Explanatory Power of Intelligent Systems: A Research Framework

Robbie T. Nakatsu

College of Business Administration
Loyola Marymount University
Los Angeles, CA
Email: rnakatsu@lmu.edu

Abstract

This paper provides a framework for understanding the explanatory power of intelligent systems. It looks at content-based enhancements, drawn primarily from the Expert Systems literature, interface-based enhancements, and the appropriate selection of an advisory strategy. Such enhancements contribute to explanatory power by increasing system transparency and flexibility, and lead to outcomes such as better decision-making and problem-solving performance.

Keywords

Explanations, Decision Support Systems, Expert Systems, Intelligent Systems, User Interfaces, Artificial Intelligence, Human-Computer Interaction

1. INTRODUCTION

Intelligent systems provide advice to the end-user to assist in decision-making and problem-solving. (Caroll and McKendree, 1987). One criticism levelled against these systems is that they are rigid dialogues that are hard to understand (see e.g., Hayes-Roth and Jacobstein, 1994; Franklin, 1997). In Expert Systems, for example, it is hard to understand how a given set of inputs produced a recommendation by the system. To the end-user, the Expert System's reasoning process is a "black box". To address this problem, this paper argues that such systems require greater explanatory power in order for users to accept their recommendations and have more confidence in the decision support that they provide.

Explanatory power refers to the ability of an intelligent system to explain its actions (Nakatsu, 2001). Two related characteristics are relevant to understanding explanatory power: **transparency**, or the ability to see the underlying mechanism of the system so that it is not merely a black box; and **flexibility**, or the ability of the interface to adapt to a wide variety of end-user interactions, so it is not merely a rigid dialogue, but an open-ended interaction that allows the end-user to explore and understand the system more fully. While transparency of the system is a quality related to the informational content of the system itself, flexibility is more related to the nature of the end-user interaction with the system. The distinction is a subtle one, for it is easy to confuse the two qualities. More flexibility in the user interface can lead to more transparency—having a more open-ended interaction can enable an end-user to seek out more ways to better understand the system; by the same token, having a more restrictive interface can impair the end-user's ability to seek out more transparency in the system, even if it does already exist (Silver, 1991). Hence, this research views flexibility as a separate quality of a user interface that exists independently of interface transparency. The explanatory power construct, as defined in this paper, integrates and captures both transparency of the user interface and flexibility of the end-user interaction.

Making the distinction between informational qualities vs. interaction-related qualities of user interfaces is an important first step in developing a framework for explanatory power. Our goal is to describe and better understand the multi-dimensional nature of explanatory power, specifically, to investigate some of the determining factors of explanatory power. By doing so, we will be in a better position to offer design guidelines, which can be used by system designers to enhance a system's effectiveness.

The research framework looks at three types of enhancements to explanatory power. **Content-based enhancements** focus on augmenting the actual informational content of an intelligent system. We review the different types of explanations that are possible, drawing primarily from the Expert Systems literature. However, offering more content alone may not offer any gains in explanatory power, because the end-user may simply not bother to utilize the explanations. Hence, this research also looks at ways to enhance the interactive experience of the end user. Two additional determinants of explanatory power, related to enhancing the "interactive experience" of the end-user, are considered. **Interface-based enhancements** relate to the interface design choices that systems designers make to increase the effectiveness of intelligent systems. A number of possibilities are suggested that may increase the flexibility of the user interface, and as a result, its overall usability and usefulness. Still, the interface characteristics, in and of themselves, may not result in improved problem-solving performance. Hence, a third type of enhancement is the appropriate selection of an **advisory strategy**, or the manner in which the explanation is delivered to the end-user.

2. CONTENT-BASED ENHANCEMENTS

How can we improve the informational content of intelligent systems so that their internal mechanism is more visible for inspection (i.e., the transparency of the system is enhanced)? This section will answer this question by addressing content-based enhancements to explanatory power. The research involved in content-based enhancements has a well-documented history in the Artificial Intelligence (AI) and Expert Systems literatures, which this section will review. From this overview, a classification of explanation types by content is provided.

