AN INTRODUCTION TO INNOVATION ADOPTION THEORIES FOR THE DESIGN
OF A PRODUCT FAMILY INFORMATION MANAGEMENT INFRASTRUCTURE

Bethany M. Byron
Graduate Student, Mechanical Engineering
Bucknell University
Lewisburg, PA 17837
bbyron@bucknell.edu

Steven B. Shooter
Department of Mechanical Engineering
Bucknell University
Lewisburg, PA 17837
shooter@bucknell.edu

ABSTRACT

As new computer technologies emerge for the design and development of product platforms, applied theories of innovation diffusion and technology acceptance will become increasingly important for securing user cooperation with an information management system for product families. Incorporation of human factors therein will shorten the time required to reach return on investment and increase the depth and breadth of information. The history of innovation adoption theories are explored and extended to include commercially available enterprise management software and the proposed product family infrastructure. Six practices, which can be included in the infrastructure in its experimental stages, are described for improving user likelihood of adoption.

Keywords: diffusion of innovation, information management systems, product platforms

1. INTRODUCTION

The product development industry faces broad challenges to shorten design and manufacturing schedules and facilitate the cooperation of geographically dispersed teams [1]. Greater pressure is placed on effective workflow and information accessibility in order to meet these challenges. Computers have been long been used to model singular design solutions, for instance, launching one software package to create a detailed solid model and another to run a finite element solution. A third, still, might be required to evaluate dynamic loading effects or model fluid interaction. However, the field of computers in design is changing, favoring new technologies that maximize the capture, storage, retrieval, and reuse of design information embedded in non-traditional sources.

Experienced engineers, themselves, are a tremendous asset to their companies for the expertise they wield [2]. Knowledge management has become a huge topic for business strategies and information technology professionals, alike [3]. Engineers may transmit this information through informal channels like email, memos, and conversation. Powerful information about products is also inserted into technical documents; the geometry of every object communicates a message about its function. Though this information is valuable, it has not been harnessed in any standardized way. The door is open to the younger generation for finding innovative and intuitive ways to accomplish this. Because much of the American workforce will retire soon, reclaiming the knowledge of experienced engineers is quickly becoming an industry priority [4].

One efficient way to reuse design knowledge is by leveraging product platforms. Robertson and Ulrich define a product platform as “the collection of assets [i.e., components, processes, knowledge, people and relationships] that are shared by a set of products” [5]. They point out that “by sharing components and production processes across a platform of products, companies can develop differentiated products efficiently, increase the flexibility and responsiveness of their manufacturing processes, and take market share away from competitors that develop only one product at a time” [5]. An effective product platform strategy puts stress on engineers, though, to recognize and capitalize on those elements. It requires both hindsight to examine previously successful products and foresight to predict the shape and appeal of similar products in the future.

There is a large potential for increased productivity in the intersection between knowledge management and product platforms. An information management infrastructure for product families, then, will appropriate scattered tacit knowledge and assemble it into specialized data structures suitable for identifying and carrying out product platform
strategies. With time, the base of knowledge grows, creating deeper and more meaningful relationships between physical products, their intended functions, and the processes used to create them. The computational demands are immense, though. Work is needed at each of the three system stages: unobtrusive knowledge capture, intelligent information storage, and efficient search and retrieval. Inside each topic are a number of equally important and interrelated issues to address.

A decision must be made on what information is necessary to characterize a product and its relationship to the family, specifically, what attributes (i.e., material and volume), interfaces (i.e., attachments and flows), and functions must be recorded. The next question to address is the requisite degree of similarity that must exist between the design at hand and existing designs before it is recommended to the engineer. Finally, the infrastructure, as mapped out on an information flow model and relevancy matrices, must be operationalized into a graphical user interface (GUI) that will receive raw data from the user and distribute it to the appropriate module for further processing. The GUI must also be capable of interpreting processed data and returning an understandable display results to the user [6].

The realization of benefits from the system requires more than just the work of ingenious academians and software developers, however; the user plays a vital part in setting up the system to perform as it is intended. Many thoroughly documented, industry-specific designs are necessary to stock the design repository with the breadth and depth necessary to make accurate recommendations based on product (or subassembly) function. That product information must be supplied by the users. Optimal performance of the system requires widespread cooperation from all members of the project team to ensure the comprehensive capture of engineering and marketing information, as both are necessary to successfully carry out a product platform strategy. Going only half way, failing to secure widespread user support, has plagued many enterprise-wide software installations in the past, so much so that the Enterprise Resource Planning (ERP) and Product Lifecycle Management (PLM) software industries have a reputation for poor return on investment (ROI) [7].

