RC 20835 (92272) 08 May 97
Computer Science

IBM Research Report

Building Commercial Agents: An IBM Research Perspective (Invited Talk)

Benjamin N. Grosof

IBM Research Division
T.J. Watson Research Center
P.O. Box 704, Yorktown Heights, NY 10598
(914) 784-7783 direct -7455 fax -7100 main
Internet: grosof@watson.ibm.com (or grosof@cs.stanford.edu)
World Wide Web: http://www.research.ibm.com/people/g/grosof/home.html

Talk Abstract

This talk presents our experience at IBM in building commercial agents, including philosophy, market opportunities, tools, applications, lessons, and future research directions. IBM has released reusable tools for building agents, including Agent Building Environment based on our group's research. IBM has also built practical intelligent agents applications in e-commerce shopping, customer service, mail, news, and Web information, based in part on our group's research. Agent Building Environment "situates" reasoning by augmenting it with clean and dynamic procedural attachments for perception and action. It is especially useful to enhance "information-flow" applications with agents that not only find and filter information items, but also categorize, save, and selectively disseminate them. Some of our latest research responding to practical building lessons includes: conflict handling; learning from observation of users; and learning from inter-agent knowledge-level communication.

The following paper body corresponds mainly to the portion of the talk that focuses on our group's research. It was created by revising and integrating material available on our Web pages.

Publication Information
Appears in: Proceedings of the Second International Conference and Exhibition on Practical Applications of Intelligent Agents and Multi-Agent Technology (PAAM97), held London, UK, April 21–23, 1997, edited by Barry Crabtree of British Telecomm Laboratories, Martlesham, UK. Practical Application Company Ltd., http://www.demon.co.uk./ar/PAAM97/, P.O. Box 137, Blackpool, Lancashire, FY2 9UN, UK.
Copyright has been retained by the author.

Keywords: intelligent agents, Internet, electronic commerce, applications, developer's toolkit, information flow, search, retrieval, filtering, dissemination, categorization, prioritization, workflow, news, e-mail, customer service, logic programming, procedural attachments, dynamic registration, object-oriented, rules, user authoring, situated reasoning, C++, Java.

For More Info, including Talk Slides and Follow-on Papers:
See on the Web: the author's personal page (above), or the project page http://www.research.ibm.com/iagents/. These include links to the IBM CyberJournal Research Reports server http://www.research.ibm.com:80, and to the main IBM intelligent agents page http://www.raleigh.ibm.com/iag/iaghome.html which includes additional information about Agent Building Environment and its applications. The personal and project pages above also link to our (project) publications page which groups and abstracts selected publications, presentations, and white papers.

1 Embeddable Intelligent Agents for Information Flow

Our group's mission is to develop intelligent agent technology that is highly reusable and easy to integrate with a broad spectrum of networked applications. Towards this end, we prototype applications in tandem with developing reusable componentry. We also contribute to company-wide efforts in strategy and in common architecture, e.g., for inter-agent knowledge-level communication and inter-operability. For more information about company-wide efforts, you can see the main IBM intelligent agents page (http://www.raleigh.ibm.com/iag/iaghome.html).

Our reusable intelligent agents technology is embodied as an extensible C++ class library, called *RAISE*. RAISE is mnemonic for: Reusable Agent Intelligence Software Environment. The first phase of RAISE's capabilities include: rule-based inferencing, user authoring of rule bases, integration with external software components, and basic support of inter-agent knowledge-level communication. RAISE is deeply object-oriented in design. RAISE also features dynamic pluggability of user-authored rule sets (including easy merging and updating), and development-time pluggability of reasoning engines.

RAISE is at the core of the *Agent Building Environment* (ABE) developer's toolkit product alpha (http://www.raleigh.ibm.com/iag/iagsoft.htm) which has been released by the IBM Intelligent Agents Development team at Endicott, New York (in collaboration with our research group). ABE is available free for Internet download.

There are several aspects to our work on RAISE/ABE, which can be described via an anthropomorphic analogy.

- "Brain": reasoning and learning. To begin with, this includes inferencing with yes-no rules, in a primarily forward direction, e.g., driven by a set of facts that represent the *event* of an information item's arrival. (By "yes-no", we mean beliefs that are true or are false, as opposed to having a degree of truth in the manner of probabilities or fuzzy logic.) For brevity's sake, below when we speak of a rule set we will mean a set that contains rules and/or facts.

- "Body": situating the "brain" in a larger environment.