Computer-generated explanations have long been associated with expert systems use. Explanations use in expert systems begins with MYCIN, developed by Edward Shortcliffe at Stanford Medical School in the 1970's for the diagnosis and treatment of bacterial infections. MYCIN is considered to be one of the classic expert systems, certainly the most widely-cited, and it has introduced several features that have become standards in expert systems technology: rule-based knowledge representation, probabilistic rules to capture uncertainty, the backward chaining method, explanation, and a user-friendly interface (Turban, 1995). To aid with system debugging, Shortcliffe added a RULE command that, when requested, asked MYCIN which rule was currently being used. The rule was displayed in LISP, but later was displayed in English to make this simple explanation more user-friendly (Buchanan and Shortcliffe, 1985). Later, this RULE command was changed to WHY to enable "the user to examine the entire reasoning chain upward to the topmost goal by asking WHY several times in succession." (p. 333). The HOW explanation was also developed that enabled the user to descend the branches of the reasoning network. Today, WHY is commonly used by the user to ask why a certain type of input is needed by the system (typically the rule requiring the input is displayed). The typical HOW question is posed by users when they would like to know how a certain recommendation or conclusion was reached (typically the entire rule trace of the reasoning process is given).

Why does the user need an explanations facility? Buchanan and Shortcliffe (1985) offer several reasons. First, both the system builder and the end-user need to understand the knowledge base in order to maintain it and use it effectively. Second, systems builders can use explanations for debugging purposes. Third, explanations serve an educational function. Users who feel they learn something about the knowledge base are more likely to feel more comfortable with such a system. Fourth, explanations can help to convince users that the conclusions the expert system reaches are reasonable, and lead to their acceptance.

2.1 Limitations of Rule Traces

Explanations based on rule traces are clearly limited in terms of providing explanations that are instructive, inclusive, and easy to use. This section will report on the limitations of rule-trace explanations and suggestions for improving the capabilities of traditional explanation facilities.

The How-Why paradigm of explanations use described above offers one limited form of explanation. Other types of questions might need to be asked, especially if the end-user feels unsure of the system's advice. For example, "what-if" questions might enable users to explore the effect of changing assumptions in the rule-base, or to perform sensitivity analysis on input variables. "Why-not?" questions or "Why did you not conclude that <such and such> is true?" are frequently asked when probing real human experts (Ellis, 1989), yet this capability is a relatively difficult one to develop in an expert system.

Probably more problematic is that the "why" and "how" questions are based on rule traces, which an end-user is likely to have difficulty in comprehending. One solution is to replace rule-traces produced directly from computer code, with canned text that is easier for the end-user to understand. However, the replacement of computer-generated explanations with canned text explanations comes at a stiff price: user questions must be anticipated in advance and it is unlikely that all such questions will be thought of ahead of time (Moffit, 1994). Maintenance of such canned text explanations (keeping them in sync with an ever changing rule-base) could also create problems further down the line. Moreover, by using canned text explanations, the system has no conceptual model of what it is saying so that it is not possible to develop more advanced types of explanations, such as providing explanations at different levels of abstraction to the end-user (Swartout, 1983).

Another problem with rule traces is that they sometimes provide too much detail, which an end-user is simply not interested in seeing. As Swartout (1983) astutely observes in discussing the MYCIN rule traces: "Parts of the program [i.e., rule traces] appear mainly because we are implementing an algorithm on a computer. If these steps are described by physicians, they are likely to be uninteresting and potentially confusing." (p. 312). There is often too much algorithmic detail in a rule trace that an end-user cannot understand, or may not care to understand. An effective explanation must be pitched at a higher level, so that unnecessary details are left out.

Several researchers have also commented on the **opacity** of rules, or a rule's inability to make visible to the end-user the underlying reasoning process. Clancey (1983), for one, has noted that rules typically do not contain justifications or fail to shed light on underlying causal processes. This may be due, in part, to the way that expert knowledge is compiled: "rules are 'compiled' in the sense that they are optimizations that leave out unnecessary steps—evolved patterns of reasoning that cope with the demands of ordinary problems." (Clancey, 1983, p. 225) However, these intermediate steps frequently need to be explained to end-users who may not understand how an expert system reached a conclusion.

Clancey (1983) has also pointed out how strategic knowledge can be hidden in the premises of rules. He defines strategic knowledge as "an approach for solving a problem, a plan for ordering methods so that a goal is reached" (p. 233). For example, these rules, by the simple ordering of the premises, dictate to the inference engine which conditions should be checked before the others. Such strategic knowledge is effectively lost to the end-user requesting a rule trace.

