Virtual Mattie—an Intelligent Clerical Agent

by Stan Franklin, Art Graesser, Brent Olde, Hongjun Song, and Aregahegn Negatu

Institute for Intelligent Systems
The University of Memphis

stan.franklin@memphis.edu

Abstract

One important role for autonomous agents is to assume tasks previously performed by humans. Such tasks often require communication with humans, and the coordination of multiple activities. What sort of agent architecture will empower an agent to collaborate with humans in the process of autonomously carrying out an every day clerical task? What if this task requires the coordination of several different activities, including the composition of simple documents? Here we propose such an architecture and include a preliminary report on its first implementation. This architecture enables a softbot, Virtual Mattie, to actively gather information from humans, compose announcements of next week's seminars, and mail them each week to a list that she keeps updated, all without the supervision of a human. VMattie's architecture combines Maes' behavior net architecture and Hofstadter and Mitchell's Copycat architecture and significantly extends them. ``Living'' in a UNIX environment, VMattie's communication with humans is entirely via email with no agreed upon protocol. Thus natural language processing in a narrow domain is required. She also warns seminar organizers of impending time or place conflicts. In future versions, VMattie will learn the habits of organizers in order to more effectively dun them for information. With her low level operations based entirely on codelets, VMattie can also be viewed as a multiagent system. The implementation is in Java and Perl. VMattie is also an interdisciplinary project, leaning heavily on a psychological analysis of a corpus of the real Mattie's messages as a basis for natural language understanding.

Introduction

The task described here is to design, build, and experiment with an intelligent autonomous clerical agent. Our agent will be a software agent "living" in a UNIX system (Franklin and Graesser 1996). It will be intelligent, with multiple high level perceptions and actions satisfying several drives, all happening in a complex dynamic environment. Performing a task normally the duty of a clerical worker, it will do so without any human intervention whatsoever, making it autonomous. Email communications with humans using natural language is required.

The high level design stage, resulting in a new control architecture, is complete. As this is written we're in the midst of the low-level design phase that leads to coding. By the camera-ready copy date, a prototype should be up and running.

Clerical tasks requiring only email communication abound, for example some payroll clerk's tasks, and some travel clerk's tasks (not travel agents). Agents capable of automating such tasks should prove individually of small, but significant value, by freeing a clerical person for other duties. Collectively, they'll be immensely valuable by reducing personnel costs and improving efficiency.

VMattie

The particular demonstration task we've chosen is to produce an agent, Virtual Mattie, that each week emails an announcement of the next week's seminars scheduled in an academic department. VMattie will communicate with human seminar organizers by email, compose seminar announcements, send them in a timely fashion to a mailing list that she

maintains. She'll also dun organizers for their information when it's not forthcoming, and mail addenda and errata as needed. In a latter incarnation, she'll also handle one-time colloquia.

Though these tasks aren't difficult for the real Mattie, it requires high-level perception in the form of recognition and understanding in dealing with free-form, typically incomplete, email messages from both organizers and listees. Understanding requires inferencing. The task also requires keeping track of dates and deadlines, of default seminar information, and of email addresses of both organizers and listees. And, VMatiie must compose and send email messages of several types such as announcements, addenda, errata, reminders, and acknowledgments. These tasks are by no means trivial, given the current state of our knowledge of software agents. Much should be learned while solving the problems that will inevitably arise with human-VMattie communication, and with implementing the underlying architecture.

Development of VMattie requires us to formulate, implement and observe a new, high-level software agent architecture. This architecture leans on, and significantly extends, two previous architectures, Hofstadter and Mitchell's copycat architecture (1994, Mitchell 1993), and Maes' behavior networks (1990, 1992) [For brief accounts of these architectures see Franklin 1995]. A Maes network, extended to deal with variables, with variable goals, and with activation arising from internal states, provides informal planning and action selection. The integrated copycat-like architecture composes outgoing messages in a flexible and distributed manner. VMattie's narrow-domain natural language understanding is pattern directed using another copycat-like architecture, rather than a classic symbolic parser. We are experimenting with building some emotions into the architecture (Bates, Loyall, and Reilly 1991, Sloman and Poli 1996), such as guilt at not getting an announcement out on time, confusion at not understanding a message, and anxiety at not knowing the speaker and title of an impending seminar. These replace the single temperature variable in the original copycat architecture. VMattie, in addition to being an autonomous agent, is very much a multiagent system with all internal actions taken by internal agents (codelets in the terminology of Hofstadter and Mitchell).

