IntegrationPartners, Inc.

Integrating People, Process, & Technology

By Paul Lorson

INTRODUCTION AND SUMMARY

A company's efforts toward integrating its activities are typically focused initially on computerizing a particular function or organization such as manufacturing, or engineering. The productivity improvements normally achieved with this initial implementation range from zero to twenty percent. Many companies have reached this plateau of benefits and are searching for further opportunities to use computer technology in competitive or strategic ways. Integration Partners, Inc. (IPI) has found that computer technology, when applied in new and innovative ways, can enable changes to work processes and the value people add during those processes. These changes to people, process, and technology allow you to completely redefine your strategies and ultimately your position in your markets. This paper will focus on some of the changes and methods that enable these results.

During the 70's the concerns for diminishing market share, and decreasing quality forced us to evaluate our business practices and those of our competition. Most companies focused on the integration of people. Work groups were formed, management by committee ideas were implemented. Management teams were sent to classes on group dynamics and understanding motivation. The promise of 20 to 30% improvement in productivity and a "Happier more fulfilling" work environment was heard. As we diligently pursued these goals no revolutionary changes occurred, we achieved minimal improvement and we watched as our competition improved. We had gained no real competitive advantage.

So we looked for something else, something that would produce greater results and we turned to technology. Yes, HP produced the world's fastest computers. We could now do more work faster and throughout the 80's and 90's technology grew. CAD became prevalent, drawing boards vanished, MRP and JIT, Databases Relational and Object Oriented, and Information Systems became everyday terms. The promise of 20 to 30% improvements in productivity were heard along with "higher quality". As we diligently pursued these goals and achieved some improvement we watched as our competition improved and we gained no real competitive advantage.

In the 90's we began another arduous journey. Re-evaluating our people and "right sizing", re-evaluating our technology and upgrading but still no competitive advantage and then we saw our savior "process re-engineering". The promise of 20 to 30% productivity improvement, EPD and PDM were tossed into our vocabulary. And as we progressed and made some improvement so did the competition and we gained no real competitive advantage.

So we have looked at our people and we made some improvement. We have invested in technology and we saw some improvement. We have looked at our processes and we felt some improvement, but few have achieved our necessary improvements and obtained a real competitive advantage. If its not our people, and we have the technology, and we have improved our process why are we not achieving our desired improvements?

The answer can best be summarized as “Integration”. The bringing together all the necessary knowledge, experience, technologies, and processes so that decisions can be made early in the product life cycle. Integration allows for more engineering faster, manufacturing and inventory constraints can be utilized earlier and production increased.

Increased productivity of 20 to 30% right? Wrong!!! After implementing an integrated solution Mr. Tony Futo of Ahlstom Pyropower Corporation reported an “increase in productivity of 150% in Design & Drafting operation, while field rework on our projects has been reduced more that twenty-fold”.

IPI’s approach provides solutions to issues facing engineering, manufacturing, automation and information system managers. Some of these issues are answered through questions such as: ‘How can the knowledge of human experts be leveraged and applied through computerized tools?’ ‘How can Knowledge-Based Expert Systems (KBES) be made compatible with my other tools?’ ‘How can several different technologies be merged in single applications’?

Traditional System Architectures Fail to Unleash Major Benefits

The architecture of a typical computer application is shown in Figure 1. The user interacts with the software through a user interface that provides a window to the domain logic and data structures of the application. The results of this interaction are stored in a database of some kind. These application building blocks are wrapped up in an environment that is proprietary to the specific technology or system in use. The designer or engineer must learn the appropriate rules and syntax of operation to accomplish their assigned tasks. This application architecture is commonly used in many proprietary environments such as CAD applications, manufacturing applications, database applications, knowledge-based engineering applications, etc.

Although this configuration is widely utilized, it has the following disadvantages:

1. If the proprietary technology changes significantly or becomes obsolete, the application must be re-encoded in a new language that comes with the new system. Often this means an entire redesign of the application to fit the unique strengths and weaknesses of the new system.