2.2 More Sophisticated Explanations

The preceding discussion suggests a number of improvements to traditional rule traces, some of which have already been explored by a number of researchers. Obviously, an explanation facility cannot do everything, and part of the problem of good explanation design lies in understanding what types of explanatory capabilities can be feasibly developed, for a given task domain and for a given class of users. The following discussion considers some extensions to traditional rule traces.

In NEOMYCIN, an offspring of MYCIN, Clancey and Letsinger (1981) (see also Clancey, 1983; and Hasling et al., 1984), consider providing explanations that capture the overall approach used by the system to solve a problem—that is, the underlying strategic knowledge. One suggestion made to this effect is to capture strategic knowledge in meta-rules. These meta-rules provide the high-level knowledge for controlling the use of rules. An example of a meta-rule from NEOMYCIN is given below:

If (1) the infection is pelvic-abscess, and

(2) there are rules which mention in their premise enterobacteriaceae, and

(3) there are rules which mention in their premise gram-pos-rods,

THEN there is suggestive evidence (.4) that the former should be done before the latter

The firing of the above rule will cause one goal (2) to be pursued before another goal (3). Such meta-rules make the expert system's problem-solving strategy more explicit and therefore, potentially more visible to the end-user requesting an explanation.

In XPLAIN, Swartout (1983) suggests capturing strategic knowledge as a tree of goals, called a refinement structure. "Refining" a goal means turning it into more specific subgoals. Hence, the top of the tree is a very abstract, high-level goal, and the lower levels of the tree represent less abstract steps needed to implement this goal. Eventually, the level of the system primitives (i.e., built-in system operations) is reached.

Aikens (1983) identifies the problem of the user being unable to follow a line of reasoning under traditional explanation facilities. For example, MYCIN is unable to deal with more than one rule at a time, so that a line of reasoning is provided as a result of a user requesting a succession of "WHY" explanations. To make the line of reasoning more visible, Southwick (1988), for one, has advocated a hierarchy of landmarks or topics to guide the end-user. He defines a topic as "a logical and conceptual entity in the knowledge base of an expert system." Such topics serve as landmarks or anchoring points in the knowledge base. These topics, ideally, should have some intuitional appeal for an end-user so that a system can use them as convenient explanatory segments. Along a similar vein, Mockler (1989) utilizes dependency diagrams to graphically model a knowledge-based system. These diagrams show knowledge segments and their interrelationships, in a hierarchical fashion. They provide an overall summary view of the knowledge base. Such a tree structure could help end-users to better comprehend a knowledge base, unlike a flat set of rules.

Another criticism levelled against traditional explanation facilities is that they are incapable of justifying their actions, or providing an underlying reason for a system action. One solution to this lack of justification are model-based explanations that justify system actions and results by linking them to a deep causal model of the domain (Southwick, 1991). Such deep explanations are believed to give end-users an understanding of the underlying reasons for a recommendation. Swartout (1983) identifies two types of deep knowledge that might be provided as explanations: the **domain model** is descriptive knowledge about the domain and consists of such things as taxonomic knowledge and causal relationships; the **domain principles** are prescriptive knowledge and consist of such things as methods and heuristics used for problem solving.

Wallis and Shortliffe (1982) advocate the development of causal networks in order to create better explanations for medical consultation systems. The causal network, they contend, can serve as an integral part of the reasoning system, and can be used to guide the generation of customized deep explanations. For example, we might assume the causal chain in a system to be of the form t1 → t2 → t3 with each element assigned a measure of complexity. Suppose further that t2 is deemed to be too complex by a novice user. The system can tailor such explanations so that more fine-grained (and more complex) elements in the chain of causality are hidden from such users; in this case, t1 → t3 only is revealed to the user.

2.3 A Classification of Explanation Types: A Summary of Content-Based Enhancements

This section summarizes the three types of explanations that contribute to the overall explanatory power of an intelligent interface: rule traces, strategic knowledge, and deep justifications. This classification is similar to taxonomies developed by other researchers (see e.g., Gregor and Benbasat, 1999; Chandrasekeran, Tanner and Josephson, 1989; and Southwick, 1991). Under each type, points about how the explanatory power of the user interface may be increased are provided.

Rule Traces:

- Different types of questions are allowed such as "why not?" and "what if?".
- Rules are displayed in a natural language as opposed to computer-generated code.
- Rules traces are displayed at the right level of detail.