VMattie's Architecture

A high-level map of VMattie's architecture can be seen in Figure 1. The three center modules, Goals, Behaviors and Attention Registers, constitute a Maes behavior net with critical extensions. The Input Processing Knowledge and Input Processing Workspace modules, together with their associated codelets, implement an adaptation of a Copycat architecture, as do the Tracking Knowledge Base and the Composition Workspace modules. All computations within VMattie's architecture are carried out by codelets, as in the Copycat architecture. Each behavior is implemented by a collection of codelets.

VMattie has several distinct goals operating in parallel (goals in Maes' terminology—drives might be more descriptive). VMattie wants:

1) to get the weekly seminar out in a timely fashion,
2) to maintain complete information on each of the ongoing seminars,
3) to keep her mailing list updated,
4) to acknowledge each incoming message.

These goals vary in urgency as email messages arrive and as the time for the seminar announcement to be sent approaches. This variation in goal urgency, other than on and off, is an enhancement to the behavior net architecture. We are experimenting with goals being influenced by emotions, as mentioned above. Goals provide activation to behaviors that fulfill them.

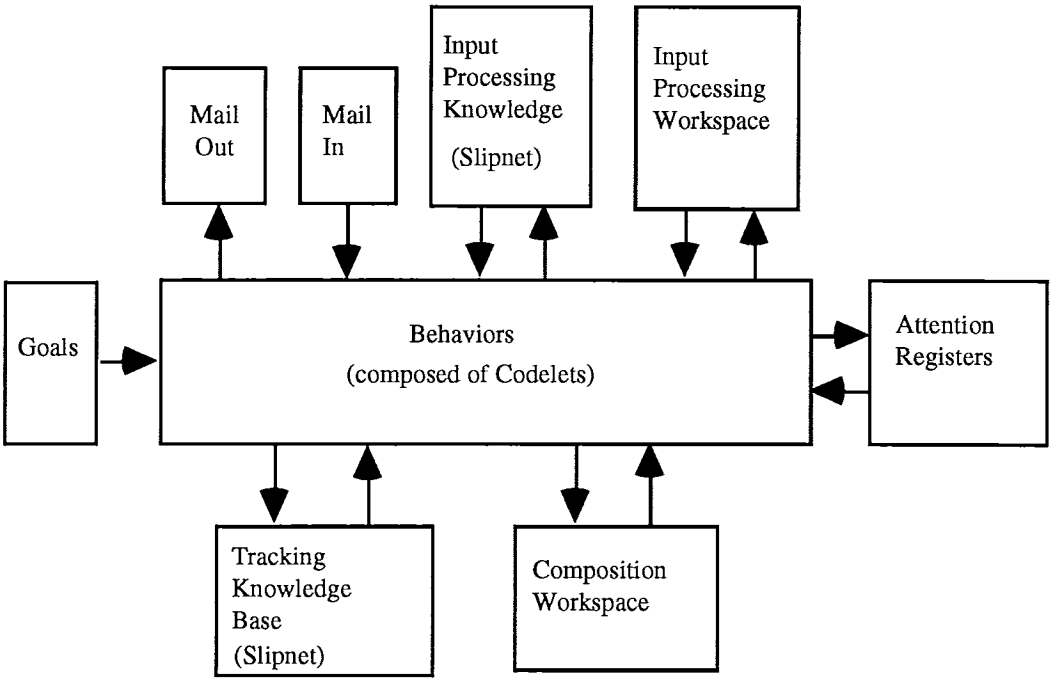

Figure 1.

Behaviors are typically mid-level actions, many depending on several codelets for their execution. Examples of behaviors might include 1) add-address-to-list, 2) associate-organizer-with-seminar, or compose-reminder (to organizer to send speaker, title, etc.). The behavior net is composed of behaviors and their various links, successor links, predecessor links, and conflictor links (see Maes 1990, or Franklin 1995, 245-251 for a brief account). Our behaviors must support variables. To associate-organizer-with-seminar immediately asks which organizer and which seminar. VMattie's behaviors implement the usual preconditions, action, add list and delete list allowing variables in the contents of any of these. Picture an underlying digraph composed of templates of behaviors and their links. Instantiated behaviors lie above their templates. Real links are instantiated above their templates if the values of their corresponding variables match. Activation spreads only through instantiated links.

