Perfect Disruption
Causing the Paradigm Shift from Mental Agents to ORGs

Carl Hewitt
MIT EECS (emeritus)

The Mental Agent paradigm has had some success in modeling and simulating human-like behavior. However, computing has changed dramatically from the time of its invention and we are in the midst of a “perfect disruption” brought on by the following:

- **Hardware.** Many-core architecture that will soon support thousands of threads in a process for widely-used software applications using semantic integration (see below).
- **Software.** Client cloud computing, in which information is permanently stored in servers on the Internet and cached temporarily on clients that range from single chip sensors, handhelds, notebooks, desktops, and entertainment centers to huge data centers. (Even data centers are clients that often cache their information to guard against geographical disaster.) Client cloud computing will provide much needed new capabilities including the following:
  - maintaining the privacy of client information by storing it on servers encrypted so that it can be decrypted only by using the client’s private key. (The information is unencrypted only when cached on clients.)
  - providing greater integration of user information obtained from servers of competing vendors without requiring them to interact with each other.
  - providing better advertising relevance and targeting without exposing client privacy.
- **Applications.** Scalable semantic integration, e.g., integrating the following:
  - calendars and to do lists
  - email archives
  - presence information including physical, psychological and social
  - documents (including presentations, spread sheets, proposals, job applications, photos, videos, gift lists, memos, purchasing, contracts, articles, etc.)
  - contacts (including social graphs)
  - search results
  - marketing and advertising relevance influenced by the above

This paper explains how this perfect disruption is causing a paradigm shift from Mental Agents to ORGS (Organizations of Restricted Generality) as the foundation for implementing large-scale Internet applications.

How ORGs Address the Perfect Disruption

Organizations of restricted generality (ORGs) is our name for the paradigm in which organizations have people that are tightly integrated with information technology that enables them to function organizationally. ORGs formalize existing practices to provide a framework for addressing issues of authority, accountability, scalability, and robustness using methods that are analogous to human organizations. (Of course, as any reader of Dilbert comics knows, these organizational practices are imperfect.) Functional specialization within ORGs aids scalability by dividing up the workload and aids robustness by requiring specialized suborganizations to negotiate agreements according to their differing viewpoints and responsibilities.

ORGs are structured around organizational commitments defined as information pledged. For example, ORGs can use contracts to formalize their mutual commitments to fulfill specified obligations to each other. Yet, manifestations of information pledged will often be inconsistent. For example, any given contract might be internally inconsistent, or two contracts in force at one time could contradict each other. Issues that arise from such inconsistencies can be negotiated among ORGs.

ORGs are very well suited to many-core architecture. They can be implemented by actors—the universal primitives of concurrent computation. ORGs are message composable meaning that systems are composed by sending messages. Contrast that with update composability in which systems are composed by updating a transactional
memory.\textsuperscript{6,7} Message composability has numerous advantageous over update composability, including modularity, scalability, and the fact that message composability works across the entire computational spectrum from within many-core chips to geographically distributed systems, whereas transactional memory works only within the address space of a single process on a single computer.

ORGs provide guidance on how to extend the X86 architecture (AMD, Intel, Via, etc.) to make it more suitable for many-core to provide:

- concurrent nonstop automatic storage reclamation (garbage collection) and relocation to improve efficiency,
- prevention of memory corruption that otherwise results from programming languages like C and C++ using thousands of threads in a process,
- nonstop migration of ORGs (while they are in operation) within a computer and between distributed computers

ActorScript\textsuperscript{5} is a kind of Actor programming language that’s well suited for implementing message-composable ORGs. Distributed ORG implementations can accommodate client cloud computing by maintaining some parts on the clients and some on the servers. Scalable semantic integration is also well suited for ORG-based implementation because ORGs supply a framework for implementing two critical components:

- a scalable semantic engine provides inference and action capabilities for semantic information, and
- an intuitive semantic user interface based on ontologically-extended natural language provides visualization and interactive capabilities for semantic information.

ORGs have important advantages over mental agents as described in the next sections.

The Dream of Mental Agents

A Mental Agent is defined as cognitively operating in human-like fashion. The paradigm is deeply and pervasively psychological.\textsuperscript{8,9} The most popular kind of mental agent can be characterized as BDIA: beliefs, desires (goals), intentions (plans), and affect (emotions). It has moved beyond its original sequential conceptualization by introducing parallelism, which can be used for low-level input–output (vision, for example), (associative) memory operations, and other basic operations as performed by the parts of the brain. Yet, none of these changes the mental agent paradigm, which draws its fundamental strength from staying close to the mental operations of a single person.