Strategic Knowledge:

- The problem-solving strategies are made explicit to the end-user (or are available upon request)
- The overall line-of-reasoning is made visible to the end-user.
- The strategic knowledge is appropriately structured for the end-user (e.g., a tree of goals, or a tree of topics).

Deep Justifications:

- An explanation is tied to an underlying domain model that provides structural knowledge (e.g., causal relationships about the domain) and taxonomic knowledge about the domain.
- An explanation is tied to underlying domain principles that provide knowledge about methods and heuristics used to problem-solve.

3. INTERFACE-BASED ENHANCEMENTS

Whereas content-based enhancements are concerned with enhancing the actual explanations themselves, interface-based enhancements focus on designing the user interface to foster transparency and flexibility in the system. The trend toward high-powered PC's and workstations during the 1980's and 1990's has given rise to the user interface taking on a more central role in the development of intelligent systems. In addition, the increased need to support the user's cognitive task so that the human user remains an active user, working in a cooperative manner with the system, has also necessitated that interface design considerations take on a more prominent role (Hayes-Roth and Jacobstein, 1994; Stelzner and Williams, 1988). Indeed, most of the current and most powerful Expert System shells on the marketplace contain powerful tools to design object-oriented, graphical user interfaces (e.g., Gensym's G2, Gensym Corporation, 1997).

Still another reason for the importance of the user interface is the increasing size and complexity of systems. Large-scale, industrial strength systems in organizations require that the user interface manage complexity well. This means that such interfaces must enable end-users to browse quickly through large amounts of information and to obtain multiple views of the same knowledge, in order to support the varying task requirements of the different users of the organization.

3.1 Characteristics of the User Interface

The careful selection of features of the user interface may also enhance the explanatory power of a system. The focus in this discussion will be on supporting the end-user—providing him or her with the capabilities to interact with the system in a variety of ways, as well as supporting the management of cognitive complexity. Stelzner and Williams (1988) identify five major requirements of the Expert System user interface, which will provide a framework for the discussion: 1) the natural idiom; 2) immediate feedback; 3) recoverability; 4) granularity; and 5) multiple interfaces to the same knowledge.

The natural idiom. This refers to the ability of the interface to represent the end-user's domain so that it maps as closely as possible to an end-user's mental model. Stelzner and Williams argue that the central metaphor of the interface should be that of the modelled world itself: instead of describing the domain (using a text-based conversational dialogue, for example), the end-user should perform actions directly on a graphical model of the domain. The end result is that the user interface is more natural and easier for an end-user to learn and use.

Immediate feedback. This refers to the ability of the interface to offer feedback to the user based on his or her actions. This quality supports the feeling of acting directly on the objects of the domain model, and removes the perception of the computer acting as an intermediary (Hutchins, Hollan, & Norman, 1985, as reported in Stelzner and Williams, 1988). Animation is one technique that might be used to endow the system with immediate feedback. Stelzner and Williams provide the example of a knowledge-based simulation of a factory, in which the graphical user interface contains a detailed layout of the factory to aid engineers in the determination of what operating strategy is best. Animation of basket movement through the factory allows the engineer to quickly identify bottlenecks and underutilized resources in the factory.

Recoverability. This refers to the ability of the interface to allow the end-user to back out of changes made to the system. End-users may wish to test out the effects of different changes on a system, and may desire an easy way to back out of these changes. Such a capability will encourage an end-user to explore and experiment with a system's capabilities. There are many possible ways of building recoverability into an interface.

An interface endowed with a high degree of recoverability may include undo facilities, history lists of the most recent actions performed, hypertext links that enable the end-user to jump back to the relevant portion of the problem-solving situation, and graphical user interfaces that enable end-users to select objects of the domain and easily modify their attribute values.

Granularity. This refers to the ability of the interface to allow the user to request different levels of detail, depending on the situation the user is currently in. Especially in complex systems composed of multiple components, this capability is crucial to supporting the cognitive limitations of an end-user. Such an end-user can become overwhelmed by the enormous amount of information required to understand the system. An interface that supports granularity may be one that enables the creation of multi-levelled, hierarchic descriptions of a domain. An end-user can drill down the branches of the hierarchy to obtain more detail, or go up the branches to obtain a higher-level view of the domain. An interface endowed with granularity would enable such a user to change the level of granularity frequently and with ease during a user session.

