AI Magazine Volume 27 Number 4 (2006) (© AAAI)

AI Meets Web 2.0

Building the Web of Tomorrow, Today

Jay M. Tenenbaum

Illustrations by Kevin Hughes

This article is derived from the 2005 Innovative Applications of Artificial Intelligence conference invited talk "AI Meets Web 2.0: Building the Web of Tomorrow, Today" presented by Jay M. Tenenbaum in Pittsburgh, Pennsylvania. I invited Marty to deliver that talk at IAAI-05 because he is a true AI and web visionary. It is common in times of grade inflation and hyperbole that speakers and authors are introduced as "visionaries." Jay M. Tenenbaum (or Marty, as he is known) actually is a world-renowned Internet commerce pioneer and visionary. Mary envisioned the commercial and industrial use of the Internet more than a decade before it became a reality, at a time when only academics, industrial R&D groups, and government labs had access to it. At the time, Marty's e-commerce vision evoked all kinds of resistance and objections about why it couldn't or shouldn't be done.

In 1991, Marty founded and became the chief executive officer of Enterprise Integration Technologies, the first company to conduct a commercial Internet transaction (1992), a secure web transaction (1993), and an Internet auction (1993). In 1994, Marty formed the CommerceNet industry association and jump-started the Internet marketplace by convincing 50 leading corporations to get online. In 1997, he cofounded Veo Systems, which pioneered the use of XML documents for automating business-to-business transactions. In 1999, Commerce One acquired Veo Systems, and Marty ultimately served as its chief scientist. Marty now splits his time between Internet startups Efficient Finance, Medstory, and Patients Like Me, and the nonprofit CommerceNet.

This article points to the technical opportunity to meet Allen Newell's criteria for intelligent systems by incrementally assembling complex web services applications out of component web services, tagged data, and inference techniques. It also articulates an exciting set of business opportunities to deliver on the economic promise of e-commerce and AI.

– Neil Jacobstein,
Chair, IAAI-05

■ Imagine an Internet-scale knowledge system where people and intelligent agents can collaborate on solving complex problems in business, engineering, science, medicine, and other endeavors. Its resources include semantically tagged websites, wikis, and blogs, as well as social networks, vertical search engines, and a vast array of web services from business processes to AI planners and domain models. Research prototypes of decentralized knowledge systems have been demonstrated for years, but now, thanks to the web and Moore's law, they appear ready for prime time. This article introduces the architectural concepts for incrementally growing an Internet-scale knowledge system and illustrates them with scenarios drawn from e-commerce, e-science, and e-life.

In this article, I want to share a vision of how to build or, more precisely, grow Internet-scale knowledge systems. Such systems enable large numbers of human and computer agents to collaborate on solving complex problems in engineering, science, and business or simply managing the complexities of life (say planning a trip or an event). It's a vision that's been evolving over 20 years since my days as

Copyright © 2006, American Association for Artificial Intelligence. All rights reserved. ISSN 0738-4602

- Exhibit adaptive goal-oriented behavior
- Learn from experience
- Use vast amounts of knowledge
- Exhibit self-awareness
- Interact with humans using language and speech
- Tolerate error and ambiguity in communication
- Respond in real time

Table 1. Newell's Criteria for Intelligent Systems.

Travel	What's the best way to get to Pittsburgh?
Meetings	Who should I have dinner with tonight? Where?
Supply chain	How can I get 100 PCs delivered by tomorrow?
Medicine	What drugs might be effective against this new bug?
E-business	Where can I get the best deal on a car and a loan?

Table 2. Classic AI Problems.

an AI researcher and, more recently, as an Internet entrepreneur. Thanks to the explosive growth of the web, it's a vision whose time has come. I also have a larger goal: to bridge the AI and web communities, which have so much to give to and learn from each other.

Twenty-five years ago, at the birth of AAAI, Allan Newell articulated a set of criteria that a system had to exhibit to be considered intelligent (see table 1). Newell was very explicit that an intelligent system had to exhibit all of these criteria. This requirement reflected the then prevailing view that intelligent systems were monolithic and were developed centrally by an individual or small group.

The web has shown us a different path to intelligence—millions of simple knowledge services, developed collaboratively in a decentralized way by many individuals and groups, all building on each other, and demonstrating a collective form of intelligence. It's time to reconcile these two views and create a new breed of hybrid knowledge systems that combine the best elements of AI and the web, and of humans and machines. Such hybrid systems can solve heretofore intractable real-world problems and potentially demonstrate heretofore unattainable levels of machine intelligence.

To give a hint of where I'm headed, I'll revisit some classic AI problem-solving tasks, which are as timely now as they were in Newell's day (table 2). I've worked on many of them, and I'm sure readers have too. How would you as an individual approach any of these tasks today? Undoubtedly, by searching the web, which provides thousands of relevant information sources and services.

If I were building an AI system to solve these problems, I too would start with the web, gradually adding a little AI here and there to automate time-consuming functions and integrate processes. Take travel, for example. I might start by creating a simple agent to check many travel sites for available flights and fares. A second agent might use these results to select an optimal itinerary based on my personal preferences, frequent flyer accounts, and calendar schedule.

Now imagine giving millions of people this ability to create agents that incrementally automate tasks, as well as the ability to publish those agents as services that others can use and build upon. Before long, interesting forms of collective intelligence will begin to emerge.

Approaching AI applications in this way has manifest benefits—the ability to leverage billions of dollars of web infrastructure, millions of web services, and millions of knowledge engineers all building on each other's work. Moreover, such systems are useful from day one, because they start with the web. Automation proceeds incrementally—one person, and one task at a time. Each step provides immediate, incremental benefit. And because people remain in the loop, such systems can fail gracefully, by handing off hard or unusual cases for manual processing.

I'll start this article by reviewing the evolution of the vision from its roots in AI research through the early days of the web and more recent developments such as web services and Web 2.0 (referring to the spate of recent web innovations that includes blogs, wikis, social networks, and the like). These recent developments make the web more programmable and more participatory, important attributes for Internet-scale knowledge systems. I'll review Web 2.0 technologies and methodologies from an AI perspective and then illustrate, through

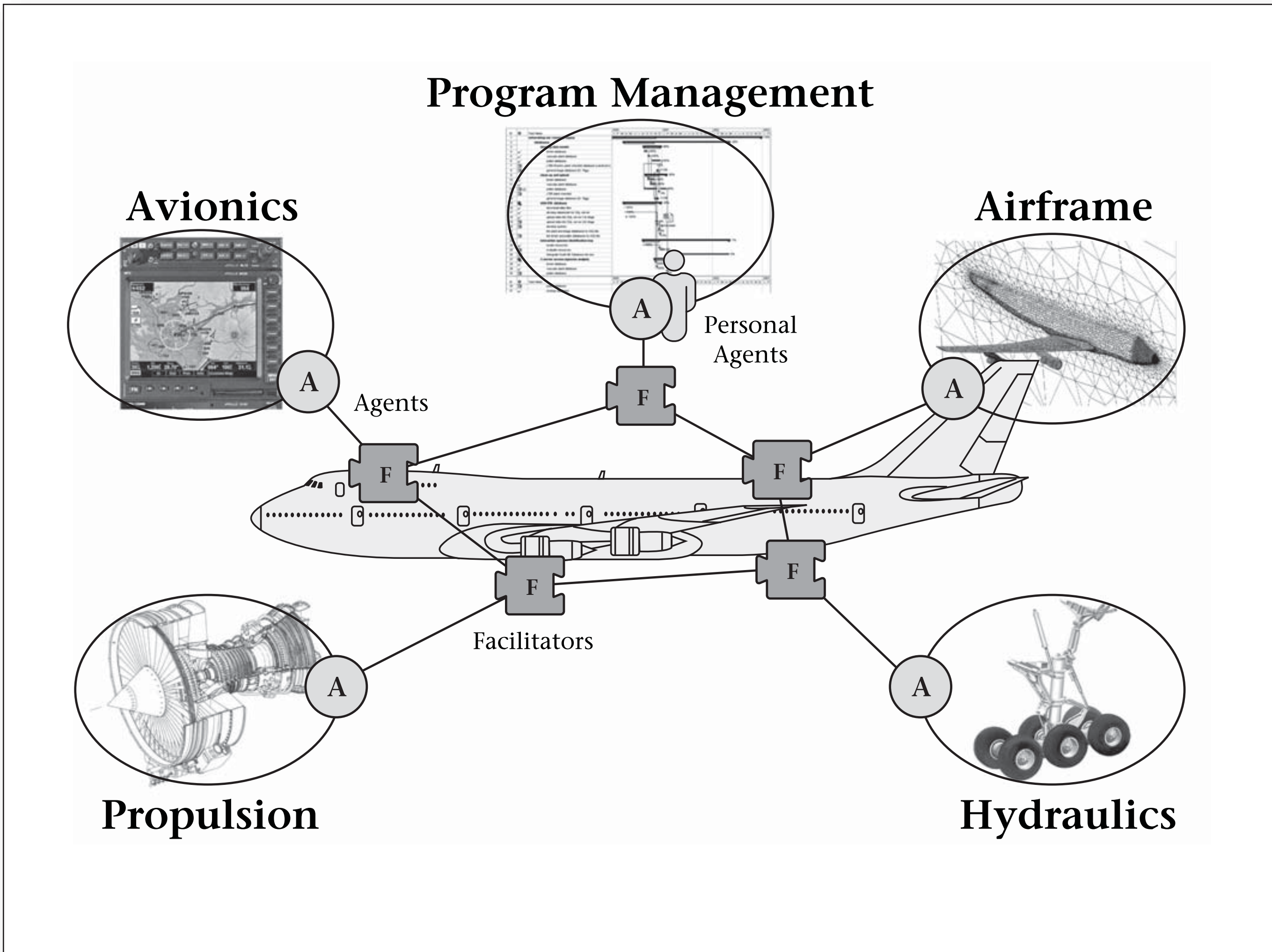

Figure 1. Concurrent Engineering and the PACT Architecture

case studies and scenarios, some knowledge systems that synthesize Web 2.0 and classic AI approaches. Finally, I'll conclude with a call for action for realizing the vision on an Internetwide scale.

