M: An Architecture of **Integrated Agents**

The study of software agents has resulted in a diverse set of views and realizations. One such view has focused on building a specialized agent that can assist a user by performing a specific task, such as scheduling an itinerary or ranking and presenting email and news or info-surfing (browsing or searching for information) over the Internet or reasoning how to best support human-computer interactions [4, 9, 20]. A second area of study concerns integrating the performance of sets of these specialized agents [5]. For example, several individuals wishing to get together might have their respective agents schedule a meeting.

Another approach to building an agent entails revising the role of the agent. In this case, I take the position that it takes many "integrated agents" to create an "assistant." In such an approach, many different reasoning processes, societies of agents, are integrated in order to realize a software assistant capable of performing a broad range of tasks. This approach has resulted in the realization of M, a software assistant, "who" attempts to recognize, classify, index, store, retrieve, explain, and present information relating to human-computer interaction in a desktop multimedia conferencing environment. M is a software system that integrates multiple "reasoning agents" whose collaborative results serve to assist a user working together with other individuals in an electronic conference room.

The domain of integrated and multistrategy reasoning and learning has been an active research area studied in a variety of problems and projects—several recommended are [1–3, 6–8, 10, 11, 16]. From my studies in this domain, the resulting design and implementation of M's architecture posed many interesting questions. How do you coordinate and manage a diverse set of agents? How do the agents communicate? What knowledge is required by an agent and is it shared with other agents? Do the agents know of one another and if so, what relationships exist? Does an agent demonstrate "intelligent" behavior or does "intelligent" behavior emerge from the coexistence of the "active" state of the many diverse agents?

In addressing these questions, a design theory for an architecture of integrated agents, influenced by Minsky's Society of Mind (SOM) theory [12, 13], was defined and implemented. In this work, aspects of spatial, structural, functional, temporal, causal, explanation-based, and case-based reasoning (see Note 1 in Glossary) capabilities were integrated in M via (1) a "semantic network" (see Note 2 in Glossary), (2) a "rule-based system" (see Note 3 in Glossary), (3) Minsky's K-lines/polynemes, transframes, and pronomes (see Notes 4–6 in Glossary), and (4) scripts (see Note 7 in Glossary). The basic theory of this work takes the position that an assistant that can classify and explain actions applied to objects within a highly dynamic world should be functionally effective if it can simultaneously generate and test multiple domain theories in relation to a given goal.

In this article I will discuss a general architecture that supports integrated multiple reasoning processes, the "agents." This architecture has been applied in several different domains: (1) a system that learns to compose music [19], (2) intelligent user interface agents [17, 18, 20], and (3) the M assistant that classifies and manages objects in an electronic conferencing system.

The Applied Problem

To test design methods for prototyping such an assistant, several pragmatic tasks were identified relating to

collaborative electronic work environments. M was applied at AT&T Bell Laboratories to perform these tasks within a virtual meeting room (VMR) that supports multimedia desktop conferencing. In a VMR, participants collaborate via pen-based computers and voice (telephone). In this context, the goal of a software assistant is to classify and index the "changing" states of a "world" (see Note 8 in Glossary) in which both the participant(s) and the assistant coexist.

In this world, each user is supported by a personalized assistant and the world is composed of domain objects (e.g., electronic documents, electronic ink, images, markers, white boards, copy machines, staplers) on which users apply actions. The assistant(s) attempt to recognize and define relationships among objects based on the actions applied by the users to the world and the resulting new states of that world. For example, in a VMR world, there may exist a set of domain objects—such as several (electronic) documents. Further, the VMR participants may apply actions to these documents such as annotating them collectively or joining/appending them together. M attempts to identify all domain objects and classify relationships between subsets of objects based on their physical properties and user-applied actions; these actions indicate (semantic) associations between subsets of objects. A simple example of an association would be two adjacent documents a user annotates by drawing a circle to enclose them together. Based collectively on (1) spatial reasoning of the nearness of the two documents and the circle, (2) structural and functional reasoning of the circle enclosing the two documents, and (3) causal reasoning of the semantic action of enclosing objects—M can infer and explain a plausible relationship between the two documents. This functionality provides a new framework for users to work together electronically.

Conceptually, a VMR is a virtual place where one or more persons can work together even though the individuals are physically separated. An example of a VMR is an "electronic place" where individuals physically located in New Jersey and England can meet and work. In a VMR, the individuals share and create information in a variety of media ranging from text to video to images to drawing to audio.

In a VMR, the assistant attempts such tasks as recognizing, classifying, indexing, and explaining all actions applied by users on all the objects present within a VMR over the entire length of a meeting. VMRs also support the functionality of persistence, thus meeting rooms can exist over arbitrarily long periods of time. A VMR is like a real meeting room where individuals can work, leave at the end of a day while leaving behind all documents and other objects, and then return at a later point in time to continue the "work at hand."

If collaborative work environments, such as VMR, are to be successful, then there exists a strong need to improve the expressiveness of the participants working in such places and to minimize many of the required computer-related actions. The following subsections describe several examples of tasks that demonstrate how M attempts to improve the performance of the participants working in a VMR.

Organizing the Electronic Workspace

Consider a typical group of designers working in a brainstorming session held within a real physical room. By the end of such a working session, the designers will have created and used many documents, bullet lists, diagrams, notes, Post-its, and other such items. Based on the properties of a physical room, the participants could organize themselves and the objects in the room using tables, walls, and whiteboards. Documents and other objects could be spatially organized and located for ease of access by the meeting participants. Typically, the designers would be able to view, engage, review, and reformulate various conceptual relationships over all the physical materials and information generated as the meeting progresses.

When the designers' brainstorming session is moved to a VMR, their "view" of the work environment is significantly constrained to the physical size of their respective computer screens (e.g., 1000×1000 pixels at best). What if a software assistant took on the responsibility to organize the creative output and interactions of all the participants? In essence, the assistant assists a user to access and manipulate many different materials created and used during a meeting, independent of where the materials are located within a VMR or when the materials were last used or created.

M can define, generate, and explain an extensive set of relationships and classifications which represent various conceptual information and views of both (1) the complete set of objects created and used by the participants in a VMR and (2) the actions applied to VMR objects by the participants. Thus, each participant can ask, via dialog boxes or direct manipulation techniques, their respective assistant to present organized views of the various related materials used during a meeting.

Functionally, M observes the actions performed by VMR participants and attempts to reason how the current actions applied to VMR objects relate to other VMR objects and previous actions. As a participant interacts with an object, such as an electronic document, M can provide the user with contextual "hyperlinks" to related objects, such as documents, drawings, notes, lists, Post-its, pen annotations, and voice annotations. One of M's fundamental responsibilities is to assist a user to (re)formulate relationships among all objects in a VMR.