The development of mental agents has continued steadily since the earliest days of artificial intelligence, and researchers have realized impressive achievements (see the Summer 2006 issue of AI Magazine for some examples). Yet, progress in using mental agents as a foundation for software applications has been frustratingly slow. Impressive demonstrations of mental agents’ capabilities in some application areas have repeatedly failed to garner widespread commercial adoption. Nevertheless, researchers continue the quest to develop mental agent frameworks for software systems. Expressions of confidence and hopes for the future have long been regular features of conferences.\textsuperscript{10} Researchers have been heartened because no convincing principled arguments have proven it impossible; indeed, human behavior presents a kind of existence proof that something like the mental agent paradigm can be made to work. Moreover, the community has evolved and gained insights into multiagent systems.

In contrast to the situation with ORGs, the perfect disruption is causing mental agents to lose ground. First, many-core architectures pose a challenge because the information processing of a computer is no longer at all like the information processing of a person:

- Using human-like mental operations becomes an increasing bottleneck as the number of cores increase because the cores perform independent tasks.
- Using a human-like input/output system becomes an increasing bottleneck as the number of interconnections increase because the wires carry independent messages.

Scalable semantic integration requires capabilities beyond those of a mental agent. The current family of standards (e.g. OWL 2\textsuperscript{11}) represent significant progress. However, the semantics of OWL 2 need improvement:

- OWL 2 semantics require that an ontology (theory) must be absolutely consistent. A single inconsistency can allow anything and everything to be inferred, e.g., “The moon is made of green cheese,” whereas adopting paraconsistency would make it possible to reason safely in the presence of inconsistent information.\textsuperscript{5}
• OWL 2 semantics support only a single ontology (theory). However, ORGs require the simultaneous use of multiple inter-related ontologies (theories).\(^3\)
• OWL 2 semantics do not support reification reflection\(^5\) that is needed to relate natural language input/output to reasoning.
• Furthermore, OWL 2 semantics are rigid and inexpressive because they do not support ontologically-extended natural language.

Fortunately, the above limitations to the W3C specifications can be overcome in a way that substantially preserves the value of work already done using the current W3C specifications so that it doesn't have to be completely redone.

The mental agent paradigm might increasingly be used in avatars (both human-like and animal-like) and cognitive models\(^12\) of individual humans, but operational implementations will require ORGs, just as all large software systems will. The original conception of the development of mental agents is thus turned upside down: instead of ORGs being implemented using mental agents, humans will be simulated and avatars will be implemented using ORGs!

The Software Agent Conundrum

According to a published consensus of researchers, a software agent is basically a mental agent adapted for software engineering.\(^13\) More general conceptions have been attempted without success. As Charles Petrie noted, for example, “some have tried to offer the general definition of agents as someone or something that acts on one’s behalf, but that seems to cover all of computers and software.”\(^14\)

Despite many years of trying, none of the software agent development systems for large-scale Internet applications have had any significant commercial success. ORGs (although currently not called such by practitioners) are trumping mental agents for implementing large-scale Internet systems. No software agent architectures can compete with ORGs in understandability, manageability, and scalability. As Petrie predicted,\(^15\) the old agent technology has essentially disappeared from large-scale software systems. Consequently, a conundrum is emerging, such that researchers must choose whether to

• stay the current mental agents course despite the paradigm switch from mental agents to ORGs, or
• change course and adopt the ORGs paradigm as fundamental, thus begging the question, “Where are the agents?”

Many artificial intelligence researchers have long presupposed that agents are a principal subject of their field. This is especially poignant for the autonomous agents and multi-agent systems (AAMAS) community, which includes the term “agent” in its name twice. AAMAS is a vibrant community whose members are performing exciting and important research, but its conceptual foundations are badly in need of reformulation because of the paradigm shift from mental agents to ORGs.

Upside of the Perfect Disruption

According to Steve Balmer in a recent appearance at the Churchill Club, the “perfect disruption” represents a “perfect opportunity” enabling fundamental new capabilities in the computer industry. This author believes that Ballmer is correct in his assessment. As a result, IT will become

• \textit{Easier to use} – civilized discourse when interacting with people
• \textit{More useful} – semantic integration of information from diverse sources
• \textit{Privacy friendly} – client integration of information

Consequently IT will become more valuable and as a result:

• IT employment will increase with new employment opportunities:
  o Ontologists
  o ORGs technologists
  o Discourse experts
• IT revenue will go up
In the midst of the current downturn, we can look forward to a more prosperous future.

Acknowledgments
Charles Petrie suggested the phrase “Perfect Disruption” and made other valuable suggestions for improvement. Jean-Pierre Briot, Jeremy Forth, Michael Genesereth, Fanya S. Montalvo and members of the Stanford Logic Group provided helpful discussion and comments.

References

Carl Hewitt is emeritus at Massachusetts Institute of Technology. His research interests include logic programming, concurrency, strongly paraconsistent logic, and the philosophy and sociology of science and engineering. Hewitt has a PhD in mathematics from MIT. Hewitt was the IBM Chair Visiting Professor in the Department of Computer Science at Keio University in Japan. Contact him via www.carlhewitt.info.