The attention registers hold information extracted from an incoming email message. Acting like a structured blackboard, the attention registers make this information available to codelets that need it. Each register holds the content of a specified field. Fields include organizer-name, email-address, date, speaker, seminar-name, etc. These field names label the behavior variables discussed in the preceding paragraph. When occupied, attention registers provide environmental activation to behaviors that can use their contents.

An important attention register, different from the others, is the type-of-message register. Due to VMattie's narrow domain, there are only a small number of message types. Examples include requests to delete a name from the mailing list, and messages from organizers giving the title and speaker of a seminar session.

The input processing knowledge module plays something of the role of the slipnet in the Copycat architecture. It contains knowledge needed to understand incoming messages. For example, it knows the various forms of "tuesday" (tuesday, Tuesday, tues., Tues.), and the names of

buildings that typically house the seminars (Dunn Hall, Dunn, DH, Psychology, psychology, PSYC, Psyc, psyc). This module differs from slipnet in that it's not associative.

The information to be stored in this module was obtained from careful analysis of a corpus of messages sent to the real Mattie over a two year period. The types of messages were identified, abbreviations noted, and usages surrounding speaker, title and affiliation recorded. We also learned what sort of inferences were needed due to only partial information being sent in the incoming messages. Senders assume that Mattie will use default knowledge and context to decipher otherwise cryptic messages. The analysis of this corpus also contributed to the structure of the tracking knowledge base to be discussed later.

The input processing workspace corresponds to the Copycat architecture work space, but is much more complex. This workspace hold the contents of an incoming message while codelets work to understand it. Word or phrases deemed significant, that is contents of a field, are first tentatively labeled and later, if warranted, definitively marked. The input processing knowledge module is regularly consulted. Inferences are made. The single most important inference is to the type of message, which is the key to its understanding. When understanding takes place, that is, each significant word or phrase has been definitively labeled with a field name and the type of message inferred with assurance, This information is transferred to the attention registers by primitive codelets.

Primitive codelets are codelets that don't directly subserve any behavior, but that independently perform housekeeping functions within the systems. Some primitive codelets instantiate behaviors when their needed field values appear in attention registers. Others poll for incoming messages and transfer their contents to the input processing workspace.

The tracking knowledge base module contains information used in composing outgoing messages. For example, it stores default information on ongoing seminars, knowing that the Cognitive Science Seminar, organized by Art Graesser, meets on Wednesdays at 1:30 pm in PSYC 425. This default information is updated by behaviors. This module also tracks the current mailing list for announcements. This mailing list is also updated by behaviors. Importantly, this module also stores templates for the various types of outgoing messages.

As in the case of incoming messages, there are also only a small number of types of outgoing messages. VMattie sends out reminders to organizers asking for seminar information. She sends the weekly seminar announcement. Errata and addenda are also sent. And she send an acknowledgment for each incoming message. This acknowledgment contains VMattie's understanding of the incoming message in a form something like:

> "From your message of Friday, July 5th, I understand that Jeff Whitledge of the University of Memphis is to speak to the Computer Science Seminar on "Pandemat: A Pandemonium Based Artificial Life Agent" at 4:15 pm in Dunn Hall room 245."

The acknowledgment also explicitly invites correction.

Such a message, as every outgoing message, is composed in the composition workspace. Composition consists of choosing a template and filling in the fields appropriately. This is done by a simpler Copycat-like architecture, with information coming from the knowledge tracking module and from the attention registers. An ongoing copy of the current seminar announcement template is always present in the composition workspace. This copy is updated as new information arrives from the organizers. When the announcement is mailed, a new copy containing only default information replaces it.

VMattie is written in Java and Perl. It takes advantage of experience gained with a

previous softbot, Sumpy (Song, Franklin and Negatu,1996)

The CAAT Strategy

The following quote is from Rodney Brooks of MIT:

"No one talks about replicating the full gamut of human intelligence any more. Instead we see a retreat into specialized subproblems. Amongst the dreamers still in the field of AI (...) there is a feeling that one day all these pieces will all fall into place and we will see "truly" intelligent systems emerge."

This assertion is no longer as true as when it was written (Davidsson 1996, Johnson and Scanlon 1987, Moffat and Frijda 1995, Riegler 1994).