Therefore, promoting user acceptance of the new computer-assisted product family design process is of the utmost importance. Without an intelligent interface, user resistance will hamper the transfer of knowledge into the system and further hinder effective use, rendering volumes of information a useless, disorganized pile. User acceptance is considered now, in the earliest stages of infrastructure development, because deploying a user-friendly interface is more difficult than screwing one onto a disorderly system just before its completion. A convoluted system will reveal its true nature to users in no time. Rather, by incorporating human-centeredness into the design of the infrastructure now, these traits will naturally flow out during the software development stages and ultimately promote user acceptance and project success.

This paper will examine the popular theories in innovation and technology adoption that will influence the human-computer interface of the upcoming infrastructure for product families. The field has, for a long time, debated within itself over the need for concise terminology and validation. Unfortunately, by turning inward, it has not seen much proliferation into the areas of design where it is particularly needed. Section 3 details six best practices for the assimilation of human factors into the early design of the infrastructure. The objective of this paper is to open the discussion to the design community, as it is related to the human-computer interface of a potentially powerful, but admittedly risky, emergent paradigm in design engineering and computing.

2. THE FOUNDATIONS OF THE DIFFUSION OF INNOVATION THEORY

The distribution of any innovation, whether it is a physical product, process, or ideology, has been likened to the diffusion of one liquid through another, gradually exposing the entire volume to the new element. According to Rogers [8], all exposed individuals must make a decision about whether to accept or reject the innovation. For some, the decision is instantaneous, but for others, the process is long, requiring deeper investigation of the innovation and its predicted outcomes. The innovation-decision process is formally defined as “the process through which an individual (or other decision-making unit) passes from first knowledge of an innovation to forming an attitude toward the innovation, to a decision to adopt or reject, to implementation and use of the new idea, and to confirmation of this decision” [8].

Rogers [8] also published a set of five attributes he identified to help predict when and where adoption occurs under given social circumstances: relative advantage, compatibility, complexity, trialability, and observability. He finalized these constructs after many years examining such topics as agriculture and preventive medicine from his seat as a social researcher. Relative advantage examined the degree to which an innovation is perceived as better than the thing it is replacing. Implied subcategories of relative advantage included the potential for increased profit, improved social status, a decrease of personal discomfort, and other workplace incentives. Compatibility measured the degree to which an innovation “fits” in the current climate by considering the new system’s interoperability with legacy computer systems and workflow. Drastic switchovers to novel and incompatible software systems are frequently disruptive. When adopters have the option of using the innovation on a trial basis without large overhead investments of time or financial resources, this is an increase in the trialability of the innovation. Many potential users also like to see the innovation in use by their peers and understand its benefits before they choose to adopt. This quality is known as observability. The last of Roger’s five attributes was complexity, defined as, “the degree to which an innovation is perceived as relatively difficult to understand and use” [8].
2.2 The Call for Validity

Approximately twenty years later, the topic experienced a surge in interest, as a way to gauge the potential user adoption of information technologies. The social psychology [9-10], self-efficacy [11-12], cost-benefit [13], diffusion of innovation, channel disposition [14], and human-computer interface [15] disciplines all contributed to the body of knowledge, but with it came a new awareness of the need for accurate terminology and demonstrated validity. The multi-disciplinary approach had led to the creation of countless new attribute definitions and subcategories that frequently overlapped, but no clear ontology could be formulated. Davis [16] commented in the introduction of his work, “Although numerous individual, organizational, and technological variables have been investigated, research has been constrained by the shortage of high-quality measures for key determinants of user acceptance.” In an attempt to systematize the multitudinous studies, new findings were judged quantitatively using a set of coefficients and scores standard in social research work. The metrics, when applied retroactively to Rogers’ original theory of DOI, revealed many weaknesses and questioned its legitimacy in the technology and design settings [17-21].

Moore and Benbasat [21] were the third in a line of researchers to apply social research techniques to DOI (they were preceeded by Ostlund [17] and Bolton [18]). They sought not only to make revisions to validate the theory, but do so in a way that supported its application to information technology adoptions. They arrived at a total of seven distinct and valid constructs, which suggested splitting Rogers’ original observability construct into two constructs: result demonstrability and visibility, and adding a final construct called perceived voluntariness. They claimed that perceived voluntariness was not a black/white variable, but rather, one that exhibited a number of degrees, like the other constructs. The pervasiveness of perceived voluntariness on user behavior threw into question the generalizability of many of older studies conducted under purely voluntarily conditions.