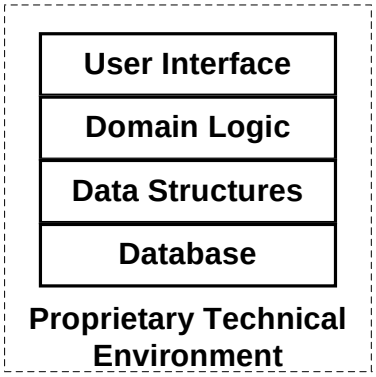

Figure 1
Traditional Architecture for Simple Applications

2. Proprietary systems and applications are usually focused around a single technology (CAD, database, KBE, MRP, etc.), and do not easily allow multiple technologies to interact in ways transparent to the user. This means that CAD systems do CAD functions well, but are not good KBE or database engines. Conversely, KBE systems do KBE functions well, but are not good CAD or database engines. The user is forced to choose a dominant technology and a system arrangement that do not fully meet all their needs. User's must often spend time re-entering or transforming data from one environment to another.

3. The user is requested to engage in fairly repetitive operational interactions with the computer application that slow down the design process. The ultimate speed at which a design is completed is not related to the high processing speed of the computer or the design skills of the user; it is dictated rather by the relatively slow speed of the man-machine interactions required by the nuances and needs of the computer.

The result of these problems, imposed by the computer and its application architecture is less than optimal performance by both the computer and the person, and, consequently, yields less than optimal technical solutions. The computer is often in an idle state waiting to receive a command, and the users spend most of their time formulating commands and calculations to instruct the computer what to do next. Too much time is spent in routine, repetitive activities and not enough time is spent in value-added activity.

Traditional Architectures Fail to Meet Work Process Automation Needs of Multi-Technology Environments

Before we can proceed to explore the full extent of advantages associated with a new computer application architecture, we need to look at other limitations of traditional architectures that emerge when applied to multi-technology environments. As shown in Figure 2, the traditional method of architecting complex applications that use multiple dissimilar technologies, is through a series of interfaces that allow data passing as a means of interaction between the technologies involved. For example, a database technology performs data processing functions of the overall application, a CAD system performs the geometric type functions of the application, and a knowledge-based engineering system or shell performs the rule processing functions of the application. This traditional architecture, however, has some disadvantageous characteristics:

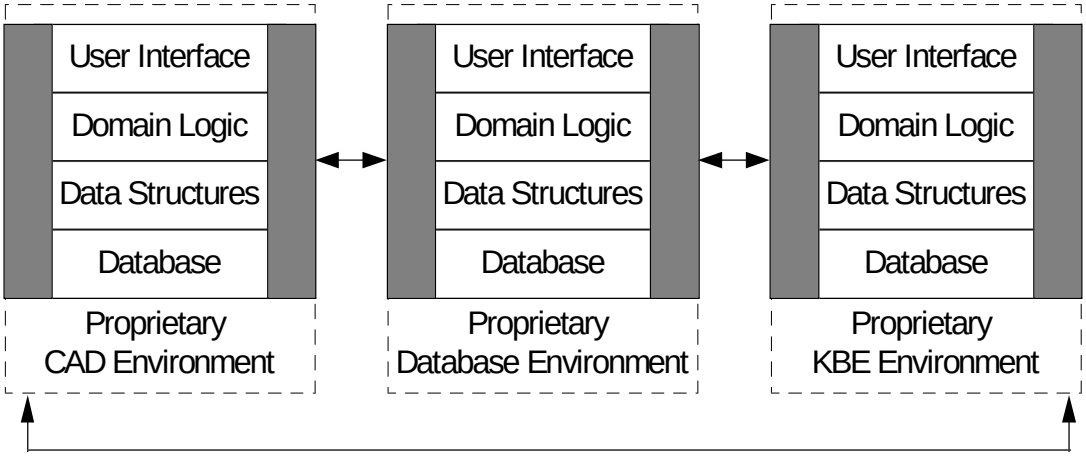

Figure 2
Traditional Architecture For Complex Applications

1. Each program requires input/output interfaces with each one of the other technologies. With 'n' being the number of technologies to be interfaced within a single application, the number of input/output interfaces is equivalent to n(n-1). Thus the programming effort required to make the system operational grows exponentially with the number of technologies involved. Maintenance of the interfaces becomes a large and complex effort.