Multiple interfaces to the same knowledge. Given that different tasks and different users may have different requirements for utilizing knowledge, a system that enables the development of multiple interfaces to the same knowledge would permit greater flexibility. For example, in a real-time process control system, it might be critical for an operator to receive alerts whenever there is a malfunctioning in one of the components of the system. One interface might employ the use of multimedia, say the use of sound, to alert the user to a potentially dangerous situation. Another user of the system (say the supervisor) may not need these alerts, but would require hourly summaries of the outputs of the system, and the detection of underutilized resources in the system. A different interface and view of the system is obviously required.

4. ADVISORY STRATEGIES

The third type of enhancement to explanatory power are the strategies employed to deliver the advice to the end-user. Once explanatory content has been created, and the appropriate interface type and interface features selected, a designer of an intelligent interface may also utilize an appropriate advisory strategy to enhance the system's effectiveness. Selection of an inappropriate strategy may result in sub-optimal usage of the explanations that the system offers: end-users may not bother to request the explanations at all, or the explanations, when requested, may not be usable or useful at a given point during a user's interaction with the system. This section will consider different types of advisory strategies, and when and how they may be employed to enhance a system's effectiveness.

4.1 Types of Strategies

What are the different ways in which advice can be delivered to the end-user? A designer of an intelligent system may consider a variety of strategies that address the following questions: At what point during a consultation should advice be presented? Should advice be user-invoked or automatically provided by the system? How much advice should be given? Should the system allow for "special" modes of operation that protect the system from unintended consequences?

The **timing of the advice** is the first type of strategy that will be considered. Under this category of advisory strategies, one must consider timing issues of advice delivery. Dhaliwal and Benbasat (1996) distinguish between **feedforward** versus **feedback** advice, using the cognitive feedback paradigm (Todd and Hammond, 1956). Under this paradigm, there are three differences between feedforward and feedback: temporal order, cues focused upon, and case specificity. In terms of temporal order, feedforward is always presented prior to task completion, while feedback is presented after task completion. In terms of cues focused upon, feedforward focuses on input cues, whereas feedback is advice related to outcomes. Finally, in terms of case specificity, feedback is case-specific since it provides specific advice regarding the outcome (or recommendations) that the expert system makes, while feedforward tends to be more generic to the task at hand. Some researchers have chosen to think of feedforward as non-case-specific training provided prior to task performance. Case specificity is a content issue more than an issue of timing, but it is useful to note how the timing of the advice can (and should) affect its content.

The **provision mechanism** is another type of strategy that can affect explanations usage. Two types of provision mechanisms are considered. **User-invoked** explanations are explicitly requested by the user, whereas **automatic** explanations are automatically provided as determined by the system (Gregor and Benbasat, 1999). This distinction corresponds to the active vs. passive distinction, which Fischer, Lemke, and Schwab (1985) employ in their research on help systems: active help systems interrupt the user's actions, while passive help systems wait until the user explicitly requests advice.

Moffit (1989, 1994) conducted an experiment on the effectiveness of user-invoked vs. automatic explanations provision. She called the automatic explanations "embedded-text" explanations, since these explanations were embedded within the interface dialogue, and hence, the end-user would automatically see them. Her experimental evaluation sought to discover which explanations provision mechanism would enhance learning the most when using a production-oriented scheduling expert system. Subjects were randomly assigned to one of four treatments: (1) no explanation; (2) user-invoked, rule-trace facility; (3) user-invoked, canned text facility; and (4) embedded text (automatic provision).

Both declarative and procedural knowledge were tested and measured. The embedded text treatment appeared to offer the greatest advantage in terms of learning. This result led Moffit to conclude that the more difficult it was to access the explanations (i.e., user-invoked), the more the subjects perceived the expert system as a separate computerized tool—as opposed to a natural part of the human-computer interaction—and they, therefore, became less aware of its informational value.

Controlling the **amount of advice** is another way that a system designer may wish to affect explanations use. Having enhanced explanatory content is generally considered a good thing, and the more the better. However, there is a body of research that suggests just the opposite (see Carroll and McKendree, 1987, for a summary): having advice could actually distract a user whose goal is something other than learning. Proponents of user discovery of the system, for example, argue that advice should be provided only when the user explicitly requests it, countering the design recommendation that Moffit's study seems to suggest—that explanations should be embedded within the dialogue. As Carroll and McKendree (1987) observe, "the discovery approach takes advantage of opportunistic learning, that is, making the most of each unique personal experience" (p. 23). Many researchers seem to be in agreement with the discovery learning approach. Brown, Burton, and de Kleer (1982) argue that providing large amounts of advice was often quite deleterious, and that it is often better to leave the user alone, especially if the problem-solving task is small. They also suggest that no advice be provided at all if the user gets too far off track, the implication being that it is unclear what sort of advice can help a user in such a situation.