Not surprising, Davis [16], who called for more stringent methods for innovation adoption studies, also published an article in the arena of information technology innovation that became a leader. From his investigation of the other disciplines, Davis determined that there were only two determinants of usage behavior that warranted further study. Perceived usefulness and perceived ease of use, he demonstrated, “were significantly correlated with self-reported indicants of system use” in the case of two computer applications, PROFS email and XEDIT file editor, though he could not ignore ease of use may in fact be a “causal antecedent to perceived usefulness” and not itself a separate construct. Davis focused on perceived characteristics because, as Moore and Benbasat [21] later elaborated, “many studies which have examined the primary characteristics of innovations have been inconsistent,” and what a potential adopters feels toward an innovation might differ from how they feel about performing the associated action. Davis’ work was highly respected for ten years, but recent studies [22-27] (e.g., post the year 2000) have noted that it is only applicable for voluntary usage simple computer applications like email and word processing and not applicable for multi-functional programs.

3. BEST PRACTICES FOR ADOPTION OF INFORMATION TECHNOLOGIES

Clearly, simple computer applications have been adopted into standard business practice, but the historical theories are experiencing a second periodic resurgence of interest, now applied to information systems for document management, and on the larger scale, knowledge management. Software developers are sitting on powerful new technologies for digital workflow integration, but Davis’ [16] comments twenty-five years ago still hold, that “invalidated measures are routinely used in practice… throughout the entire spectrum of design, selection, implementation, and evaluation activities.” The proliferation of articles pointing out the unforeseen challenges and costs of enterprise-wide software initiatives, including after installation activities like interface customization and technical support, testify to the misinformation of individuals charged with selecting the best software for the company [7, 28]. Too often, the software is incompatible with current practices at the company or too complex for employees to learn in the limited time allowed for training sessions. To exacerbate the problem, researchers and software developers pay little attention to interface design, focusing instead on increasing features.

Fortunately, the topic of identifying and exploiting predictors of user acceptance is gaining popularity. But, the current wave must bring its own set of insights to the table. While the early theories serve as a launching point, they are inadequate to address the entirety of complex information management systems which are frequently mandatory. In 1998, Rawstone, Jayasuriya, and Caputi [26] recognized that mandatory settings invalidated personal intent as a usage predicator and adopted the construct “symbolic adoption” instead. The term symbolic adoption (SA) was selected from the work of Klonglan and Coward [27] to express “one’s mental acceptance of an innovation, distinct from actual adoption which refers to actual use of technology” [22]. Nah, Tan, and Teh [22] employed this concept to investigate users’ perceptions of Enterprise Resource Planning (ERP) software in a mandatory, albeit non-profit, setting. ERP is frequently installed in the corporate setting to coordinate the access and transfer of information-rich digital documents. In the study performed by Nah et al. [22], perceived ease of use, compatibility, and attitude positively affected symbolic adoption, though perceived ease of use and perceived fit (“the degree to which the ERP software is perceived by a user to met his/her organization’s needs”) [22]) did not.
3.1 Technology Adoption in the Engineering Information Management Context

Clearly, more work is needed to build up a solid body of literature on adoption of complex software systems. Experts are agreed on how to perform multi-dimensional research in this area, but the current list of attributes that influence the adoption of information management software is small and limited and the number of vendors is ever increasing. New variables are also likely to emerge as software developers and human-factors experts together drill down into the pyramid that represents the activities necessary for achieving an information management infrastructure for product families. Furthermore, Enterprise Resource Planning is only the tip of the pyramid that extends down through engineering-specific enterprise-wide systems, product data management systems without geometry and functional support and those with, and the system for product family development [29]. Due attention must be paid to the unique characteristics of engineers and engineering companies that are not present in the human services or financial sectors of the American workforce. In due time, the product family infrastructure might also require sensitivity to workflow processes in other countries that partner with American companies to produce products.