Specifically, M attempts to maintain simultaneous theories of how objects in a VMR might relate. This enables M to provide participants with multiple views or access of related materials—thus, M and a user can reformulate the relationships among VMR objects. This functionality is a direct result of M's architecture.

While M maintains an extensive schema for organizing a VMR, the

Glossary

The following notes are provided to define the use of particular terms within this article.

Note 1: Reasoning. The design of M integrates the following modes of reasoning: (1) *spatial reasoning*—a type of reasoning based on properties relating to space, (2) *structural reasoning*—a type of reasoning based on defining relationships between the parts which compose some "object," (3) *functional reasoning*—a type of reasoning based on how some "object" functions—the functional purpose of some "object," (4) *temporal reasoning*—a type of reasoning based on properties relating to time, (5) *causal reasoning*—a type of reasoning that infers from some event that some action involving some object(s) resulted in a state change, (6) *explanation-based reasoning*—a type of reasoning in which a reasoner draws from *a priori* knowledge a set of facts and applicable rules that generate an explanation of a situation via first principles, and (7) *case-based reasoning*—a type of reasoning in which a reasoner solves a new "problem" by analogy of "stored" solutions to old "problems."

Note 2: Spreading Activation Semantic Network. A network of interconnected nodes, where individual nodes represent given properties (e.g., shape, color, size) or objects or concepts, and where these nodes are linked together based on specific relationships—this network serves as a knowledge representation technology. Within the network, "attention" to specific "knowledge" is provided by the "activation" of given nodes which in turn propagate (spread) different levels of "activation" to related nodes.

Note 3: Rule-Based System. A model of processing based on a specific system composed of logical constructs called productions—where the left side of a production, the antecedent, states some condition to be satisfied, and the right side of a production, the consequence, states some action to be performed when the respective condition has been satisfied. In a rule-based system (sometimes referred to as a production system), productions are organized and represented in a collective body commonly referred to as a knowledge base. As the conditions of given productions become satisfied, the actions of these productions direct the performance of the system.

Note 4: K-line/Polyneme. K-line: "When you 'get an idea,' or 'solve a problem,' or have a 'memorable experience,' you create what we call a K-line. This K-line gets connected to those 'mental agencies' that were actively involved in the memorable mental event. When that K-line is later 'activated,' it reactivates some of those mental agencies, creating a 'partial mental state' resembling the original" [12].

Polyneme: "An agent that arouses different activities, at the same time, in different agencies—as a result of learning from experience" [13].

In M, each VMR object is represented by a K-line/polyneme structure that "links" and "activates" the correct property states over all the respective domain object qualifiers in the semantic net.

Note 5: Trans-frame. A particular type of data structure that represents an action, a trajectory between two situations, one for "before" the action occurred and the other for "after."

Note 6: Pronome. "A type of agent associated with a particular 'role' or aspect of a representation—corresponding, for example, to the *Actor, Trajectory,* or *Cause* of some action" [13].

In M, an action is represented by a set of properties: an Actor that caused the action, some set of Object(s) that the action was applied to, and the Before and After features of those Objects. These properties, called pronomes in SOM theory, act as pronouns in a Trans-frame.

Note 7: Scripts. A script is a structure that represents some consistent sequence of events. Its composition includes some <entry conditions> that must be satisfied before a script's events can occur; some <resulting properties>, which are a set of conditions that will be true following a script's sequence of events, and a set of objects and actions representing the sequence of events.

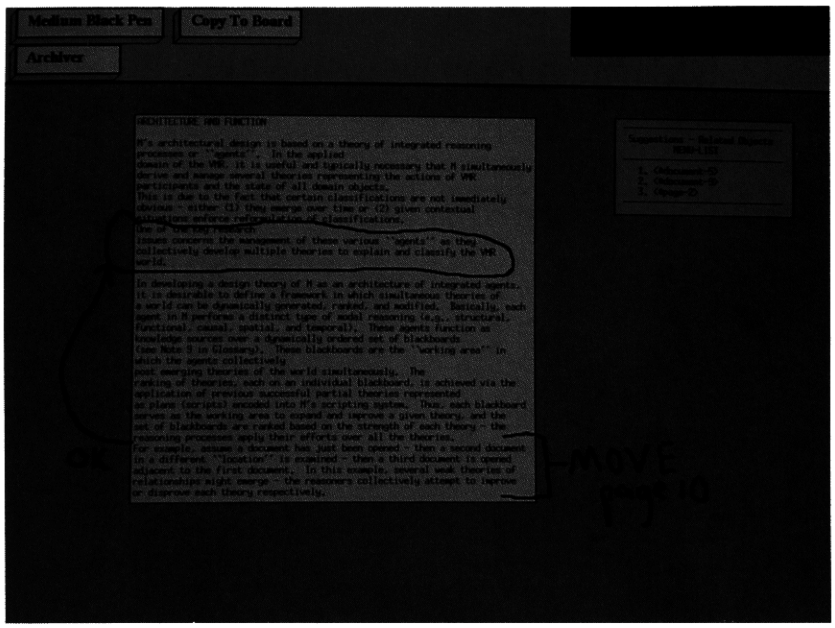
Note 8: World. A structured representation of a state space or domain or problem space. The structure contains knowledge about domain objects, legal actions that can be applied, and the set of legal situations that can exist.

Note 9: Blackboards. H. Penny Nii [15] provides the following description of a blackboard model that is consistent with the use of blackboard system technologies in the design of M. "Organizationally, the blackboard model consists of three components:

1. *The knowledge sources.* The knowledge needed to solve the problem is partitioned into knowledge sources, which are kept separate and independent. [Note: the knowledge sources are M's five reasoners.]
2. *The blackboard data structure.* The problem-solving state data (objects from the solution space) are kept in a global data store, the blackboard. Knowledge sources produce changes to the blackboard that lead incrementally to a solution to the problem. Communication and interaction among the knowledge sources take place solely through the blackboard.
3. *Control.* What knowledge source to apply when and to what part of the blackboard are problems addressed in control.

In addition to the organizational requirements, a particular reasoning (computational) behavior is associated with blackboard systems: The solution to a problem is built one step at a time. At each control cycle any type of reasoning step (e.g., data-driven, goal-driven, forward-chaining, backward-chaining) can be used. The part of the emerging solution to be attended to next can also be selected at each control cycle. As a result, the selection and the application of knowledge sources are dynamic and opportunistic rather than fixed and preprogrammed.

Note 10: Conceptual Dependency. Conceptual Dependency theory defines how to represent knowledge about events—how to represent legal actions applied to objects in a situation of some world. It was developed as an approach to represent knowledge of events expressed in natural language sentences. The goal is to represent knowledge in a way that enables inference of sentences and provides language independence. The theory provides a structure and a set of conceptual primitives to represent an event.