Evolution

Twenty years ago, I was running a research lab for Schlumberger. At the time, Schlumberger was trying to break into the CAD/CAM business through its acquisition of Applicon. I was fascinated by the possibility of using the Internet to support large-scale engineering projects, such as the design of a new airplane, involving thousands of people at hundreds of companies (figure 1).

Inspired by distributed blackboard systems such as Hearsay, we modeled the process as human and computer agents collaborating through a shared knowledge base, representing a model of the artifact. When an agent modified the design, affected agents were notified so they could critique the change or respond with changes of their own. For example, a hydraulics engineer installing some fluid lines might alert an airframe agent to check whether those lines interfered with the movement of any control surface. Although a centralized shared knowledge base is depicted, the model can be distributed in practice to facilitate scaling. Each agent maintains aspects of the model most relevant to it in local CAD systems and provides that information to other agents that need it.

The Palo Alto Collaborative Testbed (PACT)

We built several research prototypes of such agent-based collaborative engineering systems

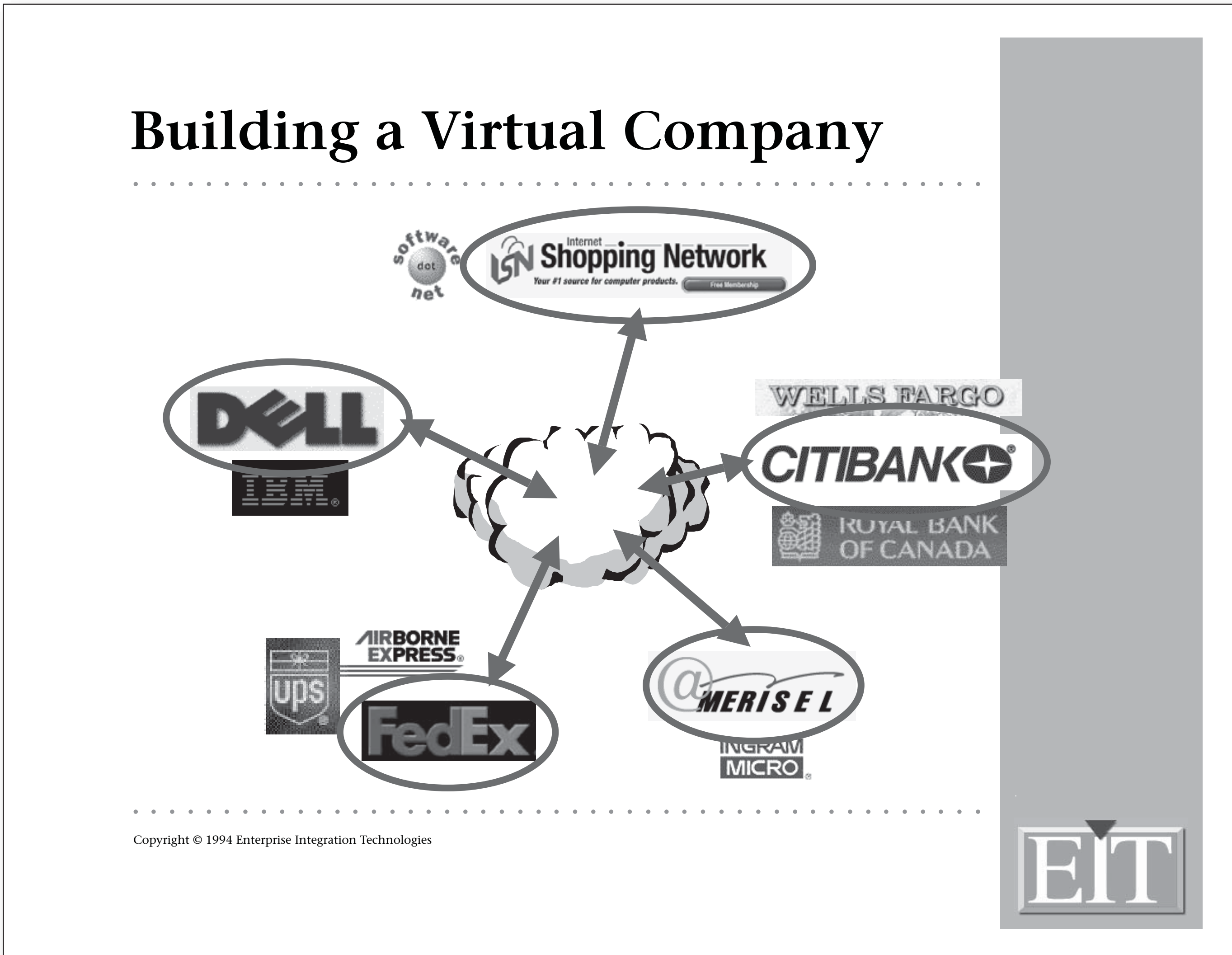

Figure 2. The Virtual Company circa 1994.

during the late 1980s. The most impressive was the Palo Alto Collaborative Testbed (PACT), which was documented in the January 1992 issue of *IEEE Computer*. PACT used the Internet to link four independently developed knowledge-based design systems (covering requirements management, kinematics, dynamics, and electronics) at three different organizations—EIT, Lockheed, and Stanford. The systems then collaborated on the design of a small robotic manipulator.

The PACT architecture consisted of three components, as depicted in figure 1. First, agents encapsulated existing design systems, transforming them into knowledge services. Second, facilitators handled interagent communication, routing messages on the basis of content to potentially interested agents. Facilitators also filtered, prioritized, and semantically mapped messages so that agents could focus on design-related tasks. Finally, personal agents enabled people to participate in this agent ecology. Users could respond directly to messages from other agents using a graphical user interface (GUI); they could also specify pattern-action rules that generated automated responses to routine messages.

In retrospect, PACT was a seminal paper, articulating three key principles for building large distributed knowledge systems. Paraphrasing from the conclusion, the first principle states that "Instead of literally integrating code, modules should be encapsulated by agents, and then invoked when needed as network services." This is the core idea behind today's service-oriented architectures. The second principle holds that agents should communicate on a knowledge level, using the semantics of the problem domain, rather than the local standards employed by encapsulated systems. Finally, messages should be routed intelligently at run time based on their content, since it's often impossible to know a priori which agents might be affected by a design action.

Electronic Commerce

By 1990, I had shifted my focus to Internet commerce. The company I founded, Enterprise Integration Technologies (EIT), developed some of the earliest shopping sites including Internet Shopping Network (ISN), launched in April 1994. ISN pioneered many innovations including online catalogs, shopping carts, and cookies and served as a role model for Amazon and other shopping sites. PACT's notion of integrating services also figured prominently. As depicted in figure 2, we envisioned ISN as a virtual company, outsourcing fulfillment services to Merisel, shipping services to FedEx, and payment services to Citibank. We even envisioned integrating a build-to-order supplier like Dell and creating a virtual computer company.

From an AI perspective, we thought of these services as special=purpose problem solvers. FedEx, for example, with more than 100,000 employees and thousands of trucks and planes was a special-purpose problem solver for logistics, invoked through a simple message-based application programming interface (API)—"Get this package to New York City by tomorrow morning." It struck us as foolish to write a logistics planner rather than leverage such a powerful off-the-shelf resource.

Web Services

Now let us fast-forward to today's world of web services. Companies in competitive industries like electronics are beginning to publish business services (for example, place an order, make a payment, track a shipment) that others can discover in registries and integrate into their own processes, then publish new services that others can build on. With critical mass, business services are expected to grow explosively like the web itself.

Prior to web services, companies that wanted to integrate their business processes had to resort to expensive custom integration or electronic data interchange (EDI) solutions. Such projects typically cost hundreds of thousands of dollars and took months to implement. They made sense for large trading partners like an IBM or Ingram who do a billion dollars a year of business together. However, they did nothing for the tens of thousands of small and medium-size companies that compose the bulk of the electronics industry value chain.

With web services, any company can publish on its web server product descriptions, prices and availabilities, shipping and production schedules, and so on in a form that both people and machines can understand. With such information, it suddenly becomes feasible to optimize forecasting, production planning, inventory management, and other enterprise resource planning (ERP)-like functions across an entire industry rather than just within a single enterprise.

At least that's the vision. Unfortunately, in practice, there are three problems with web services that stand in the way of doing business.

First is the problem of complex standards. Web services promised lightweight integration, using simple, open protocols inspired by the web. Things started out simply enough with SOAP, WSDL, and UDDI. However, they rapidly got out of hand as people began adding features that pushed web services toward a full-fledged CORBA-like distributed operating system. Today there are literally dozens of standards supporting features such as business process orchestration, security, and service provisioning and management. Additionally, there are hundreds of specialized XML languages that define the semantics for doing business in particular industries. Each such language typically entails a painful multiyear standards process whereby companies attempt to reconcile their local data and business process standards. The electronics industry has been attempting to standardize on RosettaNet, which is sponsored by leading manufacturers and distributors. However, after more than five years, only a few RosettaNet processes have moved past initial pilots.

Second is the problem of static processes. Business demands agility, but business process execution language (BPEL), the emerging web services standard for business process orchestration, was designed for static processes. At the least, a rules engine is needed to select the right BPEL script at run time. Even better, rules should be embedded within BPEL so that actions can respond dynamically to events.

Third is the problem of point-to-point communications. Standard web services rely on point-to-point communications. However, as we've seen, it's often not clear until run time who needs to know about a design change, or which supplier has the best prices and inventory on hand to fulfill a request.

