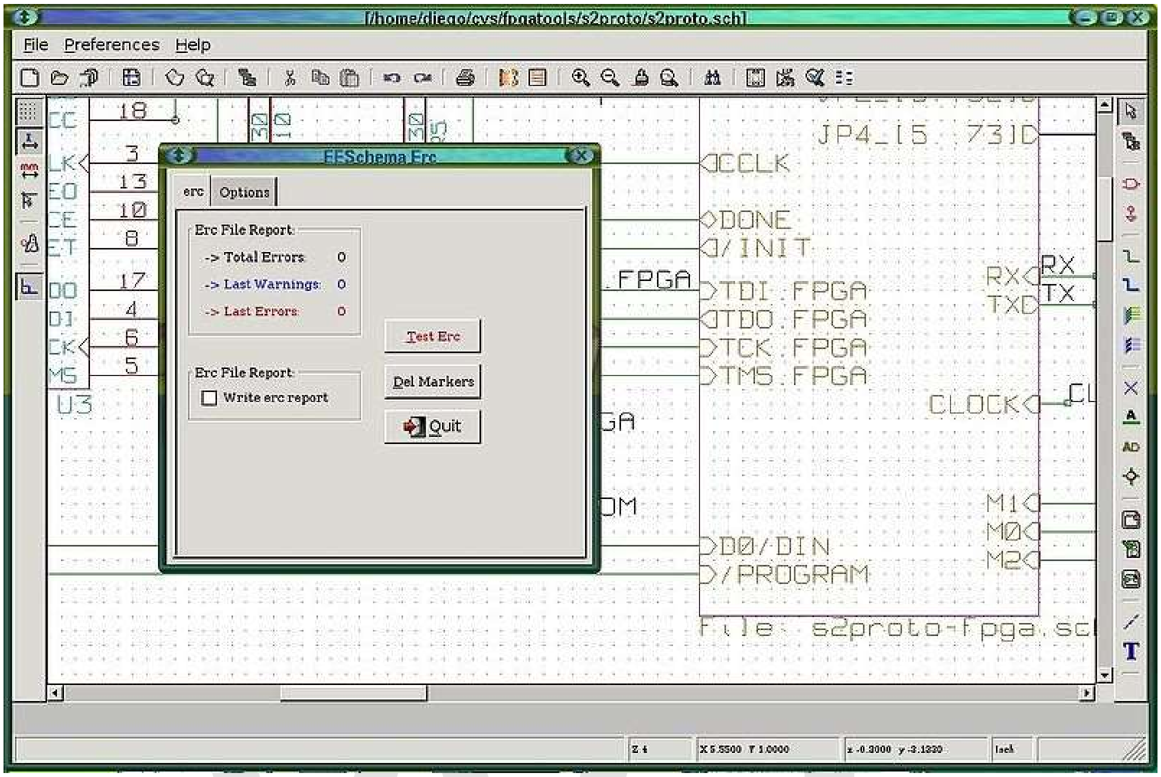
Software focuses

Design

- High-level synthesis(syn. behavioural synthesis, algorithmic synthesis) For digital chips
- Logic synthesis translation of abstract, logical language such as Verilog or VHDL into a discrete netlist of logic-gates
- Schematic Capture For standard cell digital, analog, rf like Capture CIS in Orcad by CADENCE and ISIS in Proteus
- Layout like Layout in Orcad by Cadence, ARES in Proteus

Simulation

- Transistor simulation – low-level transistor-simulation of a schematic/layout's behavior, accurate at device-level.
- Logic simulation – digital-simulation of an RTL or gate-netlist's digital (boolean 0/1) behavior, accurate at boolean-level.
- **Behavioral Simulation** – high-level simulation of a design's architectural operation, accurate at cycle-level or interface-level.
- Hardware emulation – Use of special purpose hardware to emulate the logic of a proposed design. Can sometimes be plugged into a system in place of a yet-to-be-built chip; this is called **in-circuit emulation**.
- Technology CAD simulate and analyze the underlying process technology. Electrical properties of devices are derived directly from device physics.
- Electromagnetic field solvers, or just field solvers, solve Maxwell's equations directly for cases of interest in IC and PCB design. They are known for being slower but more accurate than the layout extraction above.