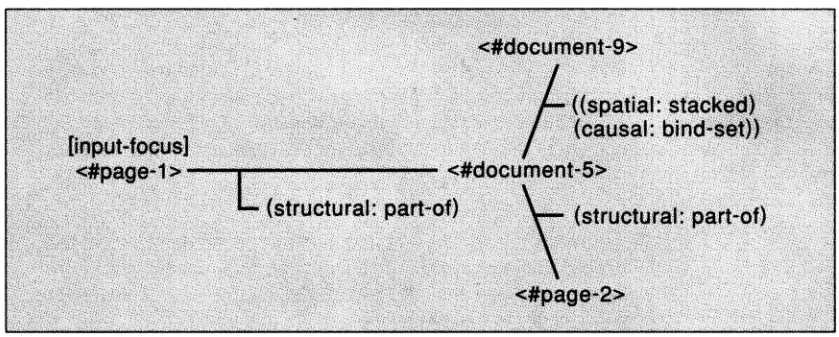


Figure 1. A VMR session where a document is being annotated and a M provided menu-list is presented to provide to access other related objects

Figure 2. M's diagrammatic representations of some of the relationships between the annotated document and the objects presented in the menu-list shown in Figure 1

user must also be allowed to (re)define existing and new relationships and hyperlinks within this schema. M must adhere to an assistant's prime directive: never take control away from the user. Figure 1 displays a computer screen where a participant in a VMR is currently annotating a document. Also present on the computer screen are several other VMR objects and a menu-list provided by M of other VMR objects (currently out of view) that might be related to the annotated document. M can bring these other objects into view and also explain why they are believed to be related. Figure 2 is one of M's diagrammatic representations of some of the relationships that exist in Figure 1.

Grouping Things

One of the services provided by M is organizing relationships between objects and user-applied actions in a VMR. As a direct result of M's classifying functions, a user has access to individual VMR objects and "groups" of related VMR objects. This feature is significant since user actions can be applied at various levels of abstraction representing sets of VMR objects. M classifies objects as members of sets based on object properties and user-applied actions. For example, all text documents or all annotated bullet lists form respective sets based on their physical properties. As another example, a sequence of actions over some space and time can form a set—consider a set of objects including a text document, a computer graphic image placed on (in) the document, and some annotations on the document. Based on a sequence of user-applied actions to visually integrate the text, image, and annotation, these 3 objects can be viewed as a set based on M's spatial, temporal, structural, and functional knowledge of composing objects by integrating such objects as text, image, and annotations. The notion of classifying objects as members of sets allows the users and M to perform such important set operations as: (1) evaluating sets (e.g., explain the composition of a set); (2) comparing sets (e.g., explain how sets differ and intersect); (3) joining sets; (4) coping sets; and (5) moving sets.

By grouping related things, an assistant reduces user-applied actions. Such tasks as "grouping" visual objects in a drawing package in order to apply a global action (e.g., the spatial movement of a group of visual objects) could be performed by an assistant. If for example, the assistant had already (1) defined relationships and hyperlinks over groups of objects and (2) told the user where "groups" existed, then the user might only have to minimally modify a group before moving the group. In a VMR, if a participant moves an object in a defined group, M can move the entire group relative to the action applied by the participant while addressing all existing constraints present in the current state of the VMR.

As a visual example, Figure 3 displays a group of related VMR objects composed of two individual documents and some pen annotations. This group is seen along with some other objects not in the group. Figure 4 displays the resulting visual presentation of this group after a user moved one of the documents. Figure 5 displays a second resulting visual presentation of this group after the same document is moved again. In both moves, the task of grouping was performed by M. Notice that in the second move, M adjusted the annotation due to existing spatial constraints imposed by other objects.

While moving groups of visual objects on a computer screen addresses one type of task, it is useful to con-

sider that movement can be visual, physical, virtual, and conceptual. M could apply the function of "grouping" to such tasks as transferring information to another person (e.g., transmitting a set of multimedia documents via email), performing queries over sets of information, and copying sets of documents to a specified storage resource.

Another task relating to the grouping of things is enumeration. For example, if a repetitive action being performed by a user over a group of objects is observed, then M could suggest that the user apply the action globally on the group or ask to complete the task on the remaining objects for the user. The ability of M to provide user access to groups can improve the interaction between users and their computers.

Architecture and Function

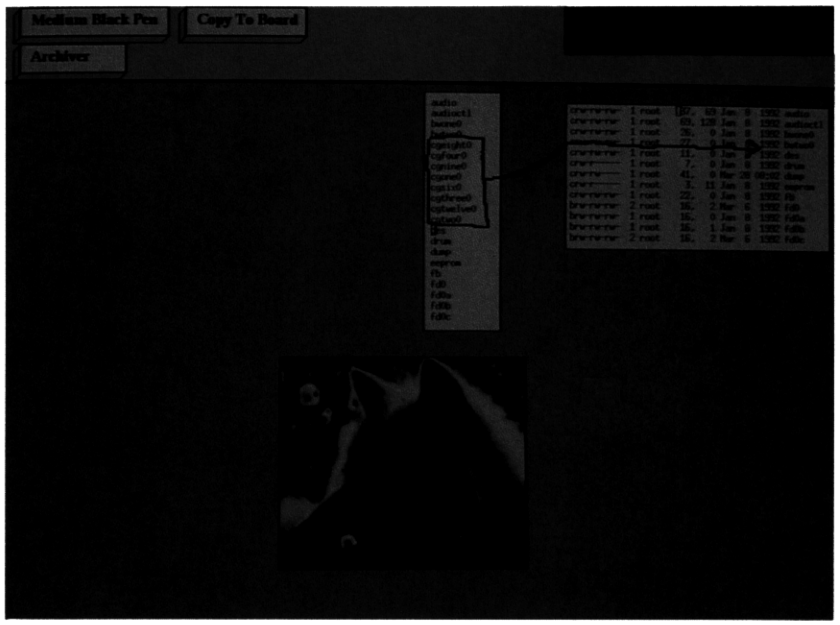
M's architectural design is based on a theory of integrated reasoning processes or "agents." In the applied domain of the VMR, it is useful and typically necessary that M simultaneously derive and manage several theories representing the actions of VMR participants and the state of all domain objects. This is due to the fact that certain classifications are not immediately obvious—either (1) they emerge over time or (2) given contextual situations enforce reformulation of classifications. One of the key research issues concerns the management of these various "agents" as they collectively develop multiple theories to explain and classify the VMR world.