In the future, research will seek out the unique set of attributes that are positively correlated to user adoption and continued satisfaction with information management systems for product families in profitable American companies that mandate usage. It is expected that this literature will equip researchers to confidently and consistently integrate human factors into their experimental projects and inspire software developers to do the same at the appropriate time. However, there can be no harm in experimenting with the inclusion of human factors while the product family infrastructure itself is still in infancy. Thus, it is reasonable to present here, even at an early stage, a number of best practices for increasing the user adoption of the proposed infrastructure from the bottom-up.

The responsibility for holding fast to these best practices must be shared between those designing the infrastructure and the software developers. As mentioned earlier, intuitive interfaces cannot simply be affixed around the system’s architecture once it is complete. Neither is it prudent for designers to intelligently “build in” human factors that software developers fail to recognize and unknowingly cloak with messy, unnecessary features. The designer is in effect the architect to the building that the engineers and construction personnel construct; cooperation from both specialties is required here.

Practice 1: Develop on Familiar Hardware and OS

Some implementation costs cannot be influenced by infrastructure designers, but a little forethought and carefulness can minimize costly problems that arise many years later. As mentioned, as a general rule, cost is reduced when an innovation is compatible with the prior behaviors. Potential adopters who are familiar with the hardware, operating system, and interface require less training and find that troubleshooting and customization can be handled without the assistance of pricey consultants.

Even though hardware and bandwidth prices continue to drop, labor and consulting will always be costly when changing to new hardware. Work completed so far on the product data management infrastructure has isolated OWL (Web Ontology Language) as an information technology platform upon which to build. OWL and other web based platforms that exploit P2P (peer-to-peer) architecture will operate on existing Internet-enabled hardware [30].

Likewise, it serves potential users for the new infrastructure to use a familiar operating system. A recent study by Peerstone Research [31] estimates that 65% of ERP users are currently operating with UNIX, 28% with Windows, and 2% with Linux. In the next two to three years, Unix could lose up to 15% of its share to Linux, with Windows holding steady. 78% of current UNIX users plan to stay with the operating system and 17% expect to transition to Linux. Before embarking on the design of the product family information management system, surveys like this should be conducted amongst potential customers. Measures should be taken to build on this operating system, even if a more robust system appears to better facilitate construction.

Practice 2: Think Like the Engineers

Keil, Beranek, and Konsynski [32] describe the case of a computer system called CONFIG (the actual name of software was disguised in the article for confidentiality) which was intended to ease the task of configuring hardware and software in a computer company. Two versions of CONFIG were issued; the second was intended to be easier to use, following user recommendations. Though the revised CONFIG system was heavily promoted, it failed to reach a desirable level of user satisfaction. In follow-up activities, opinions emerged from users that the developers did not have a clear idea of the sales process, among other reasons. The flow of information in the developers minds marched linearly from one step to the next, whereas the sales representatives’ model included a revision cycle and a small reordering of the steps.

Keil et al. [32] summarize, “... it appears that the developers harbored misconceptions about the sales process and therefore developed a tool that was not well-matched to the task (as envisioned by the users)... there are interdependencies across the steps and the flow can involve iterative sequences of activities”. The product family development process, like the hardware and software configuration and sales process described above, is particularly vulnerable to the misconceptions of developers who do not have hands-on experience with the mechanical design process. The infrastructure, therefore, which is being researched by practicing engineers and engineering professors, should have the design process firmly grounded in it to lower its
susceptibility. Just how the design process is structured, however, is an important role for the system’s designers.

The unique characteristics of mechanical designers and engineers play a second important role in the adoption of software. Social approval and image are primary drivers of innovation adoption in the United States. Most potential users question, at least subconsciously, “will adopting this innovation make me look good to my superiors and increase popularity with my peers?” The engineer in particular, is driven by competition. Any innovation that assists the engineer in designing a better product in less time will improve social image. In contrast, pledging commitment to an inefficient computer system will not, when surrounded by peers who are achieving better results using the old system [33]. Efficiency should be placed above more abstract qualities like aesthetic appeal and sensory engagement.

Engineers also thrive on feedback [34]. Good feedback encourages first time usage, including check-mark lists of automatic tasks performed, a listing of the time required to perform the tasks with graphical comparisons to how long this activity took a) before the implementation of the system and b) over time since the system was installed, results of error checks, number of files scanned, and progress indicators. Too much feedback can turn into an annoyance, however, so an enable/disable control should be included.

