THE INTELLIGENT MAN-MACHINE INTERFACE

William Bricken
May 1984

Traditional interface techniques provide a channel for symbol transfer between the user and the system, but fail to incorporate both behavioral knowledge of the user and structural knowledge of the software system. By actively negotiating the communication between the analyst and the system software, the Intelligent Man-Machine Interface (IMMI) serves as an information broker: a third party, so to speak, that is familiar both with the tasks, goals, and cognitive framework of the analyst and with the rules, constraints, scripts, techniques, and models of the Situation Examination, Hypothesis Projection, and Impact Assessment software of the system.

Mathematical formalism dominates the internal representation of the computing system. The predominant characteristic of human users is informality, in that humans usually construct partial and incomplete models of a domain, augmented by extensive knowledge in the form of experience. The primary function of the IMMI is to negotiate between these perspectives. The IMMI operates on the formal representations of the implementation as an agent of the user. It teaches the user to formalize his model as an agent of the implementation. And it strictly separates the two, to assure both the integrity of the formal analysis and the input of the intuition of the human analyst.

The purposes of the Man-Machine Interface (MMI) are:

1. To display system data and processes in a comprehensible manner,

2. To provide the analyst with fluid and comprehensible access to the data structures and processes of the system,

3. To facilitate ease of data entry, manipulation, and modification,

4. To partition irrelevant details of data processing from the complex of data that the analyst is specifically interested in.

To accomplish these purposes, we have developed a prototype MMI that negotiates between the interests and perspectives of the user and those of the mechanical system.

THE INTELLIGENT INTERFACE

We have developed a working prototype of the IMMI that can access the data structures and the functional components of a small domain. It knows how to display, construct, and control the software of the domain. It is instructable: the user can informally specify the addition of new data structures, new functions and even new rules for the manipulation of the knowledge base, and the interface will default, coerce or query the informal specification into the formal data structure needed by the software. It is flexible: the user can extend the concept hierarchy, construct new tools from existing models, and create new contexts. And it is empathetic: by tolerating ellipsis, ambiguity, and model adjustments, the interface buffers the user from the details of the representation.

The IMMI is composed of three complex sources of data, or knowledge: the domain implementation model, the user model and the self model. Input from the user is first parsed into the self model to establish whether or not the IMMI understands and is capable of responding to the communication. If the communication is intelligible, the IMMI attempts to match the content of the communication to existing data structures in the domain model. Successful matches are returned to the user, after being filtered through the user-model for appropriateness.

Features

We propose an intelligent interface with these properties, all derived from our experience with a prototype IMMI. These facilities are within the range of current technology, and do not incorporate fundamental research questions:

1. *Dialogue based interaction*

Data entry, modification, and query occurs via an exchange of information and requests for information, with the goal of establishing a mutual perspective.

2. *Maximal information transfer*

At all times, the system displays its interpretations, processes, and state. This includes presentation of abstract result schemata that indicate generally what the user should expect from a given interaction.

3. *Integration of incomplete information*

When the system has doubt about the meaning of a user input (e.g. an ambiguous phrase, ellipsis of qualifiers, or incorrect formatting), it either coerces or defaults the input into its formal structure, displays its action, and then requests approval or correction.

4. *Multiple representation techniques*

Although the system code is of a single form, this form can be mapped onto several different display representations at the level of the interface. For example, a behavioral path might de represented internally as a list of numbers; the interface could present this information as points on a map, as a path, as a sequence of latitude-longitude measurements, as a regression equation, or as bounded region containing the observed object with a parameterized degree of confidence.

5. *Multiple access techniques*

Data can be entered via the keyboard, mouse-clicking, a drawing-pad, menu-selection, or default-values.

6. *Dynamic display*

Machine processes, some as searching, are displayed while they occur. Suitable levels of abstraction of the process reduce information overload at the display.

7. *Multiple models and data abstraction*

The degree of detail in a description or an explanation is critical to its comprehension. The Intelligent Interface can provide Representation Filters at any level of abstraction supported by the internal data structures. Thus, the user may analyze behavioral intent by examining the raw data from sensors, by abstracting this data into a series of probable locations over time, by abstracting locations into a path, or by abstracting the path into a behavior pattern.

8. *User-modeling*

A parameterized template of the intentions of the user can drive the systems default interpretation actions. For example, if it is known that the user desires a high degree of confidence in a particular conclusion, the system can automatically select conservative statistical procedures. Another example is empathetic searching, in which the system has a generic outline of the types of result that the analyst desires, and constrains its reports to these limits.

9. *Reflection by the system*

The dual of a user-model is a system-model. The Intelligent Interface can maintain data structures that model the architecture, processes, and state of the system. As a representative of the user, the interface can query and display system structure and knowledge, providing the user with a clearer model of the processes that generate system output.

10. *Explanation*

The purpose of a negotiated representation is to assure a common understanding of explanations. The system would develop a map between its internal processes and the intents of the analyst. For example, if the analyst intent is to locate a particular submarine with a particular degree of confidence, the explanation that the system provides for its location might include only the confidence bounds of the track data structure. If the analyst wishes to apply a particular model of the submarine's behavior, the explanation must also include how the confidence bounds are determined.

11. *Primitive commands*

Basic interaction commands for the IMMI might include the following:

1) FIND: general database query
2) REMIND: access to interaction history
3) SHOW: general display manager
4) MAKE: general constructor for known objects
5) LEARN: object and process creation of:
    a) instances, by direct construction (i.e. MAKE), comparison to other instances (templating), and generalization from an instance
    b) categories, by refinement of parent category, generalization of child category, and generalization of an instance
    c) relations, by definition and by analogy to existing relations
    d) commands, by plan specification (definition) and modification
    e) rules, by direct entry and by generalization from observation
6) FORGET: object and process destruction
7) Primitive programming commands such as ASSIGN, STORE, LOOP.
8) Logical connectives.

These facilities are all within the range of current technology, and do not incorporate fundamental research questions.