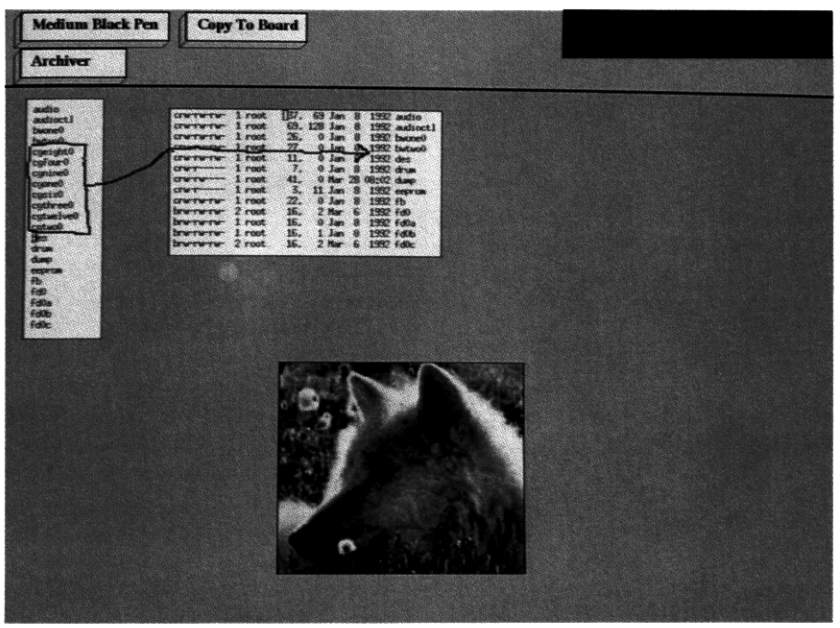
In developing a design theory of M as an architecture of integrated agents, it is desirable to define a framework in which simultaneous theories of a world can be dynamically generated, ranked, and modified. Basically, each agent in M performs a distinct type of modal reasoning (e.g., structural, functional, causal, spatial, and temporal). These agents function as knowledge sources over a dynamically ordered set of blackboards (see Note 9 in Glossary).

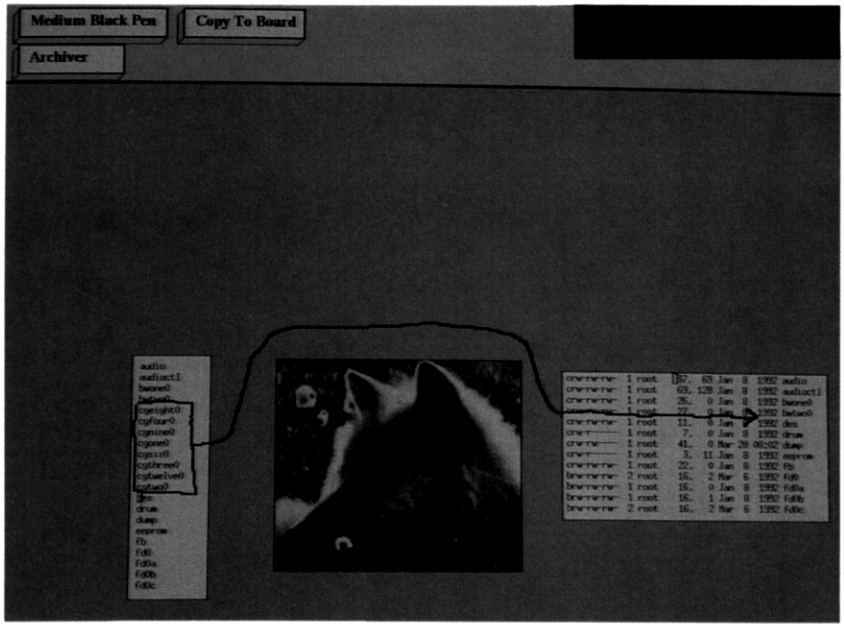
These blackboards are the "working area" in which the agents collectively post emerging theories of the world simultaneously. The ranking of theories, each on an individual blackboard, is achieved via the application of previous successful partial theories represented as plans (scripts) encoded into M's scripting system. Thus, each blackboard serves as the working area to expand and improve a given theory, and the set of blackboards are ranked based on the strength of each theory—the reasoning processes apply their efforts over all the theories. For example, assume a document has just been opened. A second document in a different "location" is subsequently examined, and then a third document is opened adjacent to the first document. In this

Figure 3. A VMR session where two documents and some pen annotations are grouped

Figure 4. The new presentation of the grouped objects after a user moved only the document with the annotated rectangle drawn in it







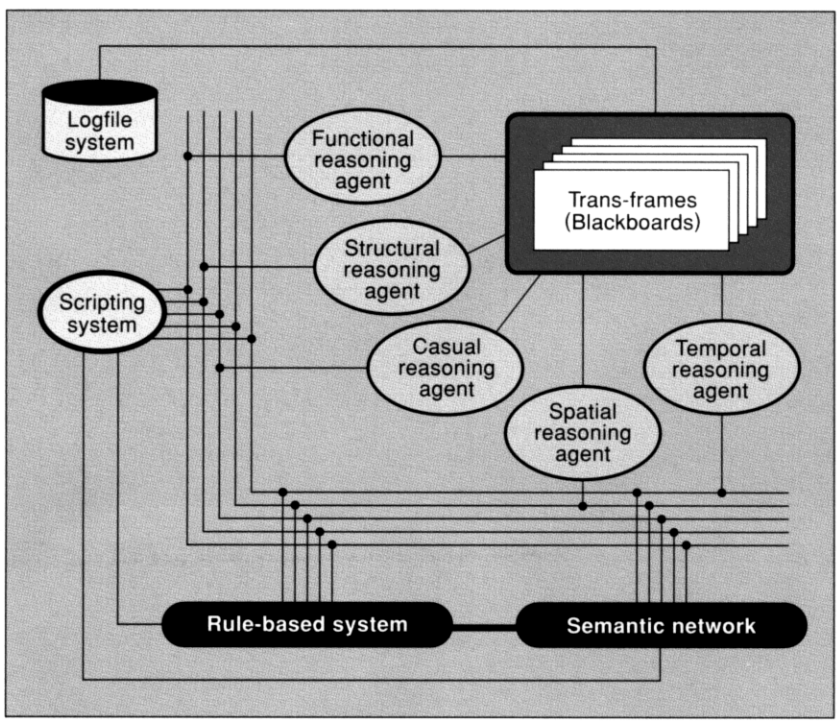


Figure 5. The new presentation of the grouped objects after a user moved only the document with the annotated rectangle drawn in it a second time. Notice that M adapted the annotations due to other spatial constraints.

Figure 6. The M software architecture

example, several weak theories of relationships might emerge—the reasoners collectively attempt to improve or disprove each theory respectively.