We can overcome these problems by combining the best features of PACT and web serv-

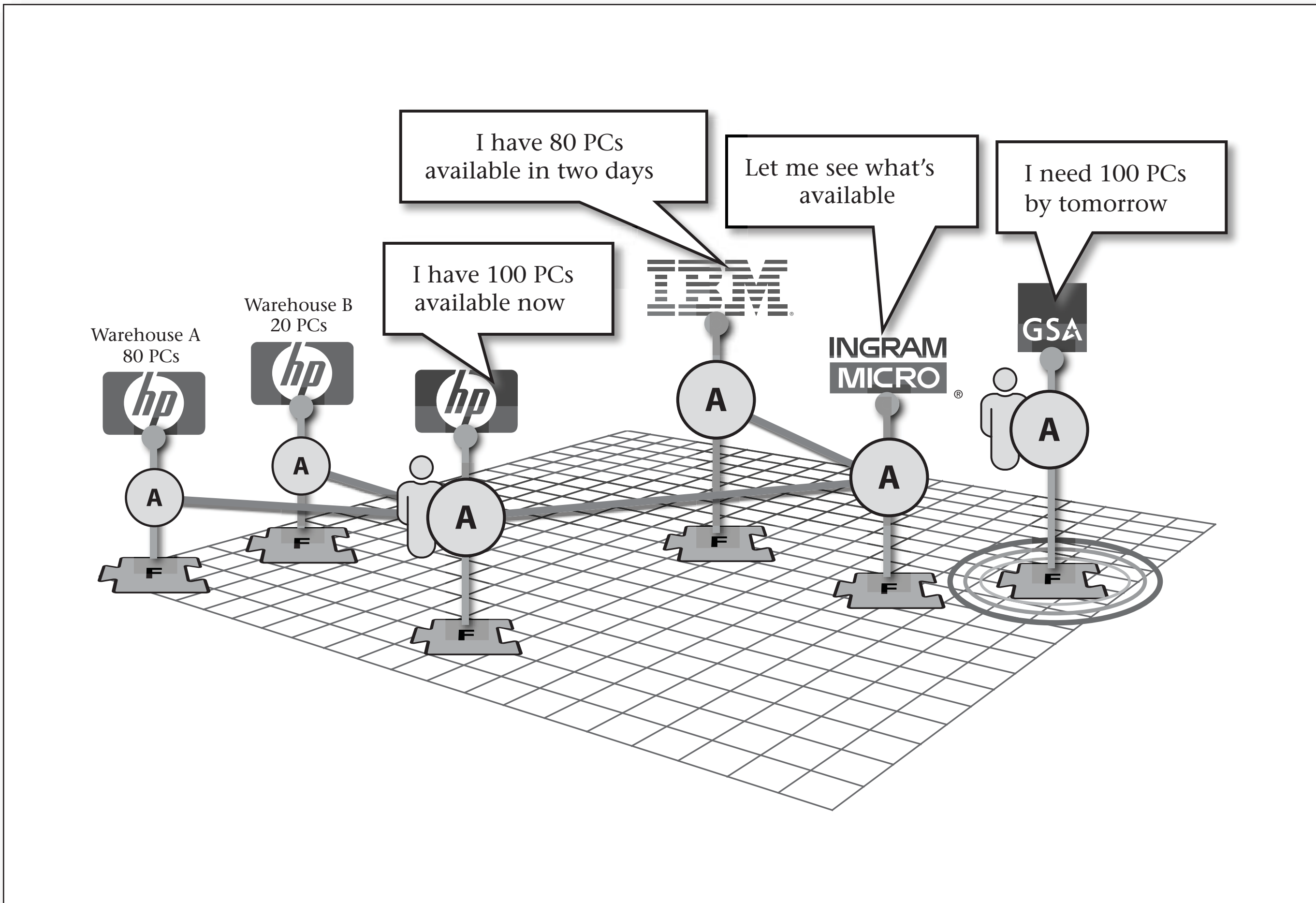

Figure 3. Knowledge Services at Work.

ices. Agents wrap web services, transforming them into rule-based "knowledge services." Facilitators enable agents to communicate on a knowledge level; they route messages on the basis of their content and translate between local data standards and messaging protocols. Finally, personal agents enable humans to participate in automated processes.

Figure 3 illustrates how such a solution might work in practice. A purchasing agent at the General Services Administration (GSA) broadcasts a solicitation for a hundred PCs needed by tomorrow. Ingram, a leading distributor, has an agent that subscribes to such solicitations. The agent picks up the request for quotation (RFQ) and forwards it to interested suppliers, whose agents respond with current inventory levels. In the case of one supplier (HP), that involves polling and aggregating inventories from several warehouses. Ingram's agent informs the GSA agent that it can fulfill the request. The order is consummated through Jane, HP's account rep for GSA; her agent automatically approves the order because it falls within GSA's contractual guidelines and credit limit.

Aggressive Interoperability

The knowledge services architecture relies on two important features. The first is aggressive interoperation (or "A.I." for short). Aggressive interoperation means never letting standards, or the lack thereof, get in the way of doing business. If Ingram and HP use different data formats or messaging protocols, a translation service can map them; if there's one thing the AI community is good at, it's mapping between two structured representations. The translation can be performed by the message sender, the recipient, or some third-party broker—whoever has the economic incentive. Thus, if HP is eager to attract new customers, it could accept orders in any major format and translate them into its internal standard. One enduring lesson

Participatory (P2P)	Blogs, wikis, social networking, RSS feeds, read/write
Semantic	Tags, microformats, semantic web, vertical search
Real time	Instant messaging, events (publish/subscribe)
Pervasive	Billions of edge devices with substantial computing and broadband access – telephones, cars, RFID readers . . . .
Community	Simplicity, rapidity, mass collaboration, empowerment

Table 3. Web 2.0 Technologies and Methodologies.

of the web is that if a service is economically viable, someone will step forward to provide it.

Incremental Automation

The second feature is incremental automation. Let's assume that every step of this cross-organizational purchasing process is initially performed manually, using e-mail, web browsers, and phones. One day, Joe in GSA's stock room, decides he's got better things to do than manually checking a website to determine whether there are sufficient PCs on hand. He scripts his personal agent to check the inventory level of PCs every morning and, if there are fewer than 100 on hand, to e-mail a purchase request (or fill out a web-based order form) for approval by his supervisor Bill. Joe's still there to handle the rest of his job as well as unexpected events concerning stocking PCs, but now he's got one less rote task to do.

Joe assumes his agent's request will continue to be processed manually by Bill. However, at some point, Bill gets smart and has his personal agent automatically approve routine requests that are within budget and pass them along to John in purchasing. In similar fashion, John might have his personal agent pass along routine purchase orders to approved suppliers like Ingram. At the supplier, an order entry clerk might task her agent to check whether the order can be fulfilled from stock on hand and, if not, to pass it along to the manufacturer (HP) for drop-shipping to the customer (GSA).

Imagine tens of thousands of individuals at thousands of organizations in a supply chain, each independently and for their own reasons, deciding to automate some routine parts of their jobs. The result is a form of emergent intelligence, where complex processes are automated not through an explicit design process, but rather through the independent actions of many, one person and one task at a time. This instance of mass collaboration is an appropriate segue to our tour of Web 2.0.

Web 2.0 Tour

The term *Web 2.0* refers to a collection of emerging web technologies and methodologies (table 3) that make the web more *participatory* (that is, two-way versus read only), more *semantic,* and more *real time* (that is, event driven). Perhaps most importantly, Web 2.0 is a cultural phenomenon. Developers start with a simple but useful idea and get it out quickly so others can refine and embellish it. The process has come to be known as mass collaboration—thousands of individuals, building incrementally upon each other's work.

I've italicized the Web 2.0 concepts that are most important to realizing my vision, and will give a brief tour of these concepts, both as building blocks for Internet-scale knowledge systems and as AI research opportunities for enhancing their functionality. I'll organize the tour around four Web 2.0 themes from table 3, beginning with making the web more participatory. The referenced websites provide more details.

Making the Web More Participatory

Our first Web 2.0 theme deals with transforming the web into a more participatory and personalized medium through blogs, syndication, wikis, and the like.

Blogs. Blogs, the first major Web 2.0 phenomenon, are shared online journals where people

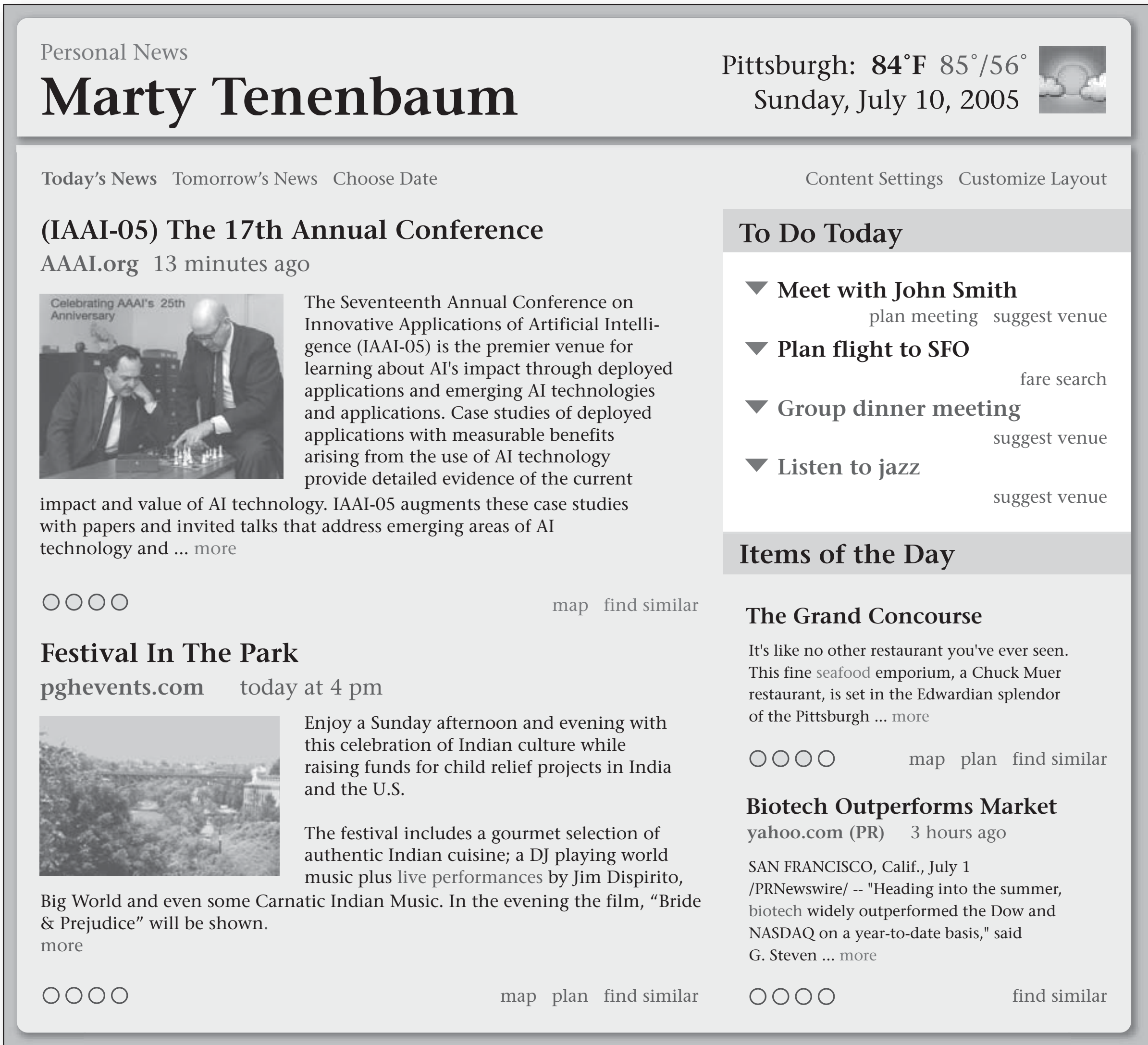

Figure 4. The Learning Browser.