2. Duplication is required in each technological module so it will be fully compatible with the overall purpose of the application. This underlying redundancy extends to the logic, the data structures, and the data stored in each technological module. This duplication creates an inordinate amount of cross checking between the different technologies to ensure that they all remain compatible with each other and to ensure data integrity.

3. The traditional approach to interfacing multiple technologies is difficult to change and maintain as new technologies or generations of technology become available. Because of the high cost of making changes to complex applications and the interdependency between the technological modules, adding a new technology or removing an obsolete one is very difficult. Consequently, companies tend to work with the old technologies, until competitive factors force them to upgrade. Then, often the cycle of technology development and implementation starts over from the beginning, at a significant cost.

These problems of duplication, complexity of interfaces, high costs to create and maintain the application and lack of flexibility pose significant challenges when using computer technologies to enable work process changes and improvements. Thus, traditional application architectures actually impede a company who is working to move up to the next plateau of benefits from computer technology.

So far, we have uncovered two major problems with the traditional approaches to implementing computer applications in the engineering environment. First, people spend most of their time doing routine or repetitive tasks using syntax-laden man-machine interfaces, and very little time engaged in truly value-adding activities. Second, applications that are developed as tools in the engineering environment are often expensive to create and maintain, and difficult to incrementally take advantage of new technologies and techniques as they become available. To solve these problems, Integration Partners embraces two concepts of utmost importance. First, automation should be applied at the process level, not at the task level, and second, a new application architecture is required that enables the integration of technology, work processes and people to achieve the results that are required in today's competitive global markets.

Advanced Applications Must Consider the Automation of Work Processes.

IPI staff have analyzed the interfaces of many different technologies, systems and applications, and have found that automated work processes free the user from the difficulties associated with proprietary syntax .

The instructions provided by the computer operator can be divided into a group of low level instructions and a group of high level instructions as shown in Figure 3. Low level decisions are characterized by exact syntax, repetitive reasoning or mundane calculations. This group of decisions and input instructions represent by far the majority of the time spent by a user working with a computer program; in many cases, up to 85%. High level decisions are characterized by creativity, ambiguity and heuristic reasoning processes. This often accounts for less than 10% of the time spent by the user when working with a computer program. Typically, a small amount of time is used by the computer for actually processing the instructions given to it.

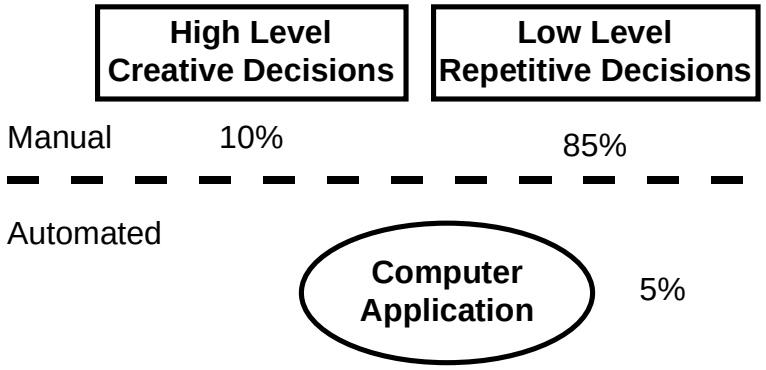

Figure 3
Task-Oriented Automation

As shown in Figure 4, the low level decisions can be encoded and made a part of the computer-aided automation. Only the work requiring the application of professional skills are left with the operator, while the rest of the work is handled by the computer.

