Knowledge Servers

Andrew Basden
University of Salford, UK

INTRODUCTION

Knowledge servers aim to provide knowledge rather than mere information. While information may be "delivered" to the user, such as via Web pages, knowledge is generated within the user by the user's own thinking processes (Newell, 1982), often stimulated by information received and the user's situation. Therefore, a knowledge server, though it works by delivering information, is differentiated from an information server by (a) being tailored to the user at the time they connect to the server, (b) taking account of context, (c) making inferences from information provided by the user, rather than merely retrieving data or pages, and (d) engaging in a process of stimulating the user, for example, by letting the user explore different possibilities and obtain explanations. For example, an information server might list some factors that cause stress-corrosion-cracking in steel, while a knowledge server would dialogue with the user to assess the risk in their specific situation and enrich their understanding. While the entire Internet might fulfil such a function, it is possible to create resources specifically designed as knowledge servers.

BACKGROUND

The idea of knowledge servers arose out of work on knowledge-based systems (KBSs). These were originally standalone programs that had a representation of human expertise, encapsulated within them, with which they provided expert advice to their users. During the 1990s it became clear that there were advantages in linking them to the Internet so they could be a portalled resource on the World Wide Web.

Knowledge-Based Systems

The original powerful notion of KBS was that the general mechanisms of inference can be separated from the domain knowledge (e.g., stress-corrosion-cracking), as an inference engine operating on a knowledge base (KB). The KB is the represented concepts, relationships, rules, and calculations relevant to the domain of knowledge (e.g., acid can initiate pitting, which starts a crack) and the engine searches these to determine what information it needs in order to make useful inferences. It usually obtains this by putting questions to the user.

The user experiences a session with the KBS as a sequence of questions to which they supply answers. What question to put each time is decided by the inference engine according to the relationships in the KB and the information received so far (e.g., if there are no acid questions related to pitting might be irrelevant). Thus, which questions are asked cannot be predicted in advance. As each answer is received, the state of the KB is gradually updated, until it has sufficient information to reach a conclusion, which is declared to the user. Since all this is guided by relationships meaningful in the domain of knowledge, it is "intelligent," and the user can enhance their understanding by exploring them. Moreover, KBS inference can involve not just logic, but also probabilistic, fuzzy, or Bayesian reasoning.

All this makes KBS particularly useful in domains of specialist expertise, ill-structured knowledge, and dynamically complex calculations. The quality criteria by which a KBS may be judged include:

- Accuracy of the knowledge (the concepts, and the type and strength of relationships);
- Completeness of knowledge (including all those "outer" (Jacob & Ebrahimpur, 2001) that are often overlooked);
- Trustability—so that it will not mislead the user who operates in a context not envisaged by the KB designers: though rare, these should be elicited and explicitly represented in the KB (e.g., some rare kinds of pitting can occur without acid);
- The way it treats uncertain input;
- Meaningfulness of the questions put to the users, so that what the users (think they) understand by the question is what the KB designer intended them to mean;
- Helpfulness and insightfulness of the explanations given for questions; and
- Transparency of the KB to the user.

To achieve these, KBS development involves sophisticated knowledge elicitation and representation techniques, with considerable testing at the knowledge level. Liebowitz and De Salvo (1989) discuss such issues.

KBS and Internet

KBS and the Internet (especially World Wide Web) can overcome limitations in each other. To static information, KBS can add dynamicity. To the delivery of information, KBS can add sophisticated inferencing (including with uncertainty) and explanation, so that knowledge results. Moreover, whereas most WWW information is placed by individuals and might be not be reliable, a good KBS is a result of a social process of elicitation, and achieves a tested degree of accuracy, completeness, and trustability. It is expert, generally applicable, and yet able to handle exceptions.

On the other hand, linking KBS to Internet can overcome two types of knowledge isolation. Geographical isolation-only those at the machine on which the KBS resides have access to it-is obviously overcome. Epistemological isolation arises from inability to link the questions or results to other available information that might aid their interpretation, and is especially important in ill-structured domains like strategic planning. This isolation can be ameliorated if the KBS presents its questions and results online as dynamically created Web pages that contain links to other relevant information, explanations, or annotations, or even a facility to e-mail questions or comments that arise during the session (Sehmi & Kroening, 1996).

INTERNET-ENABLED KBS DEVELOPMENT TOOLS

At the turn of the millennium, various types of tools were available for constructing KBS that can be linked to the Internet. Most operate HTTP (hypertext transfer protocol). Some connect existing KBS toolkits to the Internet, such as AGENT_CLIPS (Cengeloglu, 1999), CKNP (Maluf, 1999), JESS (Jess, 1999), a multithreaded version of Cyc (Guha & Lenat, 1994), and ART*Enterprise/Web (Art, 1999).

Others have been built from scratch, including WebLS (Sehmi & Kroening, 1996), LogicWeb (Loke & Davison, 1996), PiLLoW (Cabeza, Hermenegildo, & Varmaa, 1996), and Istar (Basden, 2000).

Knowledge servers must be clearly differentiated from other types of KBS-Internet linkages, especially:

- Intelligent agents that perform complex, "intelligent" actions on the Internet without human involvement on behalf of other resources with which they interact directly. Agents can extend the power of Web servers (Boley, 1996) or news servers (Cengeloglu, 1999); CKNP is designed for this.
- Conventional, stand-alone KBS that obtains some of its input by acting as client to servers (e.g., Web pages or news servers). AGENT_CLIPS and PiLLoW provide this kind of functionality.
- Active Web pages,that contain programmable code and state, which is activated on the client machine. LogicWeb is designed for this. These are only suitable for tiny KBs.

Knowledge servers, by contrast to these, are designed to be accessed by human users rather than other agents, act as servers rather than clients (though they might well obtain information from Internet sources), usually have extensive KBs, and are designed to stimulate human knowledge. It was this duty for which Istar was particularly designed, but JESS, ART, Cyc, and WebLS are also usable for this.

THE CHALLENGES OF KNOWLEDGE SERVERS

Knowledge servers present three types of challenge to their designers: technical, knowledge-level, and cultural, of which a longer discussion may be found in Basden (2000).

Technical Challenges

In addition to the obvious, common technical challenges like run-time efficiency and security, the technology of knowledge servers imposes other challenges. Most arise from the fact that the KBS operates a session with the user.

During such a session, the natural operation of a KBS inference engine is directly opposed to that of the Internet: the roles of client and server are reversed. Under the usual client-server model (CSM), the user (client) makes a request and the server responds by sending back information (such as a Web page), and each such request-response pair is seen as essentially independent of all other pairs. But a knowledge server that sends questions to users and expects answers in return sees itself as "client," and the user as "server." Each question, or "request" to the user for information, is in fact a CSM response, and each answer, or user's "response," is in fact a CSM request. This means that knowledge servers work "against the grain" of the client-server model, on which most of the operation of the Internet depends, and this gives rise to a number of problems. In particular, a mechanism must be designed into the inference engine for pairing each CSM request with the previous CSM response.

The session is a sequence of such question-answer pairs. Since the choice of question to be sent depends dynamically on all the answers thus far received, a persistent memory must be kept of the state of the KB throughout the session. This memory may be either client-side or server-side. Client-side memory requires that the entire memory be transferred with