can post diary entries about their personal experiences and hobbies. Thanks to user-friendly editing tools like Movable Type[1] and services like Blogger,[2] there are now over 12 million blogs, with nine new ones created every minute (according to a recent estimate by *Wired Magazine*). While many blogs are admittedly full of drivel, some contain real-time information and insights that would be difficult to find anywhere else. Those in business or professionals ignore blogs at their peril. Those in AI and looking for a quick way to get rich should figure out how to mine this trove of information and use the results to predict markets.

In a world of 12 million plus blogs, there are two big challenges: for bloggers, it's getting noticed; for blog readers, it's avoiding information overload. An individual with a browser can at best regularly track a few of dozen sites for updates.

Feeds. Syndication feeds, such as RSS and Atom, address both problems by defining an XML format for metadata about new content added to a website—information such as the title of a posting, its URL, and a short description. Using a feed reader such as NetNewsWire, individuals can periodically ping the feeds of favorite sites to determine quickly whether anything has changed since their last visit. Site owners can also submit their updates to search engines and aggregators. Aggregators subscribe to numerous feeds and sort postings into subject-specific channels such as "Science News" or "Travel." By subscribing to syndicated feeds, an individual can effectively track hundreds or even thousands of sites, qualitatively transforming the web surfing experience.

RSS and Atom may disappoint AI folks with their limited semantic structure and content. In the example above, an aggregator might provide only the most basic information elements, such as the name of the channel ("Yahoo! News: Science News"), a headline ("NASA Sets July 13 Shuttle Launch Date"), and a brief natural language summary ("NASA plans to blast into space on July 13 after more than two years on the ground…"). The AI community can significantly improve the ability to find and filter web content by enriching the semantics of syndication feeds and adding inference rules to "feed-reader" software. I'll identify some specific opportunities later in this article.

Wikis. Wikis are websites where content is developed and edited collaboratively. The most impressive example is Wikipedia,[3] a wiki-based encyclopedia with more than 600,000 articles on every conceivable topic, contributed and collaboratively edited by more than 300,000 people. Wikipedia provides interesting lessons for the AI community on developing and maintaining collaborative ontologies. For example, as of last July, the top-level subject headings for geography included Antarctica, landforms, and villages. You may not agree that these categories deserve equal billing. The good news is that with a wiki, if you don't like how things are organized, edit it. If you disagree with someone's edit, open a discussion.

Imagine the power of combining these participatory technologies with the power of AI. Figure 4 is a mockup of a "Learning Browser" being developed with support from CommerceNet. Unlike traditional aggregators that summarize new feeds by listing headlines or subject lines, it presents the user with a personalized "zine." In the spirit of the Media Lab's "Daily Me," the zine is a digest of the latest feeds relevant to the task at hand—whether it's researching a report, following a sport, or in the example shown, planning my trip to Pittsburgh for AAAI-05. Over time, the browser can refine its model of my interests and modify what's presented, based on implicit feedback such as what I click and what I don't, how long I spend on an article, what I recommend to my friends, what they recommend to me, as well as what's happening in the world. Many sites can benefit from this kind of implicit feedback, from search engines to dating services.

Semantics

Our second Web 2.0 theme—semantics—includes tags, microformats, and vertical search.

Tagging and Folksonomies. Tags are personalized labels for describing web content—web pages, blogs, news stories, photos, and the like. One person might use the terms *Beijing* and *Buildings* to describe a photo of downtown Beijing. Another might prefer the tags China and Vacation to describe the same photo. Collectively, the set of tags adopted by a community to facilitate the sharing of content is known as a folksonomy.

Folksonomies organize knowledge differently than traditional classification hierarchies. While tags often correspond to traditional subject headings or categories, they can also represent more subjective descriptors: "cool," "recommended," "reading list," and so forth. Moreover, folksonomies are flat: categories can overlap and items can be classified in multiple ways. Instead of force-fitting an item into a traditional classification hierarchy like the Dewey decimal system, just affix all the tags that apply. Most people find this easier and more intuitive than agonizing over whether a picture shown should be classified under China, buildings, or vacation. The most defining characteristic of folksonomies is that they evolve through mass collaboration within a community rather than through the dedicated efforts of one (or a few) expert designers. Unlike most ontologies developed by AI researchers, folksonomies are always widely used—at least within the community that evolved them.

Some of the best examples of collaborative tagging can be found at Flickr,[4] a photo-sharing site recently purchased by Yahoo! Most of the photos on Flickr have been tagged by multiple community members. Useful tags stick. Popular tags get reused, and thus reinforced, because people want their content to be found. Frequently co-occurring tags, such as China and Beijing, provide an opportunity to infer relationships (causal, structural, geographical, and so on).

The Del.icio.us website[5] pioneered collaborative tagging. Del.icio.us lets users assign arbitrary tags to a web page to help them remember it (for example, pages about China, cool places, hot night spots). Del.icio.us also lets users see how others tagged the page, which may influence their own choice of tags. They can click an existing tag and see all other pages sharing that tag. Additionally, they can drill down and see who affixed a particular tag, other pages they labeled with that tag, as well as any frequently co-occurring related tags (for example, San Francisco and California), and

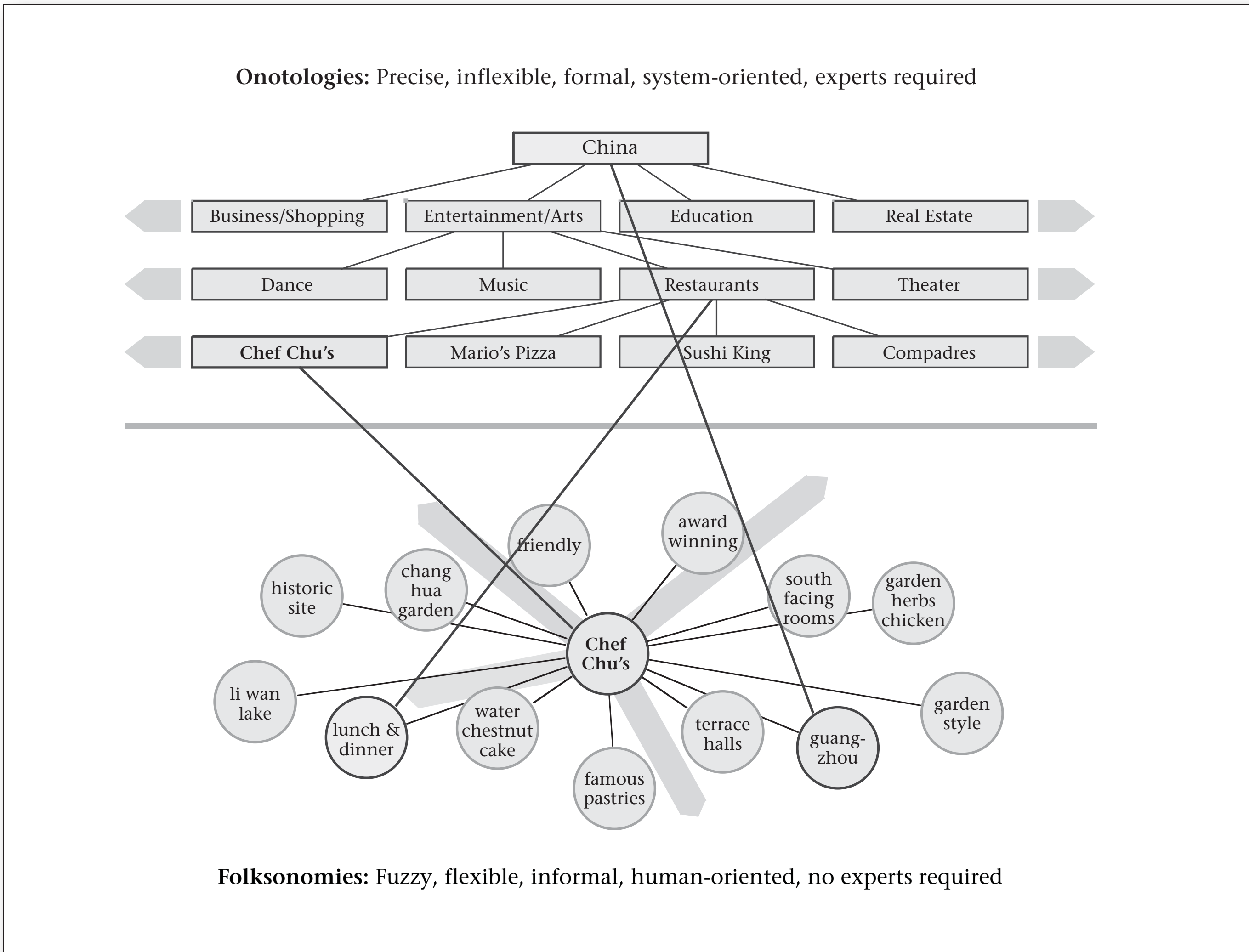

Figure 5. Ontologies and Folksonomies.

the pages associated with them. They can look generically for cool or recommended pages or for pages a specific individual thinks are cool or worth recommending.