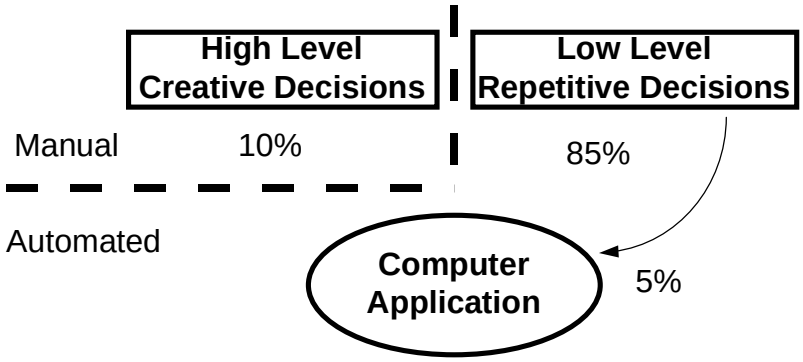

Figure 4
Process Oriented Automation

Figure 5 is a logical diagram that introduces the concept of a new computer application architecture that facilitates this higher level of automation.

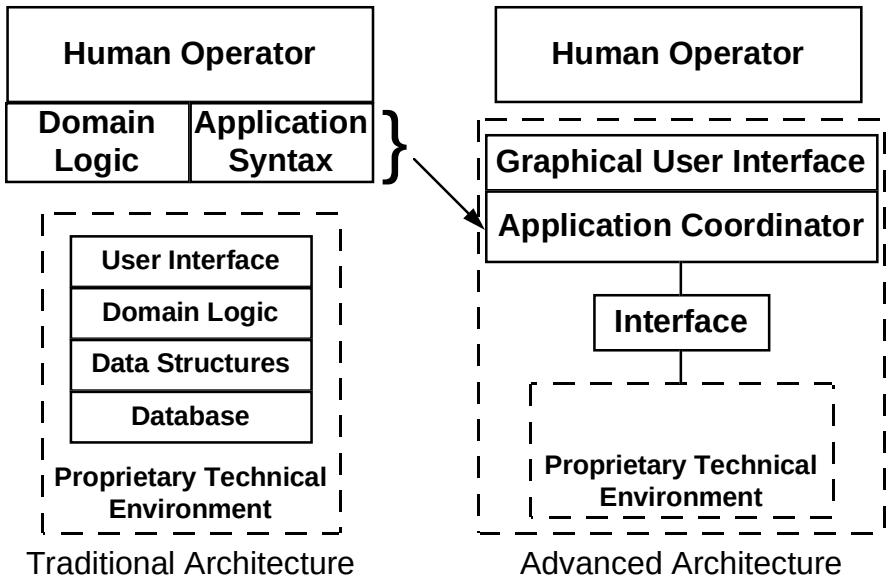

Figure 5
Advanced Application Architecture

In the traditional architecture, the user has detailed knowledge regarding the commands to give the computer, and how to apply them to get the desired results. These are referred to as Application Syntax and Domain Logic, respectively, in Figure 5. They represent the "Low Level Repetitive Decisions" shown in the previous figures. In the advanced application architecture, the Application Logic and Domain Logic are encoded into an object-oriented module called the Applications Coordinator. The programming language used in the Applications Coordinator may be a publicly available environment such as C or C++, or a proprietary object-oriented environment such as those offered by Design Power's D++, Gensym's G2 objects or NASA's CLIPS objects.

The Applications Coordinator is interfaced with the proprietary technical environment through an interface that passes instructions to be executed automatically. IPI has developed several such interfaces to work with diverse technologies such as CAD, KBE, database, and analysis programs. In each case, the syntax and logic of the application are automated and the proprietary technology is directed or instructed what to do by the Application Coordinator. There are several advantages of this architecture.

1. The computer performs most of the routine and redundant activities associated with the application. Computers are very good at performing the same task over and over. They do it fast with a high degree of accuracy and reliability. When the process is automated, the sequence of events, the required calculations and the necessary syntax are pre-determined either through direct instruction sets or through deterministic rules that can follow predictable reasoning patterns.

