Tech Info Library

AI: Some Working Definitions Of The Terminology (Part 1 of 2)

Revised: 6/17/92
Security: Everyone

AI: Some Working Definitions Of The Terminology (Part 1 of 2)

==========

Article Created: 19 August 1988
Article Last Reviewed: 9 June 1992
Article Last Updated:

As with any other computer field, "AI" (Artificial Intelligence) has a distinguishing set of buzzwords. Following are a few working definitions that either are used in this paper, or that one can expect to hear when talking with a typical "AI" professionals.

Artificial Intelligence, or AI:

Trying to define AI is similar to trying to define exactly what HyperCard is to someone who has never seen it (or worse, never worked with a Macintosh). The field of AI includes many areas, such as speech, vision, robotics, natural language interfaces, neural networks, machine learning, automated programming, intelligent agents and expert systems. Typically, when someone thinks of AI, they are associating a computer/robot with a more specifically human activity (thinking, seeing, hearing, understanding, moving).

Common LISP:

With the various versions of LISP available, Common LISP is recognized as the emerging industry standard for LISP today. This should help lead to a portability and consistency among architectures, allowing a complex problem to be developed with a LISP machine, while being delivered on a less expensive, possibly non-LISP architecture.

Conventional vs. Symbolic Processing:

Conventional software has typically been used for problems that are more algorithmic or numeric in nature and well understood, with the computer use being to perform very rapid calculations or to process very large amounts of

data in an orderly fashion.

Symbolic processing deals with information that is more conceptual ("symbolic") in nature, such as: "Given a certain CAD/CAM/CAE design criteria, what is the ideal design (lowest cost, most accurate, simplest to manufacture, most reliable)." While people will typically argue that you can do anything with any language (given enough memory, disk space, programmer time...), the main advantage of symbolic processing is to free the programmer/developer from dealing with the lower levels of implementation, allowing him to focus on a higher, conceptual level of abstraction to address his real problem at hand.

Another way to think of symbolic processing is the concept of symbols. Typically a symbol represents more than just physical definition. If someone asks for a "pen", they are probably not as interested in the physical features (gold cross, BIC, apple logo or size = 5.75 inches) but rather are asking for something that will write in ink.

Data Type Processing:

LISP programs typically use a "tag" implementation that allows data types to be checked at run-time, as opposed to mandating that data types be declared in advance. This gives applications more flexibility to deal with complex and changing situations over the development life cycle.

Development vs. Delivery machines:

This distinction indicates the difference between the computer system on which an application is developed , and the computer on which the final application is delivered to the end user.

An example: a company might use a LISP machine and AI software package (which may total well over $100,000 for initial hardware/software costs) to develop a solution, but deliver it on a microExplorer or Mac II for a much lower cost per end-user. The idea is to have the ideal development environment to save development cost and increase programmer productivity, while also being able to deliver the solution in a relatively inexpensive environment.

Domain Expert:

One typically recognized as an expert in a field (domain).

This article is continued in "AI: Some Working Definitions Of The Terminology (Part 2 of 2)".

Tech Info Library Article Number:3204