We can explore how M functions and adapts its performance via a general discussion of the components comprising M's system architecture. M consists of the following 6 subsystems: (1) a spreading activation semantic network (to realize Minsky's K-lines/polynems), (2) a "rule-based" system, (3) a set of blackboards (to realize Minsky's transframes and pronomes), (4) a scripting system, (5) a history "logfile" system, and (6) an I/O system. Figure 6 displays the M software architecture.

When a VMR event occurs, such as an individual annotating a document or moving one piece of paper on top of another, M's **I/O system** is responsible for representing "who did what." This is achieved as follows: (1) note which actor, VMR participant or VMR object, performed the action that created the event, (2) note the type of action (e.g., annotating, moving, cutting), and (3) note the VMR object(s) which the action was applied to. This input information is represented in an input "record." Based on Pazzani's input technique for his OCCAM system [16], which employs characteristics of Schank's Conceptual Dependency theory [21, 22] (see Note 10 in Glossary), M's input record represents an action applied to some object(s); this information entails a set of facts representing a state change of some object(s).

Formally, each input record identifies an ACT consisting of an action-type-identifier, an actor, an object, and a from-to qualifier. For example, an ACT represents an ACTOR (a VMR participant or object) that performed a specific type of legal ACTION on a given OBJECT and a set of properties associated with the object (e.g., spatial position, composition/structure, size) that have changed FROM some existing value TO a new value. Figure 7 displays an input record where the ACTOR <doug> applied a <move> ACTION to the OBJECT <book23> which moved the object FROM <coordinate(x1, y1)> TO <coordinate(x2, y2)>. When a VMR action is applied, the I/O system asserts the facts in the input record to the semantic net.

The **semantic net** is implemented as a spreading activation semantic network over sets of qualifiers (e.g., size, position, color) that collectively represent domain objects. These qualifiers represent the facts associated with an applied action denoted in an input record. Each qualifier acts as a state machine, a "society of property agents," which can represent one

of *n* legal states representing the current property of a specific qualifier which in turn represents a specific property of a domain object. For example, a domain object qualifier representing *shape* is a state machine that can represent a range of shapes such as square or circle. The basic idea is this: when an actor or object is identified via the I/O system, the corresponding qualifiers within the net collectively become active representing the correct property states of the respective object. As these qualifiers become active in a specific state, they become facts which directly influence the rule-based system.

A critical aspect of the M design is the application of Minsky's K-line theory [12, 13]. Each distinct domain object is uniquely represented as a K-line (polyneme) structure that "links" and "activates" the correct property states over all the respective "societies of property agents" in the semantic net. The activation of these agents enumerates a set of facts about the VMR.

M's **rule-based system** performs several significant functions. As various facts (in the semantic net) are asserted to be true, these facts then satisfy specific preconditions expressed in the antecedent of given rules. Thus, when the antecedents of such rules are evaluated as true, this enables the consequence of each respective rule to be asserted. The actions specified in the consequence of the rules that "fire" provide the following two results: First, new facts expressed in a rule's consequence are applied (activated) respectively to the semantic net; this then can have an iterative effect over the firing of new rules and the instantiation of other facts. Second, as new rules fire and new facts are instantiated, M's reasoning processes, the "agents," apply this new information to strengthen or weaken or create or purge the various theories explaining a VMR world.

As various facts and rules evaluate as true, this directly influences the scripting system and the behavior of the reasoning processes as they evaluate and apply various scripts of partial plans provided by the scripting system. In essence, we can view the rule-based system as a collection of domain methods to organize facts and bias the selection of partial plans by M's reasoners to create and explain relationships among objects.

The **scripting system** is a corpus of partial plans, scripts [23] that have demonstrated frequent success in previous classification problems. In M, a script is a partial ordering of elements in a set. The set represents an interval of time over which a consistent pattern of facts and rules have frequently been applied successfully to predict the state of some object(s) following some action.

A critical feature of the scripting system entails its use of coefficients to weight each script's potential to either initiate or improve on a theory that attempts to classify and explain some set of actions, objects, and relationships within a VMR. Functionally, these weighted scripts bias the various reasoning "agents" to dynamically rank all coexisting theories where each theory is formulated on one of the individual blackboards. These weighted scripts serve to minimize combinatoric growth of all possible classification theories available to the reasoning agents. The agents will select weighted scripts which formulate or improve the top seven ranked theories.

As an example, let us assume that some VMR participants have just "opened" an electronic document. After several actions have been applied to this document, they "move" to another location in the VMR. During this time, they perform some actions on other objects (for this example it is not necessary to consider the relationships that might exist among the actions on the first document and the other objects). Let us now assume that the participants (1) move back to the location of the first document, (2) open a second document near the first document, and (3) perform some actions only on the second document. At this point, a "weak" theory of relationship between the two documents might be suggested by a partial plan based on spatial and structural properties. For this example, assume the "weak" theory is suggested due to a corresponding script being applied that has a "low" coefficient. Now let us assume that the participants then move spatially to some other objects, perform some actions, but then return to the location of the two adjacent documents. As they repeatedly return to this same location, there probably is no rule or script that significantly strengthens the relationship between the two documents, but should a participant drag one of the documents toward the other or cut-and-paste a portion of one into the other or annotate with electronic ink (via a pen) using specific symbols or gestures across both documents, then a corresponding stronger weighted script could be applied to suggest a "strong" theory of relationship.

These weighted scripts serve a second design task. Over time M can dynamically adjust the weights of the scripts in order to adapt its performance to the behavior and needs of an individual user. Basically, M adapts its performance as an assistant by (1) generating new rules and scripts and (2) adjusting the weights of individual scripts.

M's **blackboard system** provides a shared "work area" in which the individual reasoning processes develop various classification theories about the world. The blackboard system consists of a dynamic set of ranked blackboards, which are allocated and deallocated as needed. The maximum number of blackboards allocated at any given moment is seven.

Figure 7. A typical M input record

```
ACTOR: (#doug ((class:participant) (host: wolfgang)))
ACTION: (move-object)
OBJECT: (#book23 (class:book))
FROM: (coordinate (100, 100))
TO: (coordinate (150, 200))
```

Each blackboard contains an emerging classification theory over some subset of actions and objects. Basically, an emerging theory can be thought of as a hypothesis to be proved by M's reasoners (in the context of a blackboard system the reasoners can also be referred to as knowledge sources). M's reasoners attempt to develop a strong theory by individually applying "axioms" to a given theory's hypothesis on a blackboard. As new facts are asserted as true, new rules fire, and new scripts become applicable. M's reasoners collaborate by applying these facts, rules, and scripts as axioms to the respective blackboard. Further, as M computes the weighted scripts for each blackboard, the theories with the greatest weighted "sum" are ranked high to low, thus defining how to dynamically rank the blackboards.