  Historically, reasoning systems have been difficult to marry with other software technologies. RAISE/ABE has an especially *innovative* technique for *situating* reasoning engines ("brains") by connecting them to sensors ("eyes"/"ears") and effectors ("hands"). Sensors and effectors are procedural attachments. Linkages to sensors and effectors are treated as a syntactic and semantic extension of the pure-belief knowledge representation.

  Attached procedures (external to the agent's core reasoning engine) are structured by packaging them into dynamically registered adapters. Each adapter corresponds to an application or a useful software component. Several adapters, including for stock quotes, net news (NNTP), and the World Wide Web (HTTP), are included in the currently available version of ABE. ABE supports and includes adapters written in Java, as well as in C++.

- “Society”: inter-agent communication, especially at the knowledge level. To begin with, this includes support for exchanging rules and facts in a standard fashion, using the approach of the ARPA Knowledge Sharing Effort (http://www.cs.umbc.edu/kse/); e.g., ABE currently supports Knowledge Interchange Format. Sensing, effecting, and delivering trigger events (see below) also provide modes of inter-agent knowledge-level communication.

- Human-agent interaction: both the deep and the surface aspects of user interface (UI). To begin with, this includes the user authoring of rule sets. Employing rules (with clean semantics) as a method for a user to instruct an agent helps ease the user's job in modifying the instructions, and in understanding what the agent does. (Rules are also simpler to use than scripting-type programming languages, and are more powerful than menus.) This is an example of deep UI. Employing graphical template forms helps structure the authoring of rules that are particular to a specific application; this is an example of surface UI.

  ABE includes a rule editor and a persistent rules library, as well as facilities for changing rule sets at run-time (i.e., dynamic management). This enables users to author their own rule sets. This authoring is further supported by two other capabilities of ABE. ABE detects syntactic and semantic problems in rule sets at the time of rule set editing, rather than just at the time of the agent “running” the rule set. ABE logs the agent's activities and can present (“explain”) that to the user.

  RAISE/ABE's knowledge representation is highly declarative (as opposed to procedural) in its semantics. Its rule sets are thus highly reusable and portable. ABE supports Knowledge Interchange Format (KIF), an emerging standard in the area of inter-agent knowledge-level communication.

RAISE/ABE is especially appropriate for *enhancing network-centric/Internet applications with embedded intelligent agents* that perform *information flow* functions: finding, searching, filtering, categorizing, storing, routing, and/or selectively disseminating information items. Pilot *prototype applications* for RAISE/ABE, currently running, include: *electronic commerce* shopping and *customer service* support workflow (e.g., customer questions and staffer answers), on the Web and in Lotus Notes; and *news* and *e-mail* on Internet.

2 More Details: Situated Reasoning in RAISE/ABE

RAISE/ABE has an innovative approach to embeddability. The reasoning engine is “situated”, in the sense of being attached in a rich and tight way with attached procedures outside of itself. This addresses the major challenge of knowledge-based technology: to integrate closely, smoothly and productively into mainstream software technology.

Attached procedures are important for several roles:

- “sensing” to test a rule antecedent condition by querying another software component (perhaps itself an intelligent agent);

- "effecting", upon drawing a conclusion, to execute a corresponding action by invoking another software component (perhaps itself an intelligent agent); and
- "triggering" the overall engine into an episode of activity, e.g., by delivering an event that represents the arrival of an information item via a set of facts that describe that item.

RAISE/ABE's innovative approach to procedural attachments includes:

- run-time association of attached procedures to knowledge representation expressions (e.g., predicates);
- clean semantics for the above kinds of procedural attachments; and
- packaging of attached procedures into "adapters".

Procedural attachments are tightly, yet flexibly, integrated with the knowledge representation and the reasoning. The mapping of an attached sensor or effector procedure to a knowledge representation expression (e.g., predicate) is specified via a special kind of rule, as part of a rule set. This mapping can thus be specified at run-time, rather than being fixed statically in the manner of built-ins. Also at run-time, the attached procedures may be registered to make their availability and interface known to the engine.

Clean semantics for sensing and effecting is based on a control invariance property similar to that for pure-logical classical inferencing (soundness, completeness, independence with respect to sequencing of inferencing steps). The underlying inferencing algorithm and knowledge representation in the *current* ABE version has the following characteristics:

- Extended-form rules are permitted: these may contain AND and OR logical connectives in the antecedent expression, and may contain AND logical connectives in the consequent expression. Within the reasoning engine, these extended-form rules are transformed into Horn-form rules. (Horn form permits AND but not OR in the antecedent, and permits no logical connectives in the consequent).