Finally, the use of **special modes** of system operation is considered another class of advisory strategies. Carroll and McKendree (1987) distinguish between two special modes of operation. **Control blocking** means that a portion, or subset, of the system's functions is made inaccessible to the user to prevent their accidental usage. One example of this type of strategy is to use a training wheels approach, in which a portion of the system is rendered inaccessible to novice users, who often access advanced functions by mistake, and then become distracted and confused by the consequences (Carroll and McKendree, 1987). Carroll and Carrithers (1984) showed that such an approach could lead to more efficient learning of a word-processing application. Another type of special mode is a **protected mode** in which a user action is protected from harmful consequences. One type of protected mode is a *reconnoiter mode* in which the actions of system commands are simulated without actually affecting the system's data (Jagodzinsky, 1983) A user of this system can switch to reconnoiter mode and try different things out, without fear of destroying or damaging system data, and then return to normal mode. Such a mode of system operation can encourage the user to more fully explore a system. Table 1 summarizes the advisory strategies that a system designer may wish to consider to improve the delivery of explanations.

4.2 A Summary of Advisory Strategies

This section summarizes the preceding discussion on advisory strategies. Selection of an appropriate advisory strategy can mean the difference between effective usage of explanations or ignoring the explanations altogether. This listing is by no means an exhaustive enumeration of the possible advisory strategies.

Timing of the advice:

- Appropriate use of feedforward and feedback should be made in the delivery of explanatory content.
- Feedback should be provided immediately after a system error has occurred (not delayed).

Provision mechanism:

- Automatic explanations should be provided if the system designer wishes the end-user to use them since the cognitive effort required to use them is low.
- User-invoked explanations should be provided if the system designer does not want to force explanations on the end-user; rather they are requested at the end-user's discretion

Amount of advice:

- Too little advice means the interface lacks explanatory content.
- Excessive amounts of advice may hamper discovery learning: too much advice may be distracting.

Special Modes:

- Control blocking may be used to limit an end-user's access to system functions that may be confusing and distracting to use.
- Protected modes may be implemented to promote user discovery and creative uses of information systems.

5. THE THREE FACES OF EXPLANATORY POWER: A RESEARCH FRAMEWORK

The emphasis of this paper has been on the three types of enhancements to explanatory power: content, characteristics of the user interface, and advisory strategies. Figure 2 depicts the framework, which includes both the enhancements and outcomes of explanatory power.

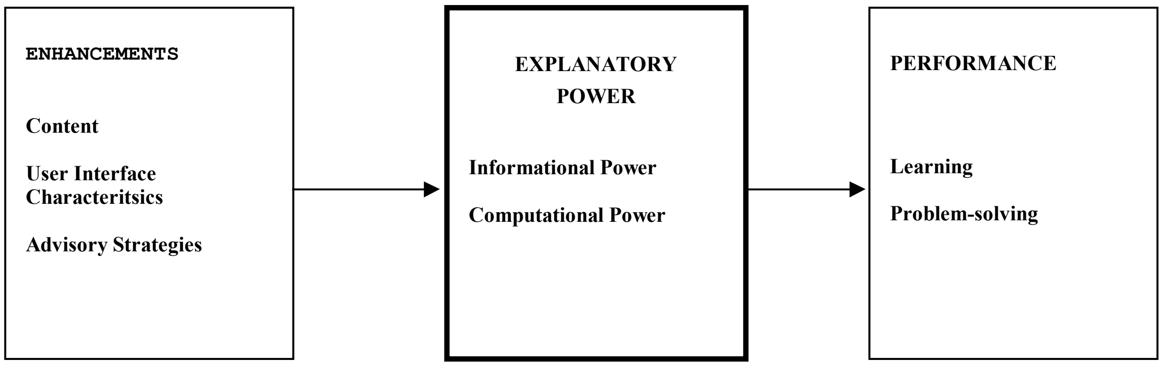

Figure 2: A Framework of the Explanatory Power of a User Interface

Two different aspects of explanatory power are included in Figure 2, informational power and computational power of a user interface. These concepts can be used to evaluate two different user interfaces. Larkin and Simon (1987), in their discussion on comparing diagrammatic representations, define two representations as **informationally equivalent** "if all the information in the one is also inferable from the other, and vice versa" (p. 70). Moreover, two representations are **computationally equivalent** "if they are informationally equivalent and, in addition, any inference that can be drawn easily and quickly from the information given explicitly in the one can also be drawn easily and quickly from the information given explicitly in the other, and vice versa" (p. 70).