When a reasoner posts an axiom to a blackboard, this information can be viewed either as some type of modal information reflecting a modality of reasoning (e.g., spatial or temporal) or some set of "Conceptual Dependency information" representing an action. New modal information describes an object and is posted into a graph-based schema (e.g., like a "semantic net") as described by Winston et al. [24] and Mitchell et al. [14]. The Conceptual Dependency information includes such qualifiers as actor, action, object, goal, origin, and destination. This information describes an action associated with pre- and post-states of the world, as based on Minsky's trans-frames in SOM theory [13]. Both types of information are extremely useful.

For example, in the first case of model information, if a functional reasoning "agent" posts specific functioning features of an object to a blackboard, then this new information can serve to bias a structural or causal reasoning agent to derive structural or causal features of the object with strong correspondence with the posted functional features. As for the second type of information, the Minsky trans-frames are a significant architectural feature to represent different "things" and theories on individual blackboards using Conceptual Dependency schema. The trans-frame approach provides a canonical form that enables M to effectively compare different theories or subtheories posted over the ranked blackboards.

Both types of information, the graph-based schema and the trans-frame schema, serve one other significant function. Both representations are symbolic structures providing symbols (names) to identify each schema element (think of a schema as a record—each schema element as a field in the record and each field is identified by a name). Based on these symbolic identifiers of each schema element and the respective semantics, M can generate explanations describing relationships over VMR objects and actions.

After M develops a simultaneous set of classification theories of the state of a VMR, these theories are then collapsed from their respective blackboards into a ranked set of linked data structures. These data structures are then indexed into M's **history logfile system** and their respective blackboards are then deallocated. This allows new blackboards to be allocated for the next set of classification tasks.

As M continues to process new VMR events and generate new classifications, all information in the logfile system is retrievable and available to the reasoning agents. All information stored in the logfile system is indexed based on the temporal ordering of VMR events, the spatial position of VMR events, and the individual identifiers to index and access each VMR object directly. This allows M to reformulate previous theories based on the sequence of VMR actions that continue to occur. Basically the history logfile system is a complex representation of an entire VMR world over time.

Knowledge Required

M's five reasoning agents share a common corpus of domain knowledge relating to the VMR. Conceptually, this knowledge consists of individual class definitions representing each type of VMR object and the legal actions performed by each object. Each definition entails a set of qualifiers represented as facts in the semantic net and rules in the rule-based system. The facts define descriptive properties of each domain object and relationships between domain objects. The rules define (1) methods applied on or by each object and (2) actions that focus the problem-solving process to classify relationships over the current state of all VMR objects and user input events.

The application of these knowledge class definitions simplified the design effort of M and serves to minimize future modification and maintenance functions. Further, these classes provide logical conceptualization for a designer to realize a distinct function provided by a given class and the relationships that exist when given classes are applied in conjunction to define new classes. M's current knowledge is partitioned over five general classes. These classes are individually composed of subclass definitions representing distinct VMR objects. The five general classes are as follows:

- Primitive drawable objects: Knowledge concerning general properties of VMR objects on which users may perform such actions as typing text, annotating with pens, and pasting other visual graphics and images. Several examples of primitive drawable objects include pages of paper, whiteboards, bulletin boards, and Post-its.
- Composite drawable objects: Knowledge concerning general properties of VMR objects composed of drawable objects (e.g., a chapter in a notebook composed of pages of paper, a notebook composed of chapters).
- Drawing tools: Knowledge concerning general properties of tools that generate graphical/visual output" (e.g., pens, erasers, paint brushes, keyboards, cameras, fax machines, scanners).
- Primitive editing tools: Knowledge concerning general properties of tools that apply actions (e.g., cut, join, move, transform) to graphical VMR objects (e.g., paper, images).

• Composite editing tools: Knowledge concerning general properties of editing tools composed of editing tools (e.g., a stapler that performs joins, a "copy machine" that performs cut, copy, and move).

Representing the individual domain objects from all five general classes is realized via a set of qualifiers. Obtaining the "best" set of qualifiers is critical for the following two reasons. First, these qualifiers enable M to derive plausible classifications over the current state of a VMR. To define a useful set of qualifiers, it is first necessary to identify the types of reasoning required of M to solve the types of problems discussed earlier. Based on M's multistrategy reasoning approach, it is then necessary to identify useful features and properties that are representative of the domain objects and their legal domain actions. These features and properties will serve as the qualifiers applied by M to determine solutions to various classification problems. The second reason that selecting qualifiers is critical addresses a human-computer interaction issue. It is important that the qualifiers demonstrate strong correspondence in the manner in which they are perceived both by humans and M. This attempts to ensure a consistent view, by both M and the VMR participants, of the various reasoning processes that are applicable to VMR classification tasks.

A key design feature in the selection of qualifiers for M was the importance of identifying contextual properties associated with objects and actions in a VMR, not to actually determine the content of the VMR objects. Some of the most general qualifiers applied over all the domain objects are: (1) type identifier, (2) author, (3) group and affiliation, (4) conceptual set, (5) content, (6) composition, (7) appearance and geometry, (8) color, (9) time and date, (10) spatial, (11) service and functionality, and (12) origin.

The *service and functionality* qualifier enumerates all the legal actions that a given object can perform. This knowledge is critical, since this enables M to apply functional descriptions and reasoning to complement structural and causal reasoning processes.

A *type identifier* denotes a specific type of VMR object (e.g., piece of paper, whiteboard, picture image, multimedia document). In some cases, this identifier may also include a "tag field" containing several keywords or phrases defining a user's purpose for the object. The tag field is optionally provided by a user. The *author* qualifier identifies the person responsible for an event. It also defines an individual as the author of some VMR object just created. Typically, an individual is associated with some *group* of persons and some type of *affiliation* in an organization. For example, I am a member of the Computer Systems Research Laboratory (the group) and I am an employee of AT&T Bell Laboratories (the affiliation). These qualifiers enable M to infer relationships to actions applied over time by myself and other persons based on our various group and affiliation relationships. The *origin* qualifier identifies sources of information accessed by participants. Several examples include fax, email, netnews, voicemail, and electronic news, journals, newspapers, magazines and books. Identifying an information source has many useful characteristics. For example, based on <origin> and <functionality> qualifiers, M can classify documents derived from information extracted from a real-time electronic news service as being "very recent or new information" compared with information extracted from an "electronic book."

Sometimes VMR participants group objects into *conceptual sets*. These actions are significant in various ways. This qualifier identifies that under a specified context, a given object belongs to a set of objects. For example, a piece of paper might be positioned by a participant near or on top of other pieces of paper to form a "set" (at an implementation level—each piece of paper is realized via a graphic-generated window possibly containing textual information—thus the windows are stacked). Several other examples of manual set construction by participants includes binding pages into documents or binding documents as chapters into a book or notebook. Another example demonstrates a different context in the application of this qualifier. Consider that several distinct objects might be spatially adjacent to one another during some interval of time in a meeting. M's ability to represent and replay this history of visual/spatial information is a useful feature for VMR participants to recall some earlier context.