  We have found that this extended form is a significant convenience to users during their authoring, enabling them to specify rules more concisely and naturallly.

- Aside from the procedural attachments, the reasoning essentially can be viewed as a forward-direction logic program. (The sensing procedural attachments enable backward-direction reasoning aspects to be mixed in to the primarily forward-direction reasoning, however.) This logic program (i.e., rule set) is currently restricted to be pure, monotonic, acyclic and Datalog, in the logic programming community's terminology. That is, the set of rules contains no use of cut, contains no use of negation-as-failure, is free of recursion (dependency cycles among predicates), and contains no (non-0-ary) logical-function symbols. We have found that these restrictions are expressively adequate for many applications, help by guaranteeing that reasoning can be computed efficiently, and help keep users out of trouble during their authoring of rule sets. Note that logical functions can be represented essentially equivalently via predicates, so they are not essential expressively.

- Efficiency in inferencing is improved by (pre-)computing a control *agenda* for each (persistent) rule set. The agenda is essentially a *stratified* (in the sense of logic programming terminology) sequencing of the rules, based on an analysis of the dependencies between the predicates that appear in the rules (and facts). Given the agenda, each rule only has to be attempted once during inferencing, as opposed to repeatedly as in many inferencing algorithms. This enables reasoning to be performed fast. Indeed, reasoning's computational complexity thereby *scales linearly* in the number of rules. Essentially, this agenda-based approach is an adaptation of [Dowling and Gallier, 1984]'s linear-time algorithm for propositional Horn inferencing.

- The situated inferencing is exhaustive and complete as well as sound. That is, all forward-direction conclusions are indeed generated, all relevant sense-able knowledge is indeed obtained along the way to help draw those conclusions, and all effector calls justified by the conclusions are indeed invoked.

Overall, we can summarize the knowledge representation and reasoning in the current ABE as follows. A reasoning episode in ABE is the application of a user's selected (persistent) rule set to a fact set corresponding to a trigger event that represents arrival of an information item. In this episode, ABE's reasoning engine performs exhaustive forward-direction situated inferencing over a monotonic acylic Datalog extended-form situated logic program. This situated inferencing includes execution of (external-to-the-engine) attached procedures for the sensing and effecting (as well as for the delivery of the triggering event). The logic program is the union of the user's selected (persistent) rule set and the trigger-event fact set. One can view the former as previously asserted premises and the latter as newly asserted premises.

The ABE approach to situating the reasoning is patent-pending.

The theory of RAISE/ABE's situated reasoning, including the situated control invariant, and of its agenda-based inferencing approach, will be described much more in forthcoming papers.

3 Current Research; Information Economies

A number of enhancements to RAISE itself are currently under development. We have designed (and currently have running as a prototype) an especially *innovative* technique for handling *conflicts between rules*. Rules may override each other, based on specified override orderings. The approach, called *courteous logic programs*, is computationally low-overhead, guarantees a consistent set of conclusions, and is semantically clean. It facilitates merging and updating rule sets, as well as agents learning from data mining (statistical induction) or from inter-agent knowledge-level communication (taking "advice" from "friends"). (For cognoscenti: it constitutes a practical form of "prioritized default" reasoning, a kind of "non-monotonic" reasoning.) For more details, see our Web publications page (URL given above).

Our further technical directions include: extension of reasoning capabilities to include machine *learning* and *probabilistic* / fuzzy reasoning; closer integration with *text analysis and* *search*; closer integration with *data mining*; and inter-agent knowledge-level communication, especially to exchange rules and facts, e.g., about electronic commerce. We have designed an architecture for *itinerant* (mobile-execution) intelligent agents that interact at *Agent Meeting Points*.

Future versions of RAISE/ABE may include additional kinds of embeddability, as well as of reasoning.

Finally, we are working on *Economies composed of intelligent agents*, including micro-economic interactions (and decision-making) and macro-economic emergent phenomena, especially for information goods and services. We call these *Information Economies*.

Biography of the Author

Dr. Benjamin Grosof is a senior researcher at IBM Watson Research and a pioneer of practical intelligent agents. He has led research design and productization of Agent Building Environment, as well as piloting of its applications. Benjamin's areas of expertise also include e-commerce and inter-agent communication, as well as fundamentals of reasoning and learning.

Acknowledgements

Thanks to the other members of the RAISE and ABE teams at IBM, especially David W. Levine, Hoi Y. Chan, and Jeffrey Kephart at Watson.

References

[Dowling and Gallier, 1984] W.F. Dowling and J.H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *Journal of Logic Programming*, 3:267–284, 1984.