Erc File Report:
-> Total Errors: 0
-> Last Warnings: 0
-> Last Errors: 0

Erc File Report:
☐ Write erc report

Schematic capture program

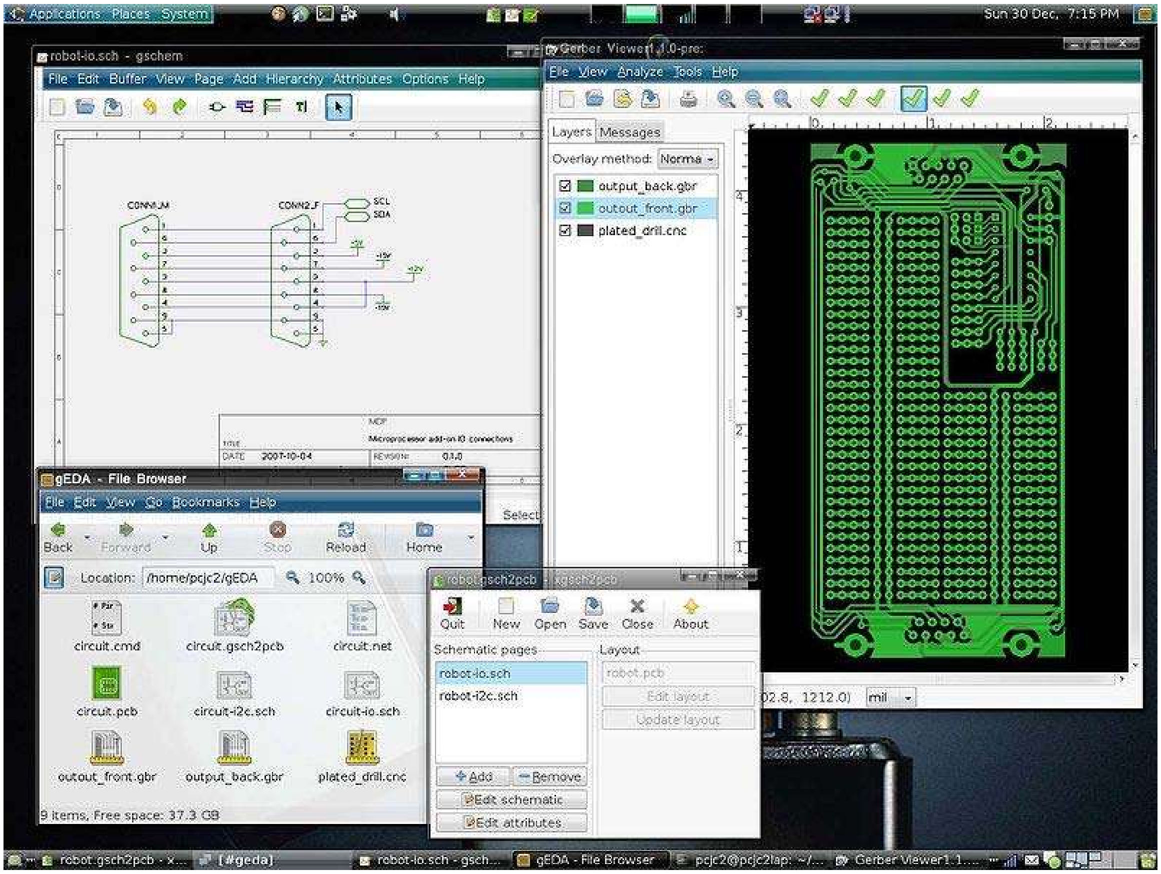
Analysis and verification

- Functional verification
- Clock Domain Crossing Verification (CDC check): Similar to linting, but these checks/tools specialize in detecting and reporting potential issues like data loss, meta-stability due to use of multiple clock domains in the design.
- Formal verification, also model checking: Attempts to prove, by mathematical methods, that the system has certain desired properties, and that certain undesired effects (such as deadlock) cannot occur.
- Equivalence checking: algorithmic comparison between a chip's RTL-description and synthesized gate-netlist, to ensure functional equivalence at the *logical* level.
- Static timing analysis: Analysis of the timing of a circuit in an input-independent manner, hence finding a worst case over all possible inputs.
- Physical verification, PV: checking if a design is physically manufacturable, and that the resulting chips will not have any function-preventing physical defects, and will meet original specifications.

Manufacturing preparation

- Mask data preparation, MDP: generation of actual lithography photomask used to physically manufacture the chip.
    - Resolution enhancement techniques, RET – methods of increasing of quality of final photomask.

- Optical proximity correction, OPC – up-front compensation for diffraction and interference effects occurring later when chip is manufactured using this mask.
- Mask generation – generation of flat mask image from hierarchical design.
- Automatic test pattern generation, ATPG – generates pattern-data to systematically exercise as many logic-gates, and other components, as possible.
- Built-in self-test, or BIST – installs self-contained test-controllers to automatically test a logic (or memory) structure in the design



PCB layout and schematic for connector design

Companies

Top companies

- $3.73 billion - Synopsys
- $2.06 billion - Cadence
- $1.18 billion - Mentor Graphics
- $233 million - Magma Design Automation
- $157 million - Zuken Inc.

Note: Market caps current as of October, 2010. EEsof should likely be on this list, but does not have a market cap as it is the EDA division of Agilent.

Acquisitions