From these definitions, we derive the notion of informational power, which is strictly related to content alone, while computational power evaluates efficiency and usability as well. Interfaces having greater informational power contain superior content, and interfaces having greater computational power have superior content that can be used and accessed more efficiently and effectively. Two user interfaces are said to be equivalent, in terms of explanatory power, if they are computationally equivalent (which, by Larkin and Simon's definition, means they are informationally equivalent as well).

Let us consider how each of the three determinants of explanatory power can affect informational power and computational power. Content-based enhancements, strictly speaking, can only increase informational power of the user interface. Interface-based enhancements and selection of an appropriate advisory strategy can also increase the computational power of a user interface. In summary, explanatory power of a user interface, then, is a more comprehensive construct than explanations content alone, because it also considers gains to be achieved in computational power as well.

Finally, it is worthwhile to point out that the three determinants of explanatory power—content, user interface, and advisory strategy—can often interact with one another in interesting ways to increase system performance. While it is useful to separate out the three determinants individually, in actual practice it is often difficult to speak of one determinant, in isolation, as creating more explanatory power. In fact, two determinants frequently work in tandem to create more explanatory power, so that it is often difficult to determine where one determinant ends and the other begins. For example, the support for graphical representations in the user interface may allow for the development of hierarchic strategic models that may be more powerful than the specification of strategic knowledge through a text-based interface. In this particular instance, both the interface, and the actual content of the system are enhanced. Similarly, the ability to inspect the components of a system may result in an interface that allows for more powerful deep justifications, which would not be possible in a more static user interface. Still another example would provide for appropriate feedback advice in a timely manner, in which content is dynamically generated and case-specific (interaction of advisory strategy and content).

Enhancing the explanatory power of intelligent systems can result in systems that are easier to use, and result in improvements in decision-making and problem-solving performance. Two experimental studies have been conducted that show promising results. In one, we developed hierarchic models of an Expert System knowledge base (an interface-based enhancement) intended to help a user to better understand the way that an Expert System reasons (Nakatsu and Benbasat, 2003). Users overwhelmingly preferred this type of interface, which gave them the ability to visualize how an Expert System reasons. In a second study (Nakatsu and Benbasat, forthcoming), we varied the advisory strategy of a decision support system that aided users in designing business logistics networks. Two special modes of operation were tested: a restrictive system that provided structured advice versus a non-restrictive system that was more open-ended and allowed users to request the problem-solving procedures in any order they wished. We found that the restrictive system was more effective for structured tasks (not surprising), but that the non-restrictive system helped users deal more effectively with novel problem-solving situations in which some type of system failure occurs (i.e., the system generated sub-optimal advice). Selection of an appropriate advisory strategy proved crucial. All in all, we think the investigation of explanatory power is a fruitful area of research, one that could have implications for the design of user interfaces.

REFERENCES

Aikens, J. (1983) Prototypical knowledge for expert systems. Artificial Intelligence, 20, 163-210.

Brown, J.S., Burton, R.R, and de Kleer, J. (1982) Pedagogical, natural language, and knowledge engineering techniques in SOPHIE I, II, and III, in Sleeman, D. and Brown, J.S. (Eds.), Intelligent Tutoring Systems (pp. 79-98), New York: Academic Press.

Buchanan, B.G. and Shortliffe, E.H. (1985) Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, Reading, MA: Addison-Wesley.

Carroll, J.M. and Carrithers, C. (1984) Training wheels in a user interface. Communications of the ACM, 27 (8), 800-806.

Carroll, J.M. and McKendree, J. (1987) Interface Design Issues for Advice-Giving Expert Systems. Communications of the ACM, 30 (1), 14-31.

Chandrasekeran, B., Tanner, M.C., and Josephson, J.R. (1988) Explanation: The Role of Control Strategies and Deep Models, in Hendler, J.A. (Ed.), Expert Systems: The User Interface (pp. 219-247).

Clancey, W.J. (1983) The Epistemology of a Rule-Based Expert System—a Framework for Explanation. Artificial Intelligence, 20, 215-251.