2. The person controlling the application performs high value-added activities associated with the application. The human operator is interacting with the new system at a higher level of the entire process. Routine reasoning has been encoded in the form of rules and constraints and live in the Coordinator. The operator does not have to deal with syntax anymore; he is focusing his attention to higher level decision making, i.e., those decisions that lead to innovative design improvements. One may say that, in the new system approach, the computer controls itself and keeps the work process in motion at a high speed. The human operator controls the system from a higher level with much less interaction than before. The human expert

is not held back anymore by the difficulty of interacting with the computer.

3. In the traditional architecture, most of the programming for applications is done in a proprietary environment using fourth generation programming languages. In the new applications architecture, a greater degree of automation is achievable using third generation programming languages. This allows greater flexibility in what the application can do and helps prolong the life of the application beyond the life of any single proprietary environment.

4. The new architecture enables computer applications that offer over-all ease of use and high productivity. The work process is completed much faster and the opportunity of operator error is much reduced; the system lends itself to the study of extreme design options, where each option is executed in direct similarity to the other options. As a rule of thumb, design processes that used to take days and weeks, are now accomplished within hours. The work process can be completed in as little as 10% of the previous time. Under these conditions, the human designer now has a tool to really explore innovative design options and can exceed the customer's expectations by delivering superior design value.

Flexible Software Application Architecture for Multiple-Technology Environments

The new application architecture also presents some advantages over the traditional architecture in those instances where it is desirable for multiple technologies to work together in a single application. This situation is illustrated in Figure 6.

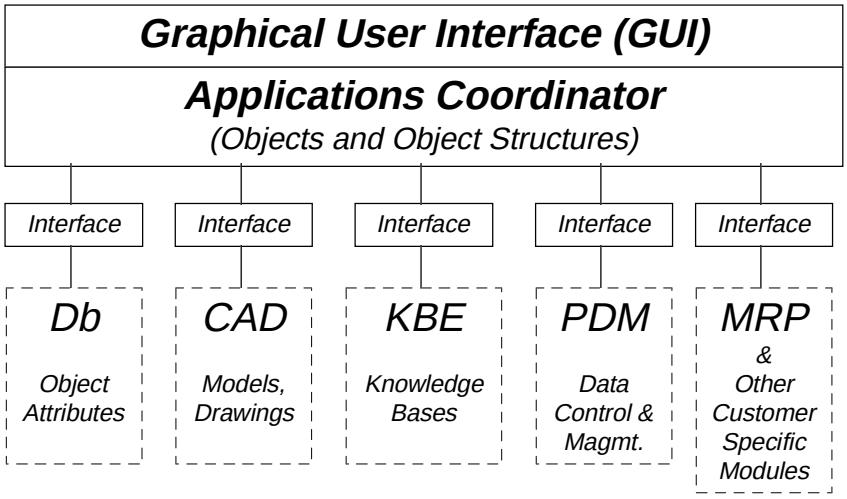

Figure 6
Multi-Technology Application Architecture

The system of several individual technologies has a single Applications Coordinator that forms a "common ground" for all technologies included in the application. A single interface module for transferring data and information in both directions is sufficient for each technology independent of the system size and the number of technologies included in the system.

The technologies work smoothly side-by-side at the applications level independent of their inherent differences. For example, the database is presented here as an external database for executing database related tasks; it is easily accessible by the applications coordinator for the purpose of transferring data as needed to and from the data processing modules. The single database is easily maintained and updated, and it

is easily accessed for other administrative purposes as well. The system may include one or more computer-aided design modules for executing design related tasks, one or more computer-aided engineering modules for executing engineering analysis related tasks, one or more modules for executing data management & control related tasks, one or more simulation modules for executing simulation related tasks, or one or more KBE modules for executing tasks requiring rule-based reasoning. The Applications Coordinator is the driver in this advanced system just as the operator who previously performed this function. The application driver allocates tasks to the best suited key technology module through the shared data structures and the interprocess interface.