**Practice 3: Evaluate the Customer’s Prior Experience**

It is important to consider that, in the future, all companies that evaluate the benefits of adopting software for the management of product families may not have prior experience with product platforms strategies. It is taken for granted now that the partnership between companies and academic researchers exists because the companies are currently leveraging product platforms. As product platform strategies and information technologies win individual successes, however, they will point at each other. A corporation well versed in one will be unable to ignore the other as they appear side by side in trade articles and advertisements. Likewise, companies that use product platform strategies, and do so with specialized product family management software, will point their younger competitors directly to it, all leading to a condition where interested buyers have no experience but high ambitions.

Take word processing applications for example. As Murakami [35] describes, initially these systems were targeted at administrative assistants who had experience with manual typewriters. The first generation of word processing units sold with a monitor, keyboard, and small processor. These rudimentary word processors would not have sold well in schools because students did not know how to type; the help menus, which might have taught typing, were limited and difficult to understand. With the introduction of the second generation, administrative assistants used word processing applications on personal computers simultaneously with spreadsheets and databases. Only after the introduction of the personal computer into homes and schools for purposes like educational games and entertainment, did students adopt word processing as a standard way of preparing book reports and science projects. The number of classes and tutorials teaching typing blossomed.

The designers of an infrastructure for product family management, then, must grapple with the question whether the software is intended to assist designers who already use product platforms in their product lines or to educate and assist designers with no prior experience. The two objectives have a considerably different effect on what design for compatibility means. (Perhaps the infrastructure designers and developers, themselves, will consider a product platform approach and release several customized versions to interested buyers.)

The potential adopter who sees a computerized product family management system as a complement to his or her workflow approaches the innovation-decision with the assumption that the system is already somewhat compatible with business practices; the question there is whether or not the system supports and guides users through activities that are familiar and orderly. While designing for this is no easy task, (product platform strategies are guarded internally and there is no engineering textbook to date on the subject that explicitly states the order of steps), designing for an inexperienced product platform designer needs to include more “hand-holding.” One example of this is the incorporation of system reversibility which allows users to undo mistakes buried deep in the process and enables the system to propagate the correction to all the related parts. Though the second approach gives the designers of the infrastructure freedom to set up the process in any way, they are also responsible for teaching the users to think and design in this order. With the first approach, the responsibility to educate is somewhat alleviated, but the importance of matching the flow of information in product family design to the system increases. (For more details on information flows in product families see Fledderjohn and Shooter [36]).

**Practice 4: Develop for high visibility**

Information and opinions diffuse through both visual and verbal channels. Two visual techniques for encouraging the distribution of information are result demonstrability and process demonstrability, where the former educates potential users about the quality of results to expect and the later testifies to the length and complexity of the procedure [21]. Diffusion through word of mouth is known as communicability [20]. Observability, both visual and verbal, can be intentional, but a good product family management system will promote itself naturally and continuously, encouraging adoption among the late majority of potential users who are skeptical and interested in knowing more from their peers before proceeding.

The easiest way to demonstrate the benefits of the change to skeptical adopters is on a hard copy. Potential adopters who have not learned the software are excluded from viewing results that pop-up on screen and quickly disappear. Paper copies, on
the other hand, are viewable by anyone, regardless of whether or not the software is universally available, and can be easily photocopied and distributed. For this reason, the infrastructure should include multiple opportunities for easily printing graphics and reports. Several off-the-shelf solid modeling and finite element packages make printing cumbersome, requiring the user to resort to the “print screen” command and to cut-copy-paste into an appropriate graphics editor. The product family management system could be further improved by integrating instructive comments that could be selected to print automatically with the reports.

Late adopters would also like to see how successful their peers are operating the system outside of the contrived training sessions. A good infrastructure that accelerates user adoption makes the procedure easy to understand by a casual “over the shoulder” observer.

Lastly, the product family management system should call its modules and features by intuitive and descriptive terms, from which the potential user can easily infer the function or procedure. Acronyms and namesakes, such as “The Newton-Hertz theorem” or “The Bucknell diagram,” and otherwise wordy terms should be avoided during design so they are not inadvertently carried over into user interface. Setting the standard early makes simpler, user-friendly terms the norm instead of the exception!

Practice 5: Decrease the Perception of Complexity

Complexity is by far the most difficult of the attributes to isolate and quantify. It is, by definition, a subjective evaluation, where each user is entitled to his or her own scale and threshold. It is also tied very tightly to one’s perception of the innovation’s usefulness; a moderate amount of complexity can be tolerated if the innovation appears to be useful.