The wealth of folksonomy data available from collaborative tagging sites like Flickr and Del.icio.us creates some tantalizing AI research opportunities. One possibility: deriving formal ontologies automatically from raw folksonomies. Imagine taking the collected tags assigned to thousands of restaurant sites in China (figure 5) and inferring, for example, that Chef Chu's is a restaurant, that restaurants serve lunch and dinner, that Guang-Zhou is a city in China, and so forth. If deriving ontologies automatically proves too hard, a valuable fallback would be to map folksonomies onto existing ontologies. Such a mapping would impose a hierarchical structure on tags, enabling helpful inferences. Knowing that Guang-Zhou is in China, for instance, one could infer that Chef Chu's was in China, even though "China" did not appear explicitly as a tag for Chef Chu's web page. A little knowledge can go a long way in increasing the usefulness of tags. Conversely, a lot of tags can go a long way in increasing the usefulness of knowledge.

Microformats. Microformats are simple data schemas for describing people, places, events, reviews, lists, and so forth—the things people frequently blog about. People use them today to embed structured metadata in web pages and blogs so that their content can be more easily discovered, indexed, aggregated, and summarized. Soon, microformats will also be used by knowledge services. Microformats are an important development for the AI community because they promise to significantly increase the availability of structured web content.

Microformats adhere to five basic design principles, articulated by Tantek Çelik of Technorati, which appear to be driving their popularity and rapid adoption (See table 4):

First, design microformats to solve a specific problem, such as keeping track of your friends, or sharing calendar events. Develop them in tandem with an initial web service or application that uses them. As John Seeley Brown observed,[6] Microformats win because they include pragmatics as well as semantics.

Second, start with the simplest possible representation for the problem at hand, deploy it quickly, and add more features only when forced by a real application. By simple, think vCard—where a person might be represented initially by just their name and contact information.

Third, design microformats for humans first and machines second. Microformats consist of a little bit of metadata embedded in ordinary HTML—just enough so that a computer can understand the meaning of what is being displayed. In Tantek's words, information on the web should be "Presentable and Parseable." Microformats do that, as simply as possible.

Fourth, whenever possible, reuse existing well-known standards rather than wasting time and money inventing and marketing new ones. HTML, hCard (based on the vCard format), and hCalendar (based on the iCalendar format for events)—few standards are more well-known.

Fifth, microformats should be modular and embeddable. A reservation booking service, for example, might use as input a composite microformat that contained the previously mentioned microformats for personal information and event descriptions.

- Solve a specific problem
- Start as simply as possible, and evolve
- Humans first and machines second
- Reuse existing widely adopted standards
- Modular and embeddable

Table 4: Tantek Çelik's Microformat Design Principles

Figure 6 illustrates how adding tiny bits of markup through microformats can elegantly transform any website into a structured knowledge source. On the left of the screen is a page from Eventful,[7] a website that aggregates event listings. On the top is the vanilla HTML source for that page, with a few highlighted insertions. These are hCalendar tags, and they tell a computer the meaning of each block of displayed text. Note that the hCalendar tags piggyback on existing HTML syntax extensions for Cascading Style Sheets, such as <span> and <class>. Thanks to the hCalendar tags, Eventful users can highlight events of interest and import them directly into their personal calendars, in this example Apple's iCal program. A free web service[8] translates the hCalendar-formatted event listings used by Eventful into the iCalendar standard. Soon, a personal agent might use these listings to alert users to upcoming events of interest. They'll never again have to kick themselves for missing an event because they didn't hear about it in time.

Ideally, one should be able to extract microformatted content from any web page for use by knowledge services and applications. The Semantic Highlighter project at CommerceNet Labs is a step toward that goal. In the screen mockup shown in figure 7, the user has highlighted selected items of interest on a web page. The program then creates a structured representation of that information by selecting and populating relevant microformat(s). In this example, the software is filling out a vCard describing me. Note that it has to follow the URL to CommerceNet's website to obtain my contact information. In general, given some highlighted data on a web page, an arbitrary amount of inference may be required to pick relevant microformats, and fill in missing fields. More about microformats is available at microformats.org.

Vertical Search. Another concept for making the web more semantic is vertical search. Horizontal search engines such as Google work well for finding popular websites. However, when the site wanted is ranked 13,000 out of 147,000, or when the answer to a question must be compiled from information spread over many sites, current search engines are found wanting.

Vertical search engines overcome these limitations by using models of the user, the domain, and the available information sources to determine where and how to look. Vertical search often exploits deep web sources that aren't accessible to Google's spiders, either

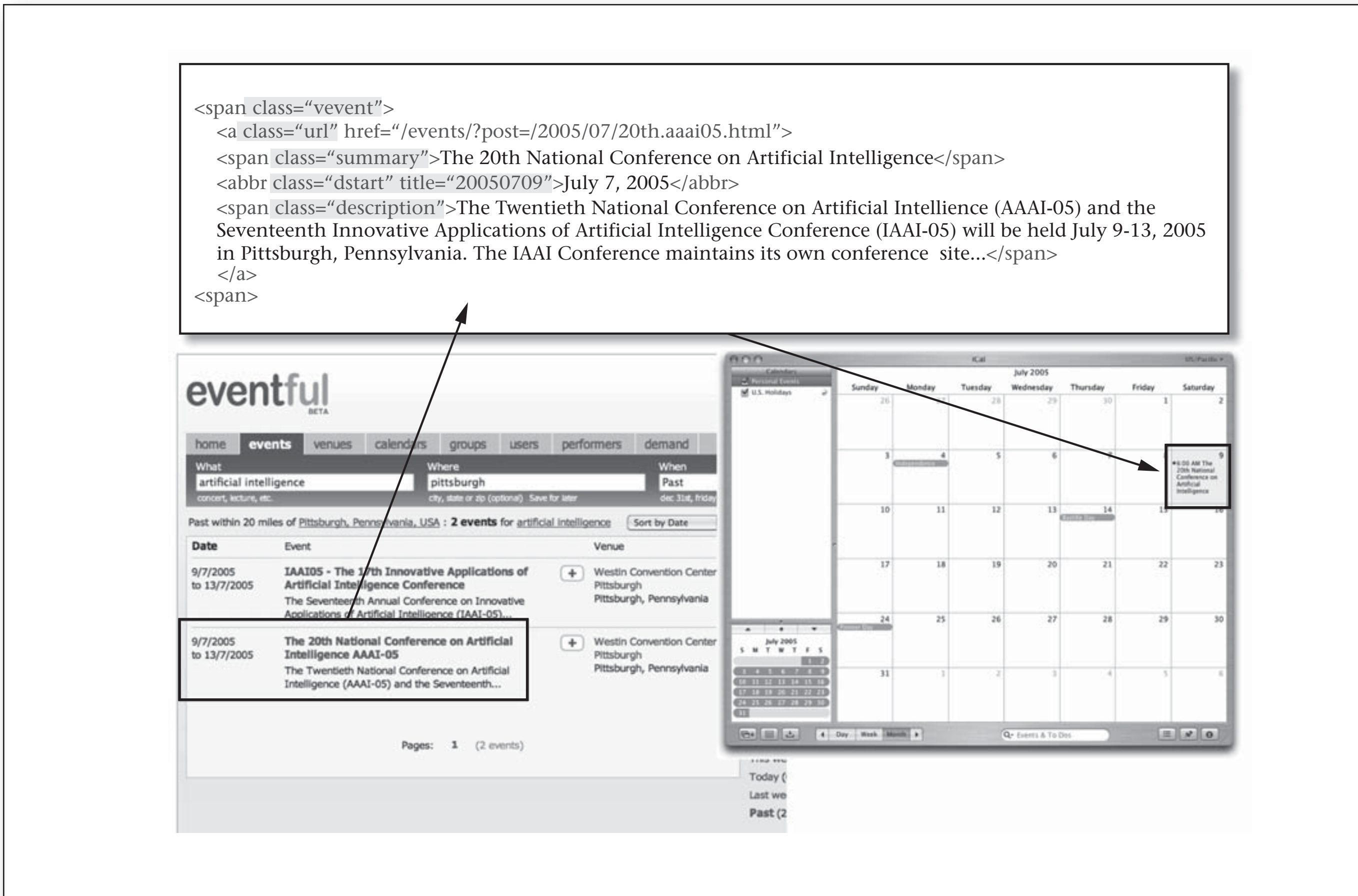

Figure 6. The hCalendar Microformat.

because they are proprietary or require specialized access protocols (for example, SQL queries, filling out forms). The raw information retrieved from these sources is initially organized into data or knowledge bases. Domain and user models are then applied to connect the dots and return useful answers. Consider three examples: Zoominfo, Dulance, and Medstory.

Zoominfo is a vertical search engine for people in the business and professional worlds. Its spiders scour the web for information about individuals, which is then aggregated into personal curricula vitae (CVs). AI-based techniques are used to recognize redundant CVs that describe the same person, and merge them into a composite summary of career accomplishments. Typing in my name, for instance, produces a CV compiled from more than 100 distinct Internet sources. It's quite comprehensive and reasonably accurate.

Dulance is a shopping search engine that scours the web for information about products and prices. Dulance is not the first shopping bot, but it's the first one I know of that offers syndicated feeds to inform users when the memory card they want becomes available at their price.

Medstory[9] is a healthcare search engine. Its spiders scour the deep web, focusing on high-quality information sources that are inaccessible to Google's web crawlers, such as the National Library of Medicine's PubMed database and subscription-based medical journals. It organizes the information it finds in a relational database and then applies domain models of medicine and related fields to infer deeper connections. For example, it can use models of molecular pathways that are common to several diseases to suggest novel uses of drugs and infer potential side-effects.