Clancey, W.J. and Letsinger, R. (1981) Reconfiguring a Rule-based Expert System for Application to Teaching, in Proceedings of the 7th International Joint Conference on Artificial Intelligence, 20, 215-251.

Dhaliwal, J.S. and Benbasat, I. (1996) The Use and Effects of Knowledge-based Systems Explanations: Theoretical Foundations and a Framework for Empirical Evaluation. Information Systems Research, 7 (3), 342-362.

Ellis, C. (1989) Explanation in intelligent systems, in Ellis, C. (Ed.), Expert Knowledge and Explanation: The Knowledge-Language Interface (pp. 103-126), New York: Halstead Press.

Fischer, G., Lemke, A., and Schwab, T. (1985) Knowledge-based Help Systems, in Proceedings of CHI'85 Human Factors in Computing Systems (pp. 161-167), San Francisco, CA.

Franklin, S. (1997) Artificial Minds, Cambridge, MA: MIT Press.

Gensym Corporation (1997) G2 Reference Manual, Cambridge, MA: Gensym Corporation.

Gregor, S. and Benbasat, I. (1999) Explanations From Intelligent Systems: Theoretical Foundations and Implications for Practice. MIS Quarterly, 23 (4), 497-530.

Hasling, D.W., Clancey, W.J., and Rennels, G. (1984) Strategic explanations for a diagnostic consultation system. International Journal of Man-Machine Studies, 20, 3-19.

Hayes-Roth, F. and Jacobstein, N. (1994) The State of Knowledge-Based Systems. Communications of the ACM, 37 (3), 27-39.

Larkin, J.H. and Simon, H.A. (1987) Why a Diagram is (Sometimes) Worth Ten Thousand Words. Cognitive Science, 8, 255-273.

Mockler, R.J. (1989) Knowledge-Based Systems for Management Decisions, Englewood Cliffs: NJ: Prentice-Hall.

Moffit, K. (1994) An Analysis of the Pedagogical Effects of Expert System Use in the Classroom. Decision Sciences, 25 (3), 445-460.

Moffit, K.E. (1989) An Empirical Test of Expert System Explanation Facility Effects on Incidental Learning and Decision Making, Unpublished Doctoral Dissertation, Arizona State University.

Nakatsu, R.T. (2001) Enhancing the Explanatory Power of Intelligent, Model-Based Interfaces, Unpublished Doctoral Dissertation, The University of British Columbia.

Nakatsu, R.T. and Benbasat, I. (2003) Improving the Explanatory Power of Knowledge-Based Systems: An Investigation of Content and Interface-Based Enhancements. IEEE Transactions on Systems, Man, and Cybernetics, Part A, 33 (3), 344-357.

Nakatsu, R.T. and Benbasat, I. (forthcoming) Designing Intelligent Systems to Handle System Failures: Enhancing Explanatory Power with Less Restrictive User Interfaces and Deep Explanations, under review in Decision Support Systems

Silver, M.S. (1991) Systems that support decision makers: Description and analysis, New York: Wiley.

Southwick, R.W. (1988) Topic Explanation in Expert Systems, in Kelly, B. and Rector, A. (Eds.) Research and Development in Expert Systems V, Proceedings of the Expert Systems '88 (pp. 47-57).

Southwick, R.W. (1991) Explaining reasoning: an overview of explanation in knowledge-based systems. Knowledge Engineering Review, 6 (1), 1-19.

Stelzner, M. and Williams, M.D. (1988) The Evolution of Interface Requirements for Expert Systems, in Hendler, J.A. (Ed.), Expert Systems: The User Interface (pp. 219-247).

Swartout, W.R. (1983) XPLAIN: A System for Creating and Explaining Expert Consulting Programs. Artificial Intelligence, 21, 285-325.

Todd, F.J. and Hammond, K.R. (1965) Differential Effects in Two Multiple-Cue Probability Learning Tasks. Behavioral Science, 10.

Turban, E. (1995) Decision Support and Expert Systems, Fourth Edition, Englewood Cliffs, NJ: Prentice Hall.

Wallis, J.W. and Shortliffe, E.H. (1985) Customizing Explanations Using Causal Knowledge, in Buchanan, B.G. and Shortliffe, E.H. (Eds.), Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project (pp. 371-388), Reading, MA: Addison-Wesley.

COPYRIGHT