Many of the EDA companies acquire small companies with software or other technology that can be adapted to their core business. Most of the market leaders are rather incestuous amalgamations of many smaller companies. This trend is helped by the tendency of software companies to design tools as accessories that fit naturally into a larger vendor's suite of programs (on digital circuitry, many new tools incorporate analog design, and mixed systems. This is happening because there is now a trend to place entire electronic systems on a single chip.

Manufacturing process management

Manufacturing process management (**MPM**) is a collection of technologies and methods used to define how products are to be manufactured. MPM differs from ERP/MRP which is used to plan the ordering of materials and other resources, set manufacturing schedules, and compile cost data.

A cornerstone of MPM is the central repository for the integration of all these tools and activities aids in the exploration of alternative production line scenarios; making assembly lines more efficient with the aim of reduced lead time to product launch, shorter product times and reduced work in progress (WIP) inventories as well as allowing rapid response to product or product changes.

Topics and technology

- Production process planning
    - Manufacturing concept planning
    - Factory layout planning and analysis
        - work flow simulation.
        - walk-path assembly planning
        - plant design optimization
    - Mixed model line balancing.
    - Workloads on multiple stations.
    - Process simulation tools e.g. die press lines, manufacturing lines
    - Ergonomic simulation and assessment of production assembly tasks
    - Resource planning
- Computer-aided manufacturing (CAM)
    - Numerical control CNC
    - Direct Numerical Control (DNC)
    - Tooling/equipment/fixtures development
    - Tooling and Robot work-cell setup and offline programming (OLP)

- Generation of shop floor work instructions
- Time and cost estimates
    - ABC - Manufacturing activity-based costing
    - Production, costs, and pricing
- Quality Computer-aided quality assurance (CAQ)
    - FMEA Failure mode and effects analysis
    - SPC Statistical process control
    - Computer aided inspection with coordinate-measuring machine (CMM)
    - Tolerance stack-up analysis using PMI models.
- Success Measurements
    - Overall Equipment Effectiveness (OEE),
- Communication with other systems
    - Enterprise resource planning (ERP)
    - Manufacturing Operations Management (MOM)
    - Product Data Management (PDM)
    - SCADA (Supervisory Control and Data Acquisition) real time process monitoring and control
    - Human-machine interface (HMI) (or *man-machine interface* (MMI))
    - Distributed control system (DCS)