The Applications Coordinator is the heart of the proposed system configuration. It represents a custom program that represents a company's design process, its know-how, and its expertise. The program is written in either an object-orient style with C or C++, or one of the available rules or constraint-based languages. Designers communicate with the Applications Coordinator through a Graphical User Interface (GUI) arranged in the Motif or another user defined style. Designers fill in the blanks of these forms to create new designs.

The key to the majority of benefits of the proposed system architecture stems from the fact that the operator is now free to focus on the design job at hand from a higher more creative level instead of wasting his valuable time repeatedly on syntax or other procedures needed only by the computer.

While the Applications Coordinator utilizes an expert system to perform its reasoning tasks and to manage its communication with the various programs, other expert systems may be represented in the system to perform specific intelligent functions such as checking the design for design interferences.

The application architecture permits the various technologies to be exchanged without losing the encoded design processes in the Applications Coordinator. If it is decided to replace one or more of the technologies, only the relatively small interface programs need to be rewritten and the encoded company know-how remains untouched during the change-over.

Writing custom software to automate routine design work offers large returns on investment. Computer programs can drive proprietary systems much faster than human operators. The principle to keep in mind is to let computers automatically control redundant or repetitive tasks, while the creative task is left to people.

The proposed applications architecture together with its Applications Coordinator was made feasible by the advent of two distinct technologies: First, object-oriented programming languages and techniques facilitate a higher level of abstraction when planning a computer application. One is not limited to the constraints of procedural programming, but rather objects can be used to perform random actions within boundaries much the same as a person might do in a typical design session. Second, graphical user interfaces (GUI's) allow people to interact with the objects and the object structure in a non-procedural fashion. Code generators called GUI Builders reduce the time required to prepare these interfaces productively for a highly customized application. Previously, because of the high cost of creating them, GUI's were limited to applications intended to be sold in mass markets performing very general tasks. Now, with GUI Builders, highly customized interfaces can be created relatively quickly and easily.

The advantages of the proposed expert applications architecture may be summarized as follows:

1. Computerization is focused on automating the work process instead of performing small disjointed tasks. This enables vast improvements to productivity, quality, and schedule performance. These create competitive advantages not otherwise attainable.

2. Dissimilar technologies work smoothly together at the application level. Each technology is used for those tasks they perform best.

3. Technologies in this architecture can more easily be added and/or exchanged. Since proprietary programming does not take place in proprietary environments, the separate technology environments can be swapped without redesigning and re-coding the entire application. The logic of the program remains intact long beyond the useful life of any single proprietary environment or technology.

IPI's Integrated Solutions have been Used Successfully in Diverse Situations

Integration Partners accepts a rather broad definition of expert systems that includes any system or technology able to perform complete work processes in a manner that frees the human expert from the mundane tasks associated with syntax or other steps they want performed. This new architecture allows integrated expert systems to be developed using a plurality of technologies in diverse domains, achieving a wide range of benefits.

In one instance, a CAD system, database system, KBE system, and selected proprietary technologies have been used to create expert systems that design industrial boilers used for steam generation on commercial scale power plants. Another, uses the same technologies, embodied by completely different proprietary products to design regenerative oxidizers used as pollution control equipment. A third application uses C++ modules, analysis software, and a database system to perform chemical and thermodynamic design and engineering. In all cases, these uses of technology have enabled companies to focus on business issues instead of computer tool issues. The results range from highly accelerated productivity and schedules, to increased product quality and market share.

IPI's experience with larger system applications indicates that design jobs requiring weeks or even months, can be accomplished within a few hours or even less than an hour depending on the particular application and the effort one is prepared to invest in capturing and programming the knowledge of the operator. The payback periods for system development, technology acquisition, and system implementation typically range from 6 to 18 months. However, the greatest benefits are not found in simple man-hour savings. The greatest benefits are found in the enabling changes to work processes and the human resources which are now focused on adding value to the customer and the company rather than diverted by unimportant peripheral tasks.

Integrating People, Process, & Technology a true competitive advantage.