In some cases it is possible to represent the *content* of a VMR object. This is achieved via tag fields and the application of "information retrieval" indexing techniques. While M's ability to qualify an object based on content is limited, it still is a useful feature and an important area for future work. An object can also be qualified based on its *composition*. For example, a document might be comprised of several different media, such as text, audio, graphics, and video. This knowledge supports various structural reasoning processes. Another qualifier concerns the visualization of an object, its *appearance and geometry*. These features include the object's size, shape, and presentation as being 2-dimensional (2D) or 3-dimensional (3D). These qualifiers also support various structural and spatial reasoning processes. The *color and texture* of an object provides M with another useful property to classify actions. For example, sometimes VMR participants listing information on a whiteboard will intentionally select different color electronic markers as they generate the list. It is possible to assume that the author of the list is noting some type of classification over subsets of the list based on the different colors.

Time and date information identifies when an action occurred. Several examples include the time and date when an object was first created and when an action was last applied on the object. Based on a given task, M applies such qualifiers to derive various partial orderings of temporal events relating to different objects. The *spatial* selection by VMR participants to apply actions is a significant qualifier. Participants can organize

objects over the virtual space of a VMR. To extend this thought, remember our discussion of a design meeting in a real physical conference room—through the aid of multimedia and multimodal technologies, it is possible to simulate the spatial context of the real room. Thus, spatial reasoning over a simulated space provides another classification technique.

Summary

I have described a general architecture, implemented in the M system, which integrates multiple reasoning processes. This architecture allows the "reasoners" to collectively formulate and examine multiple classifications of domain objects simultaneously. This approach is useful for classification problems that exhibit time-variant characteristics; that is, the strength of a relationship between objects can vary over time. In order to minimize extensive combinatoric growth of individual classification trajectories, a weighted (ranked) set of partial plans represented in M's scripting system "guides" the reasoners.

There are several key aspects concerning the design of M's architecture. First, M integrates causal, functional, structural, spatial, and temporal reasoning. Second, M's architecture is composed of six components: a "semantic net," a "rule-based" system, a "scripting system," a dynamically ranked set of blackboards a logfile, and an I/O system. Third, the "M view" of agents consists of the five reasoning agents, but also includes a hierarchical view of "societies of agents" built throughout all of M's system components (e.g., the qualifier property agents that compose the semantic net); this perspective is based on SOM theory. Also, Minsky's trans-frames, implemented as "blackboards", are a significant component that serves to (1) represent and compare theories of classified objects based on sequences of legal actions and (2) provide a short-term memory resource.

So how should we think about "agents"? I hope that in time there will exist a vocabulary to communicate what we "mean" in the various situations in which the word agent is presently used. In one specific context, I prefer the word "assistant." I see an assistant as an "intelligent machine," biological or nonbiological, composed of many "agencies of agents," not an agent being an intelligent machine. You see, to be an assistant you need to understand many things, and since (in the English language) an agent is some type of "specialist," you would probably not want an agent. Instead, you would probably want an assistant endowed with the talents of many integrated agents.

Acknowledgments

The author is deeply indebted to Marvin Minsky, Ed Pednault, and Stacy Marsella for their helpful interest and comments in this research. I especially wish to thank Sudhir Ahuja, Wayne Armour, John Carson, Yann LeCun, and David C. Smith for valuable discussions concerning my work.

References

1. Bond, A.H. and Gasser, L. In *Readings in Distributed Artificial Intelligence*. A.H. Bond and L. Gasser, Eds. Morgan Kaufmann, San Mateo, Calif., 1988.
2. de Kleer, J. Multiple representations of knowledge in a mechanics problem-solver. In *Proceedings of IJCAI, 1977*. Morgan Kaufmann, San Mateo, Calif., 1977.
3. Engelmore, R. and Morgan, T. In *Blackboard Systems*. R. Engelmore and T. Morgan, Eds. Addison-Wesley, Menlo Park, Calif., 1988.
4. Etzioni, O. and Weld, D. A softbot-based interface to the Internet. *Commun. ACM 37*, 7 (July 1994).
5. Genesereth, M.R. and Ketchpel, S.P. Software agents. *Commun. ACM 37*, 7 (July 1994).
6. Gasser, L. and Huhns, M.N. In *Distributed Artificial Intelligence Vol. II*. L. Gasser and M.N. Huhns, Eds. Morgan Kaufmann, San Mateo, Calif., 1989.
7. Guha, R.V. and Lenat, D.B. Enabling agents to work together. *Commun. ACM 37*, 7 (July 1994).
8. Huhns, M.N. In *Distributed Artificial Intelligence*. M.N. Huhns, Ed. Morgan Kaufmann, San Mateo, Calif., 1987.
9. Maes, P. Agents that reduce work and information overload. *Commun. ACM 37*, 7 (July 1994).
10. Michalski, R. Special issue on multistrategy learning. In *Machine Learning*, R. Michalski, Ed. *11*, 2/3 (May/June 1993) Kluwer Academic Publishers, Netherlands, 1993.
11. Michalski, R. and Tecuci, G. In *Machine Learning, A Multistrategy Approach, Vol. 4*. R. Michalski and G. Tecuci, Eds., Morgan Kaufmann, San Mateo, Calif., 1994.
12. Minsky, M. K-lines: A theory of memory. *Cog. Sci. 4* (1980) 117–133.
13. Minsky, M. *The Society of Mind*. Simon and Schuster, New York, 1985.
14. Mitchell, T.M., Keller, R.M., and Kedar-Cabelli, S.T. Explanation-based generalization: A unifying view. In *Machine Learning*, R. Michalski, Ed. *1*, 1, Kluwer Academic Publishers, Netherlands, 1986.
15. Nii, H.P. Introduction. In *Blackboard Architectures and Applications*, V. Jagannathan, R. Dodhiawala, and L. Baum, Academic Press New York, 1989.
16. Pazzani, M. *Creating A Memory of Causal Relationships: An Integration of Empirical and Explanation-Based Learning Methods*. Lawrence Erlbaum, Hillsdale, N.J., 1990.
17. Riecken, D. Adaptive Direct Manipulation. In *Proceedings of IEEE International Conference of Systems, Man and Cybernetics*, (Charlottesville, Va, 1991), pp. 1115–1120.
18. Riecken, D. Auditory adaptation. In *Proceedings of IEEE International Conference of Systems, Man and Cybernetics*, (Charlottesville, Va, 1991), pp. 1121–1126.
19. Riecken, D. WOLFGANG—A system using emoting potentials to manage musical design. In *Understanding Music with AI: Perspectives on Music Cognition*, M. Balaban, K. Ebcioglu, and O. Laske, Eds. AAAI/MIT Press, Cambridge, Mass., 1992.
20. Riecken, D. Human-machine interaction and perception. In *Multimedia Interface Design*, M. Blattner and R. Dannenberg, Eds. ACM Press, New York, 1992.
21. Schank, R.C. Identification of conceptualizations underlying natural language. In *Computer Models of Thought and Language*. R.C. Schank and K.M. Colby, Eds., Freeman, San Francisco, Calif., 1973.
22. Schank, R.C. *Conceptual Information Processing*. North-Holland, Amsterdam, 1975.
23. Schank, R.C. and Abelson R.P. *Scripts, Plans, Goals, and Understanding*. Lawrence Erlbaum, Hillsdale, N.J., 1977.