Looking ahead, I see vertical search engines becoming a major source of structured information for knowledge services. Agents will be able to submit microformatted query forms containing some blank fields and receive microformatted responses with those fields instantiated. Search engines themselves will increasingly rely on collaboratively tagged and

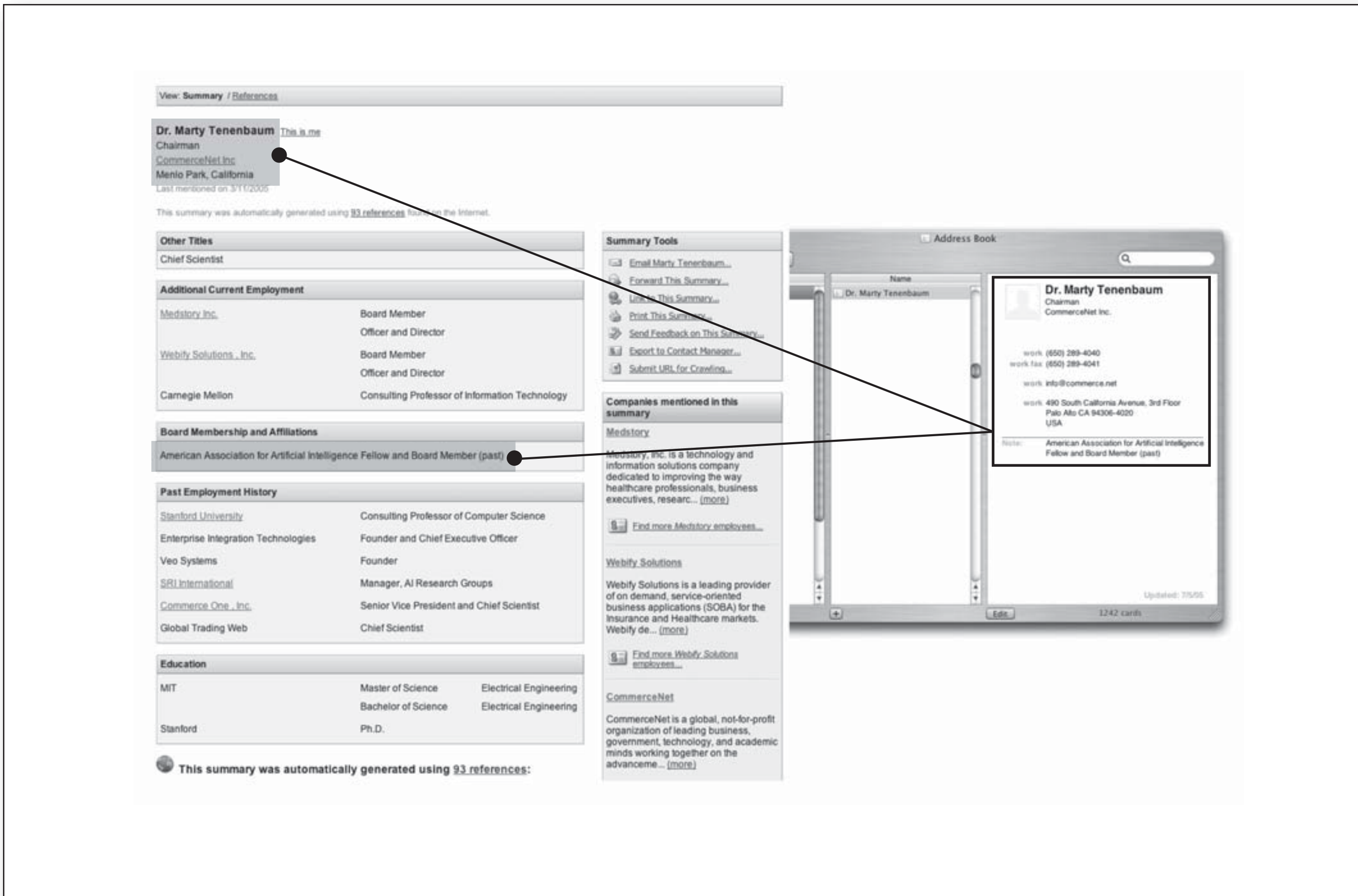

Figure 7. The Semantic Highlighter.

microformatted content to improve search results, especially within professional communities like healthcare and pharmaceutical research.

The Real-Time Web

The third Web 2.0 theme is making the web more real time and event driven. A new generation of real-time search engines, such as Technorati and PubSub, are indexing the fast-changing subset of the web that some call the "World Live Web." Technorati provides the latest blog, Flickr, and Del.icio.us posts, usually within minutes of their posting. PubSub tracks more than 12 million sources, primarily blogs and news feeds. Users subscribe to topics of interest and receive instant message (IM) alerts as soon as items of interest are published.

How do these real-time search and alerting services work? Several major blogging tools ping a trackback cloud when new items are posted. Technorati and PubSub.com monitor this cloud and use the postings to update their databases and alert users. While perhaps adequate for tracking blogs, which change relatively slowly, such centralized notification services are not likely to scale. A truly Internet-scale event bus must accommodate potentially billions of agents publishing and subscribing to each other's information and events in near real time (100 milliseconds to a few seconds, as fast as the underlying networks permit). And it must route information intelligently, not blindly copy spam. A supply chain manager tracking events at airports is likely to be interested only in those events that could interrupt the supply chain—a fire or strike for example, but not the appointment of a new landscaping contractor; and once notified, he or she doesn't need 20 redundant messages from other news sources on the same event. Such a notification service requires a federated peer-to-peer architecture and intelligent agents that can filter and route content using domain knowledge and context, not just keywords.

At CommerceNet Labs, we're prototyping such a federated, Internet-scale event bus.

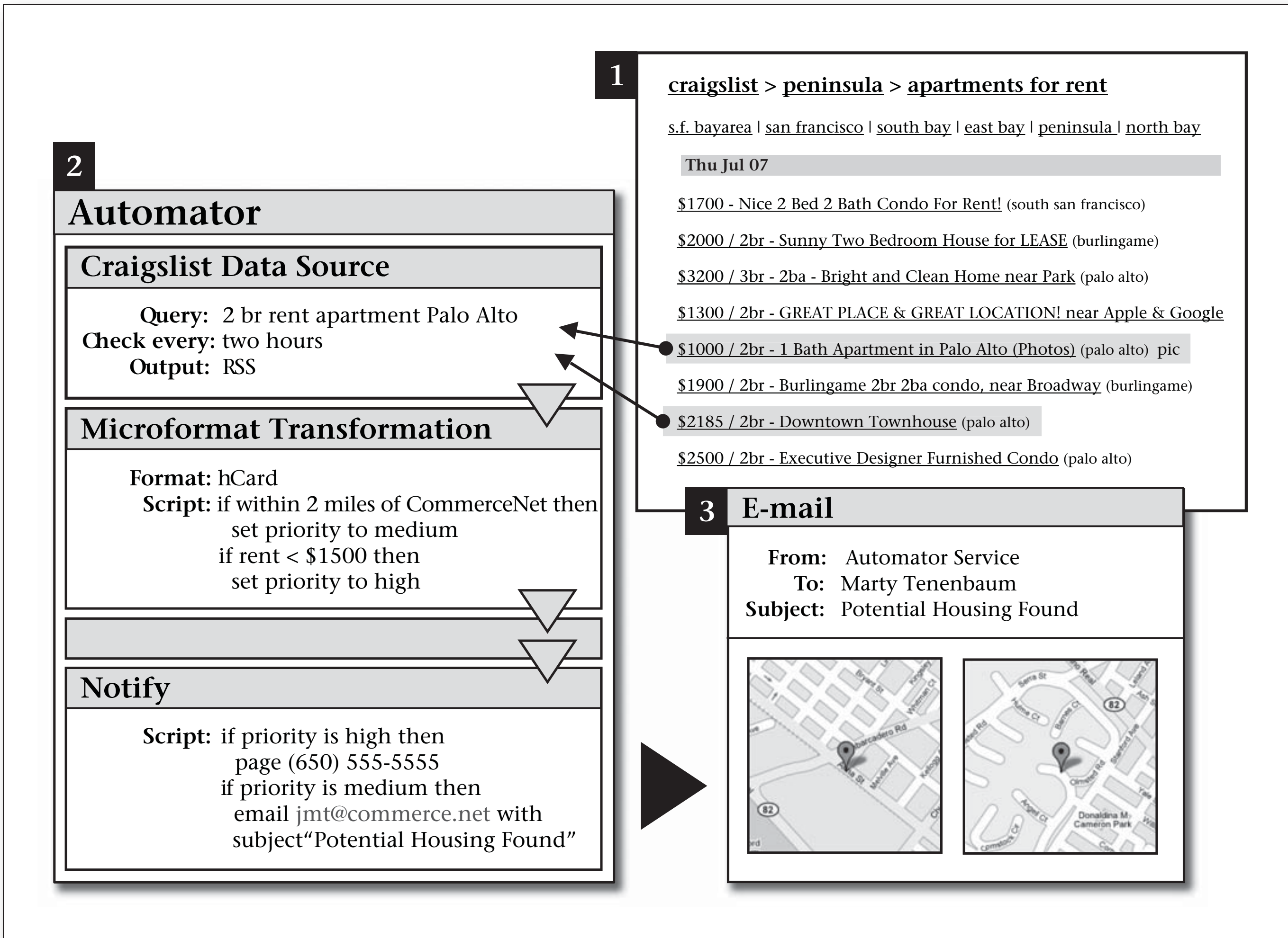

Figure 8. Transforming Websites Into Knowledge Services through Personal Agents.

Agents aggregate events in their areas of interest and also act as facilitators, routing selected items to other agents. An agent aggregating auto industry news, for example, might pass along items that affect the price of cars to an agent aggregating car rental industry news. That agent, in turn, might forward news affecting the price or availability of rental cars to an agent aggregating travel industry news. CommerceNet's notification service is being built on TPd, an experimental pub-sub network infrastructure for semistructured syndication feeds. Unlike other middleware systems, it is designed from the ground up to deal with the potentially hostile world of decentralized information, where multiple competing sites and users are sharing their own (possibly conflicting) opinions of the world. Aggregation, filtering, and propagation occur throughout the network at edge nodes, eliminating central points of congestion. In AI terms, TPd provides a decentralized "blackboard" for triggering agents when the world changes.