Virtual Mattie is the first project under the Cognitive Agent Architecture and Theory (CAAT) mantel. Cognition typically includes short and long term memory, categorizing and conceptualizing, reasoning, planning, problem solving, learning, creativity, etc. An autonomous agent capable of many or even most of these activities, including desires and perhaps emotions, will be referred to as a cognitive agent. [Sloman calls such agents "complete." (http://www.cs.bham.ac.uk/~axs/)]

Recently designed mechanisms for cognitive activities such as those mentioned above include the Maes and Hofstadter and Mitchell architectures as well as several others (Agre and Chapman 1987, Drescher 1991, Edelman 1989, Jackson 1987, Kanerva 1988, Wilson 1985, etc.). The CAAT strategy proposes to fuse sets of these mechanisms to form control structures for cognitive software agents. VMattie is the first exercise of this strategy.

Since the specification of every control architecture underlies some theory, this strategy also entails the creation of theories of cognition. Conversely, improvements in theory should lead to possible enhancements of the architecture. This synergy between theory and architecture should lead to productive experimental work in both artificial intelligence and cognitive science. VMattie is a start in this direction.

References:

Agre, Philip E. and Chapman, David (1987), "Pengi: An implementation of a theory of activity," *Proceedings of AAAI- 87,* 268-272.

Joseph Bates, A. Bryan Loyall, and W. Scott Reilly (1991). "Broad Agents," Proceedings of the AAAI Spring Symposium on Integrated Intelligent Architectures, Stanford University, March. These proceedings are available in SIGART Bulletin, Volume 2, Number 4, August 1992.

Davidsson, Paul (1996), Autonomous Agents and the Concept of Concepts, dissertation Lund University, Department of Computer Science. http://www.dna.lth.se/Research/AI/Papers/PhD.ps

Drescher, Gary L. (1991), **Made-up Minds**, Cambridge MA: MIT Press.

Edelman, Gerald M. (1989), **The Remembered Present: A Biological Theory of Consciousness**, New York: Basic Books.

Franklin, Stan (1995), **Artificial Minds**, Cambridge MA: The MIT Press.

Franklin, Stan and Graesser, Art (1997), "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents," Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag to appear.

Hayes-Roth, B. (1995), "An Architecture for Adaptive Intelligent Systems." Artificial Intelligence: Special Issue on Agents and Interactivity, 72:329-365, 1995.

Hofstadter, D. R. and Mitchell, M. (1994), "The Copycat Project: A model of mental fluidity and analogy-making." In Holyoak, K.J. & Barnden, J.A. (Eds.) **Advances in connectionist and neural computation theory**, Vol. 2: Analogical connections. Norwood, N.J.: Ablex.

Jackson, John V. (1987), "Idea for a Mind," *SIGGART Newsletter*, no. 181, July, 23-26.

Johnson, M., and Scanlon, R. (1987). "Experiences with a Feeling-Thinking Machine." *Proceedings of the IEEE First International Conference on Neural Networks*, San Diego. 71-77.

Kanerva, Pentti (1988), **Sparse Distributed Memory**, Cambridge MA: The MIT Press.

Maes, Pattie (1990), 'How to do the right thing', *Connection Science*, **1:3.**

Maes, Pattie (1992), "Learning Behavior Networks from Experience," **Toward a Practice of Autonomous Systems**, Cambridge MA: The MIT Press.

Mitchell, Melanie (1993), **Analogy-Making as Perception**, Cambridge MA: The MIT Press.

Moffat, David and Nico H. Frijda (1995), "Where there's a *Will* there's an Agent," in M. J. Wooldridge and N. R. Jennings, eds., **Intelligent Agents**, Berlin: Springer-Verlag

Riegler, Alexander (1994), "Constructivist Artificial Life," In: Hopf, J. (ed.) Proceedings of the 18th German Conference on Artificial Intelligence (KI-94) Workshop "Genetic Algorithms within the Framework of Evolutionary Computation" Max-Planck-Institute Report No. MPI-I-94-241.

Sloman, Aaron and Poli, Riccardo (1996). "SIM_AGENT: A toolkit for exploring agent designs in Intelligent Agents," Vol. II (ATAL-95), Eds. Mike Wooldridge, Joerg Mueller, Milind Tambe, Springer-Verlag pp. 392--407.

Song, Hongjun, Stan Franklin and Aregahegn Negatu (1996), "SUMPY: A Fuzzy Software Agent" in F. C. Harris Jr. ed., **Intelligent Systems**: Proceedings of the ISCA 5th International Conference, Raleigh NC: International Society for Computers and Their Applications - ISCA, 124-129.

Wilson, S. W. (1985), "Knowledge Growth in an Artificial Animal". *Proceedings of the First International Conference on Genetic Algorithms and Their Applications* (pp. 16-23). Hillsdale, New Jersey: Lawrence Erlbaum Associates.