Keil, Beranek, and Konsynski [32] developed a model for the interaction between usefulness and ease of use in IT systems. On the model, IT systems that have both low usefulness and low ease of use are quickly rejected, and products that are easy to use, but do not increase productivity, are affectionately referred to as toys and games. Low ease of use and high usefulness systems are “Power User Tools.” Many IT systems currently fit this category. Ultimately, all systems would like to be in the fourth quadrant, reversed for tools that are both easy to use and useful. Keil et al consider many of today’s spreadsheet generation packages to be “Super Tools.”

Psychologist D.E. Berlyne [37] noted that one’s perception of complexity tends to increase with the number of distinguishable elements. If that is constant, then perceived complexity increases with the dissimilarity between the elements. Groups of elements that belong in pre-existing units (such as the alphabet), or that exhibit a common orientation, are seen as less complex than groups of elements that have no known pattern. Wherever possible highlight the similarity in dissimilar elements. A universal ontology or architecture is the single largest unifier of disparate product platform data elements. The fundamental structure will be reinforced each

<table>
<thead>
<tr>
<th>Usefulness</th>
<th>Ease of Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>I</td>
<td>Rejects</td>
</tr>
<tr>
<td>II</td>
<td>Toys</td>
</tr>
<tr>
<td>III</td>
<td>Power User Tools</td>
</tr>
<tr>
<td>IV</td>
<td>Super Tools</td>
</tr>
</tbody>
</table>

Figure 1 - Categories of Ease of Use vs. Usefulness [32]
users to see too many menus and buttons. Not only is it taxing learning all the functions for personal use, but one might feel obligated to use them immediately and skillfully in meetings and presentations. Therefore, it is advisable for the infrastructure to include master controls that gradually increase accessibility of features. System administrators could arrange for particular access times, during which new users could participate in projects together without risks or undue pressures from management.

4. CONCLUSION

Product platform strategies and information technology both have the potential to alleviate some of the problems faced by corporations today: shortened schedules, geographically dispersed teams, and greater demands for profitability. Product platforming capitalizes on the best part(s) of a design and reuses it in a family of derivative products -- products which are better able to meet diverse customer needs and desires while saving engineers time and money on redesign. There exist numerous commercial software packages for managing the vast repositories of documents which contain product data, but to date, no information management exists for managing product family information. The combination of information technology and product platform strategies will be very powerful.

However, like the introduction of any technological innovation, attention to the human-computer interface is critical in making the vision a reality. Potential users need encouragement to adopt the software and to continue using it, in order to fill the database with design information and train the system to make intelligent recommendations. Any system that is costly, complex, and incompatible with the current workplace procedures will suffer setbacks and risks being uninstalled completely. For the systems’ designer, such a result could be considered a failure and prompt the engineers to redesign.

Theories for evaluating and improving human-computer interfaces and for diffusing innovation originate primarily in the field of social science. In the last thirty years, social scientists have defined the attributes and user behaviors that increase the self-reported likelihood of innovation usage. More recently, specialists have applied those theories directly to technological innovations in the information management sector, like Enterprise Resource Planning software. Even ERP, however, is too detailed to manage the highly-abstracted relationships between members of a product family. It is necessary, then, to do more than sample interfaces from other sources.

Six practices were suggested for enhancing the human-computer interface of a product family interface while it is at the experimental level: development of the system on familiar hardware and operating systems, encouraging system designers to think like the engineers, evaluating the customer’s prior experience, striving for high visibility, decreasing the perception of complexity, and increasing the efficiency of the presentation mode. When these best practices are put into place early in the system design process, the interface is more likely to be consistent and appear seamless to users, thus increasing its overall usability.

5. FUTURE WORK

The past and current work presented in this article concern technologies that are foundational for the development of an information management infrastructure for product families; they do not actually address the proposed system. Therefore, new studies need to investigate the unique variable interactions that occur in situations where usage of an advanced, abstract, engineering-based information management system is mandated. Several iterations of each study will likely be necessary to arrive at those variables, just as was necessary to determine valid constructs for more simple computer applications.

Best practices in the separate but equally important domain of human-computer interaction (HCI) need also be evaluated for their applicability and prioritized. Admittedly, the present work has focused solely on applications of the Diffusion of Innovation theory in the technological domain (for brevity) and has omitted much of the field of HCI, which is far less structured and more dense. The amalgamation of DOI and HCI will yield a comprehensive list of specifications for the proposed product family information management system.

REFERENCES