Internet-scale pub-sub is a potentially disruptive technology because it reduces the need for big aggregators like eBay and Google. Once millions of players can directly subscribe to offers to buy and sell, they'll invent many new business models and ways to add value. When the information you need can find you—that your order is ready for pickup, that a new drug is available for your disease—without your having to search for it, life will be a little easier.

Community Empowerment

The fourth major thrust of Web 2.0 is about empowering end users. A great example is Greasemonkey. Greasemonkey is a Firefox extension that automatically invokes JavaScript scripts when selected URLs are loaded. The scripts can transform the appear-

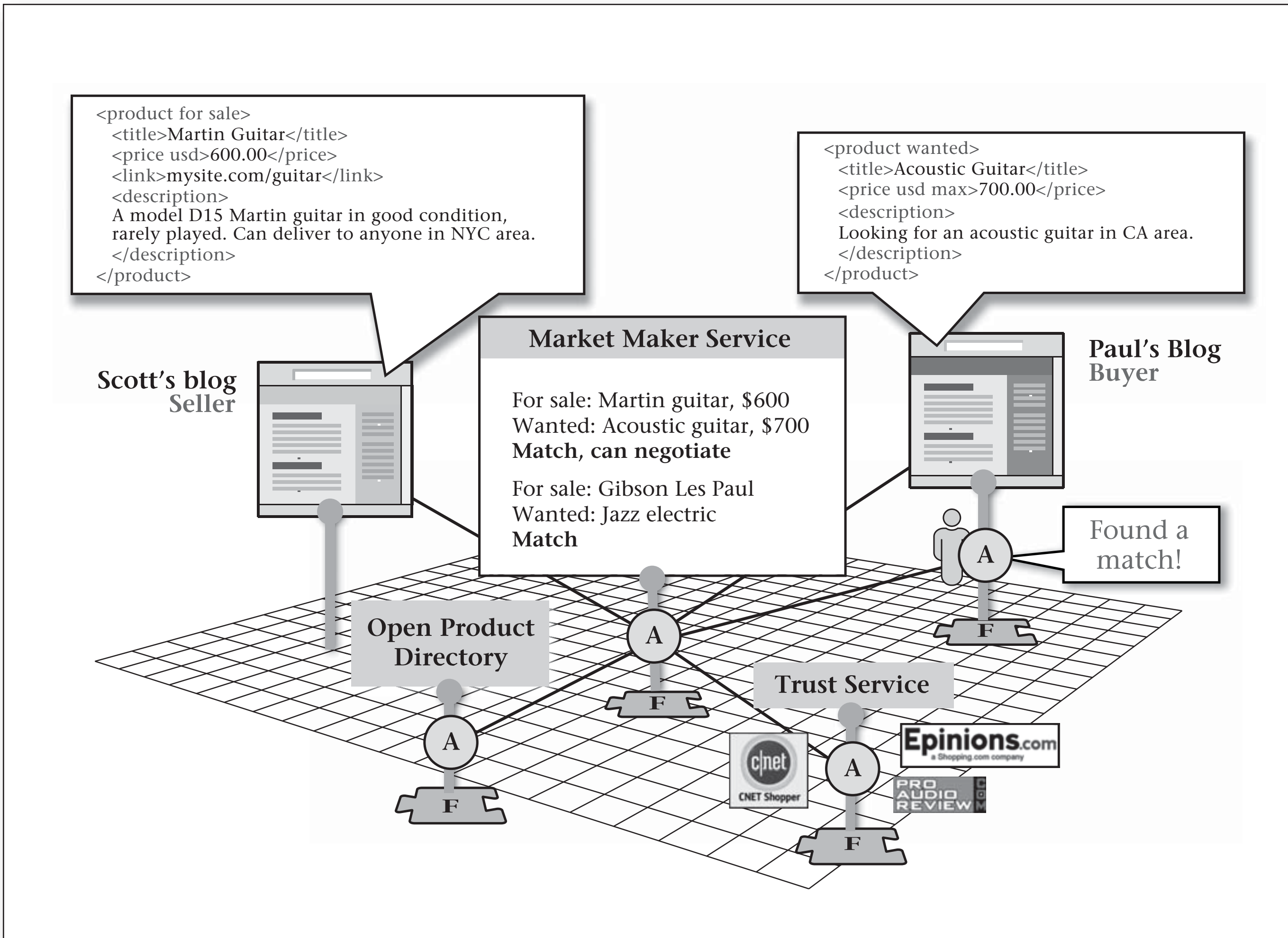

Figure 9. zBay.

ance of pages, perform computations, and even invoke external web resources. A popular Greasemonkey script is Book Burro, written by Jessie Andrews at the University of Kentucky. Activated by any Amazon page offering a book for sale, Book Burro fetches and displays prices from competing booksellers. Another good example is Greasemap. Greasemap scans every page for street addresses, sends them to Google's mapping service, and displays the results at the top of the page.

By inserting computation between the viewer and content source, Greasemonkey scripts let users take charge of their screen real estate. Imagine the possibilities if one could insert logic, decision making, and learning. What if ordinary users could create scripts using visual programming techniques, without having to become JavaScript wizards? Our goal at CommerceNet Labs is precisely that: to give everyone a personal agent tool that can turn any website into a knowledge service. Here's a prospective scenario, inspired by the Semantic Highlighter, and by Apple Automator, a popular visual scripting tool for Macintosh computers.[10]

The task is to find a reasonably priced apartment near work. I start by going to Craigslist[11] and highlighting a few apartments of interest (step 1 in figure 8). Generalizing from those examples, my agent infers that I'm interested in two-bedroom apartments in the Palo Alto area. Next, I create the Automator-like script shown in step 2 of figure 8: "Check every two hours for new listings. If they're within two miles of work then notify me by e-mail (see step 3 in figure 8). However, if the rent is less than $1500, page me." Imagine millions of people composing scripts like this and publishing them as services that others can use and build on. Such a vision would indeed transform AI and the web as we know them.

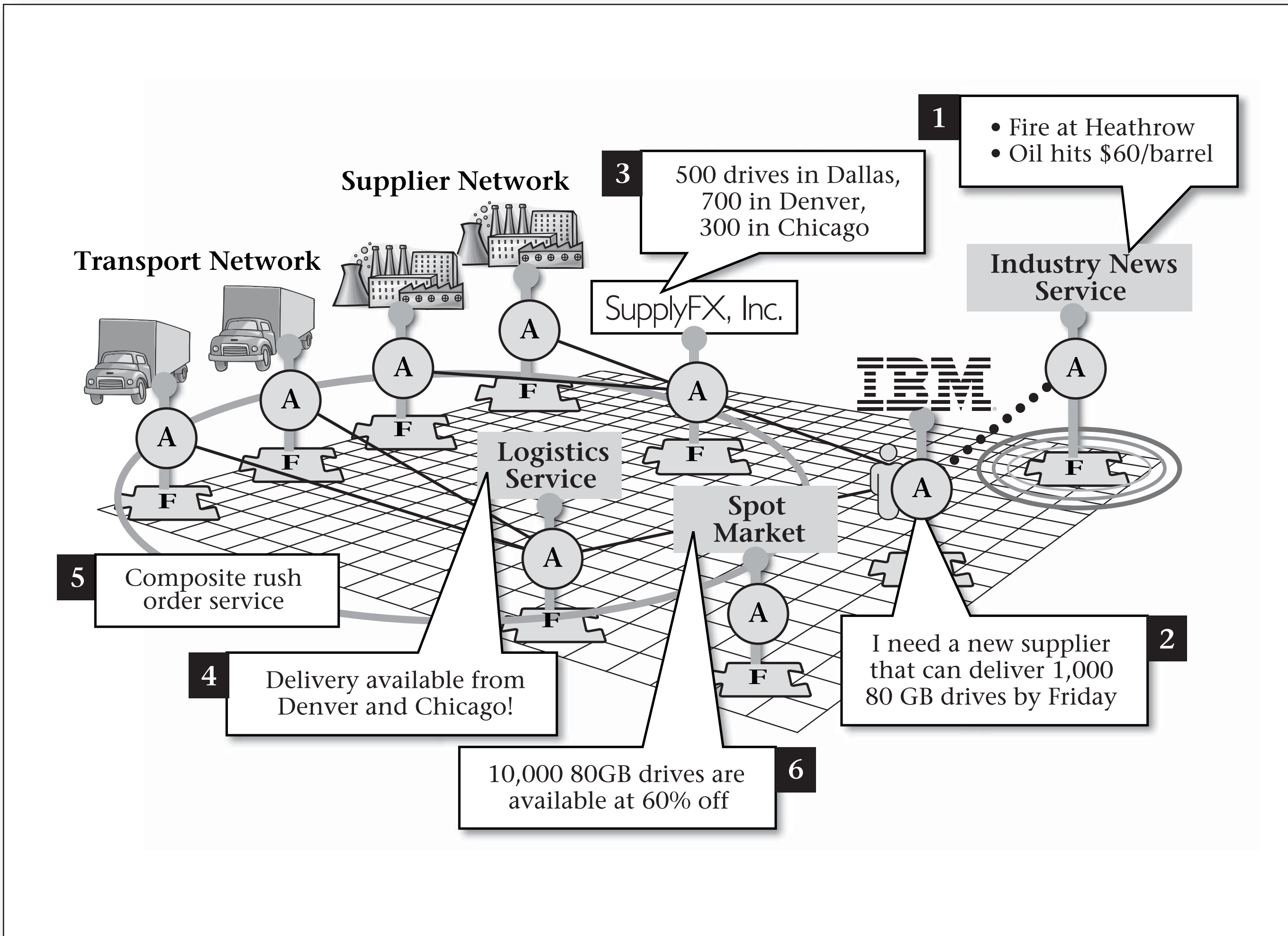

Figure 10. Electronics Industry Knowledge Services.