CONTINUED ON PAGE 146

Acknowledgments

Steven Ketchpel and Chris Ramming implemented various parts of the system. We thank Ron Brachman, Oren Etzioni, Mark Jones, and Eric Sumner for useful suggestions and feedback.

References

1. Dent, L., Boticario, J., McDermott, J., Mitchell, T. and Zabowski, D. A personal learning apprentice. In *Proceedings of AAAI-92*, AAAI Press/The MIT Press, 1992, pp. 96–103.
2. Etzioni, O., Hanks, S., Weld, D., Draper, D., Lesh, N. and Williamson, M. An approach to planning with incomplete information. In *Proceedings of KR-92*, Morgan Kaufmann, 1992, pp. 115–125.
3. Etzioni, O., Lesh, N. and Segal, R. Building softbots for Unix. Tech. Rep., University of Washington, Seattle, Wash., 1992.
4. Maes, P., Ed. *Designing Autonomous Agents*. MIT/Elsevier, 1993.
5. Shoham, Y. Agents-oriented programming. *Artif. Intell. 60* (1993) 51–92.

About the Authors:

HENRY A. KAUTZ is a research scientist at AT&T Bell Laboratories. Research interests cover various topics in artificial intelligence, including fast approximate reasoning algorithms, planning, and software agents.

BART SELMAN is a research scientist at AT&T Bell Laboratories. Interests include randomized algorithms for satisfiability testing, the complexity of commonsense inference, and various topics in artificial intelligence.

Author's Present Address: Kautz and Selman can be reached at AT&T Bell Laboratories, 600 Mountain Ave., Murray Hill, NJ 07974; email {kautz, selman}@research.att.com

MICHAEL COEN is a graduate student at the MIT Artificial Intelligence Laboratory. He is interested in tools that help simplify software agent construction and provide guarantees of behavioral correctness. **Author's Present Address:** MIT Artificial Intelligence Laboratory, 545 Technology Square, NE43-823, Cambridge, MA 02139; email mhcoen@ai.mit.edu

CONTINUED FROM PAGE 40

view. In B. Laurel, ed., *The Art of Human-Computer Interface Design*. Addison-Wesley, Reading, Mass., 1990.
10. Kozierok, R. and Maes, P. A learning interface agent for scheduling meetings. In *Proceedings of ACM SIGCHI International Workshop on Intelligent User Interfaces*. ACM Press, N.Y., 1993, pp. 81–88.
11. Kozierok, R. A learning approach to knowledge acquisition for intelligent interface agents. SM Thesis, Department of Electrical Eng. and Computer Science, MIT, May 1993.
12. Lai, K. Malone, T. and Yu, K. Object Lens: A "spreadsheet" for cooperative work. *ACM Trans. Office Inf. Syst. 6*, 4 (Apr. 1988), 332–353.
13. Lashkari, Y., Metral, M. and Maes, P. Collaborative interface agents. In *Proceedings of the National Conference on Artificial Intelligence*. MIT Press, Cambridge, Mass., 1994.
14. Laurel, B. Interface agents: Metaphors with character. In B. Laurel, ed., *The Art of Human-Computer Interface Design*. Addison-Wesley, Reading, Mass., 1990.
15. Lieberman, H. Mondrian: A teachable graphical editor. In A. Cypher, ed., *Watch what I do: Programming by Demonstration*, MIT Press, Cambridge, Mass., 1993.
16. Maes, P. and Kozierok, R. Learning interface agents. In *Proceedings of the AAAI'93 Conference*. MIT Press, Cambridge, Mass., pp. 459–465.
17. Myers, B. *Creating User Interfaces by Demonstration*. Academic Press, Boston, Mass., 1988.
18. Myers, B., ed. Demonstrational interfaces: Coming soon? In *Proceedings of CHI'91*. ACM Press, N.Y., 1991, pp. 393–396.
19. Negroponte, N. *The Architecture Machine; Towards a more Human Environment*. MIT Press, Cambridge, Mass., 1970.
20. Salton, G. and McGill, M. *Introduction to Modern Information Retrieval*. McGraw-Hill, N.Y., 1983.
21. Schneiderman, B. Direct manipulation: A step beyond programming languages. *IEEE Comput. 16*, 8 (Aug. 1988), 57–69.
22. Sheth, B. and Maes, P. Evolving agents for personalized information filtering. In *Proceedings of the Ninth Conference on Artificial Intelligence for Applications*. IEEE Computer Society Press, 1993.
23. Sheth, B. A learning approach to personalized information filtering. SM Thesis, Department of Electrical Eng. and Computer Science, MIT, Feb. 1994.
24. Stanfill, C. and Waltz, D. Toward memory-based reasoning. *Comm. ACM 29*, 12, (Dec. 1986), 1213–1228.
25. Sullivan, J.W. and Tyler, S.W., eds. *Intelligent User Interfaces*. ACM Press, N.Y., 1991.

This research is sponsored by Apple Computer, by the NSF under grant number IRI-92056688, and by the News in the Future Consortium of the MIT Media Laboratory.

CONTINUED FROM PAGE 116

24. Winston, P.H., Ginford, T.O., Katz, B., and Lowry, M. Learning physical descriptions from functional definitions, examples, and precedents. In *Proceedings National Conference on Artificial Intelligence*, AAAI/MIT Press, Cambridge, Mass., 1983, pp. 433–439.

About the Author:

DOUG RIECKEN is a principal investigator in the Computer Systems Research Laboratory at AT&T Bell Laboratories. Current research interests include artificial intelligence, multimedia technologies, and human-computer interactions. **Author's Present Address:** Computer Systems Research Laboratory, AT&T Bell Laboratories, Room 4F-611, Crawfords Corner Road, Holmdel, NJ 07733-3030; email: wolf@research.att.com