Case Studies and Scenarios

Having concluded our tour of Web 2.0, we're now ready to talk about growing Internet-scale knowledge systems that combine Web 2.0 concepts with intelligent agents and web services. I'll illustrate such systems using both visionary scenarios and case studies of actual deployed systems.

zBay

zBay (as in decentralized) is an e-commerce scenario we're exploring at CommerceNet, that involves literally blowing up today's centralized e-commerce ecosystems. Think eBay without an eBay in the center! zBay was inspired by a prescient 2002 blog posting by writer and consultant Paul Ford.[12] Ford wanted to illustrate the potential of the semantic web. However, his scenario is a lot more credible with Web 2.0 technologies like microformats and event buses.

In zBay, buyers and sellers post their desires directly on their blogs, using a microformat for classified ads (see figure 9). In this example, Scott on the left wants to sell a Martin guitar for $600, while Paul on the right is seeking an Acoustic guitar under $700. The problem is how to connect them. Step one is aggregating the raw listings. Posting tools might submit listings automatically to large aggregators such as Google and Craigslist. Smaller, specialty aggregators could spider for listings. These specialty market makers can now apply deep domain knowledge, including open wiki-based product ontologies, to match up compatible buyers and sellers. A market maker specializing in used guitars would know, for example, that most Martin guitars are acoustic, while most Gibsons are electric. The big advantage of decentralized markets is that many specialists can add value by bringing their domain knowledge to bear.

Blowing up the center unleashes other

opportunities for innovation. Paul himself might decide to become proactive and task his personal agent with monitoring favorite dealer sites for new guitar listings. Knowledge-based trust services can lubricate the market, using social networking and blogs to track the reputations of buyers and sellers and to compile product reviews. Other service providers can facilitate fulfillment, shipping, and payment. These ideas extend directly to industrial-strength business-to-business (B2B) applications.

Electronics Industry Supply Chain

The electronics industry is fiercely competitive. Short product cycles make forecasting supply and demand difficult. Error rates of 50 percent or more are not uncommon. Firms must closely monitor changes in supply and demand and respond instantly to keep them balanced. The traditional sources of supply and demand signals are channel partners, such as distributors and resellers. Increasingly, however, the most up-to-date information comes from Web 2.0 sources including news feeds, blogs, and real-time sensors.

Let's return to our earlier electronics industry scenario (figure 3) and extend it to accommodate these Web 2.0 resources. In the spirit of incremental automation, a supply chain manager might initially just have a personal agent monitor relevant feeds and notify him or her of events potentially warranting attention (figure 10, step 1). Someone who had a critical shipment of disk drives that was scheduled to be routed through Heathrow around the time of the 2004 London bombings would have been wise to explore alternative sources of supply. The manager or his or her personal agent prepares an RFQ and submits it to SupplyFX,[13] a live, commercial knowledge service for sourcing electronic parts (step 2). SupplyFX expedites sourcing by selectively forwarding RFQs to interested suppliers in its network and aggregating their responses.

Fortunately, drives are available from several suppliers (step 3). But can they arrive by Friday? Our supply chain manager submits that requirement to a third-party logistics service, which queries its network of carriers and determines that Friday delivery is possible from Denver and Chicago (step 4). Problems at Heathrow will not disrupt production! SupplyFX, sensing an opportunity, assembles and publishes a one-stop composite service for rush orders that takes into account both product availability and shipping schedules (step 5).

But why stop here? There are potentially thousands of value-added knowledge services. How about a knowledge service that tracks suppliers and learns which ones provide the most reliable on-time delivery? Or a demand forecasting service that monitors blogs for subtle demand signals, such as an emerging interest in 80 gigabyte iPods. Or a spot market service that notifies subscribers of buying opportunities that arise when another manufacturer loses a large order and is forced to dump inventory (step 6). No ERP system can deal effectively with such opportunistic, real-time information. Humans aren't very good at it either, due to information overload and limits on time-bounded decision making. Couldn't a little AI help?

Case Study: Webify's Insurance Ecosystem

Webify[14] is commercially deploying simple agent-based knowledge systems today in the insurance industry. The insurance industry is a complex distributed ecosystem. It includes the primary carriers, their agents, underwriters, and claims adjusters, reinsurers, fraud investigators, and many other service providers (See figure 11). Webify uses agents, operating within a service-oriented architecture, to route documents among service providers based on content. Routine claims, for example, can be routed automatically to $5-per-hour adjusters in Bangalore, while major claims with the potential for significant payouts are routed to $80-per-hour in-house specialists in Hartford. This ability to selectively outsource claims is especially helpful following disasters, enabling carriers to handle peak loads without unduly inconveniencing other customers.

There are many other opportunities for knowledge services to incrementally improve the performance of distributed insurance ecosystems. For example, new policy applications for coverage can be automatically offloaded to reinsurers when the exposure to a given risk exceeds predetermined levels. An outright moratorium on new hurricane coverage can be instantly imposed at the first indication of an approaching storm. Suspect claims can be forwarded to fraud investigators, and so forth.

Insurance was a prime domain for expert systems in the 1980s. Several companies, such as Syntelligence, were formed specifically to address this lucrative market. None are still around. Their technology consisted of large, monolithic mainframe-based systems. It was ill-suited to decentralized insurance ecosystems, where integration and automation are best approached incrementally, one organization and one task at a time. Web services over-

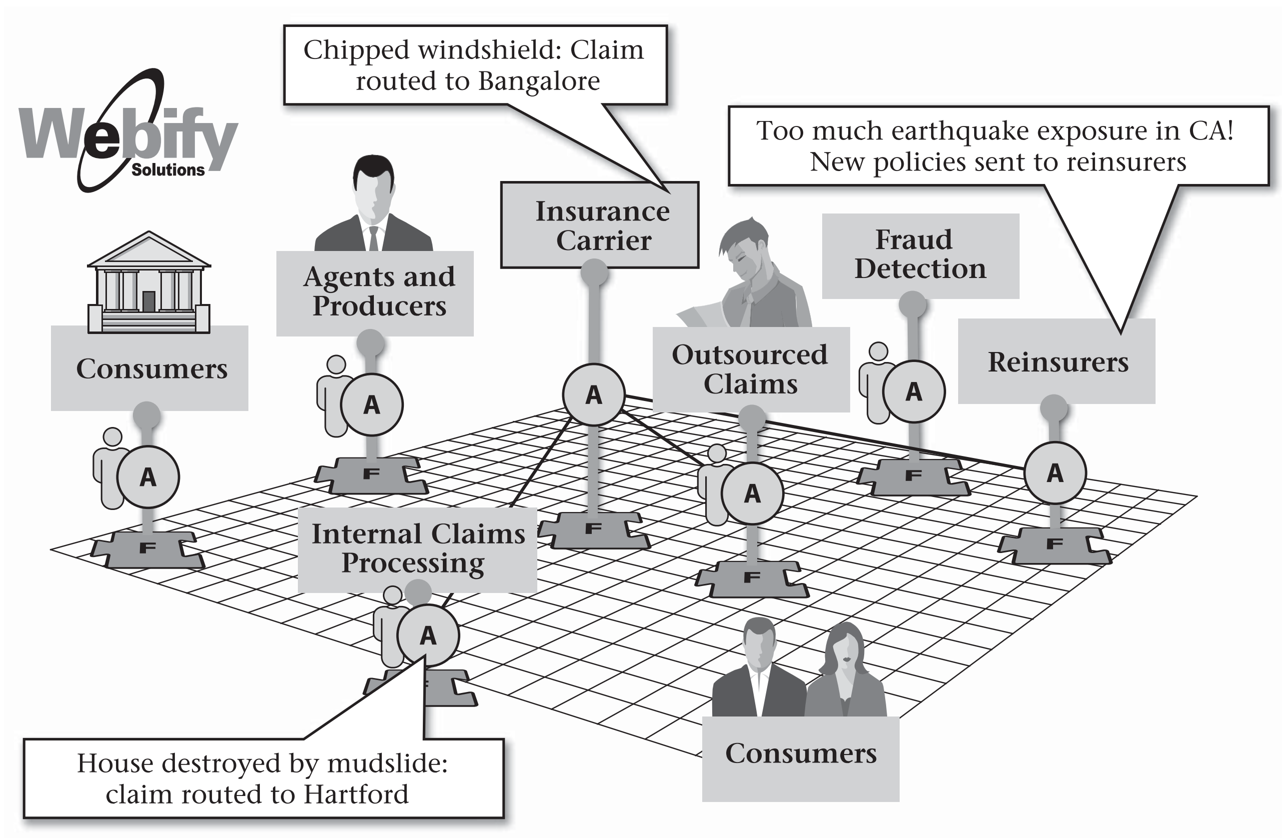

Figure 11. Insurance Industry Knowledge Services.

come this problem in insurance and other industries. They enable companies to blow up existing value chains and reassemble the pieces into radically new cross-company processes and business models.

E-Science

Scientific problems are increasingly complex and multidisciplinary, and thus beyond the reach of a single individual or group. In response, many scientific communities are now relying on Internet-based grids for sharing information, computing, and other services. A recent issue of *Science* contained three papers on service-oriented science.[15] Most of the services discussed, however, involve only the sharing of data and computing resources. Knowledge services offer a significant extension, enabling groups to share their specialized knowledge and expertise.

Alain Rappaport of Medstory is pioneering the application of knowledge services in biomedical research. In collaboration with colleagues at the Massachusetts Institute of Technology, Carnegie Mellon University, and the Institute for Human and Machine Cognition, he's developing a knowledge services grid for mobilizing the resources of the global scientific community in response to biological threats (see figure 12). Suppose a virologist at CDC is confronting a suspicious new virus? Urgent questions spring to mind. Is it natural or engineered? How is it molecularly related to other known infectious agents? Does that relationship present opportunities for intervention?

The e-science grid springs into action. First, a sequencing center with cutting edge viral RNA chips analyzes the new virus's genome and publishes it to the grid (figure 12, step 1). Based on that data, a knowledge service associated with the leading gene expression database concludes that it is indeed a new bug (step 2). Other knowledge services look for close relatives to the new bug based on its phylogenetic

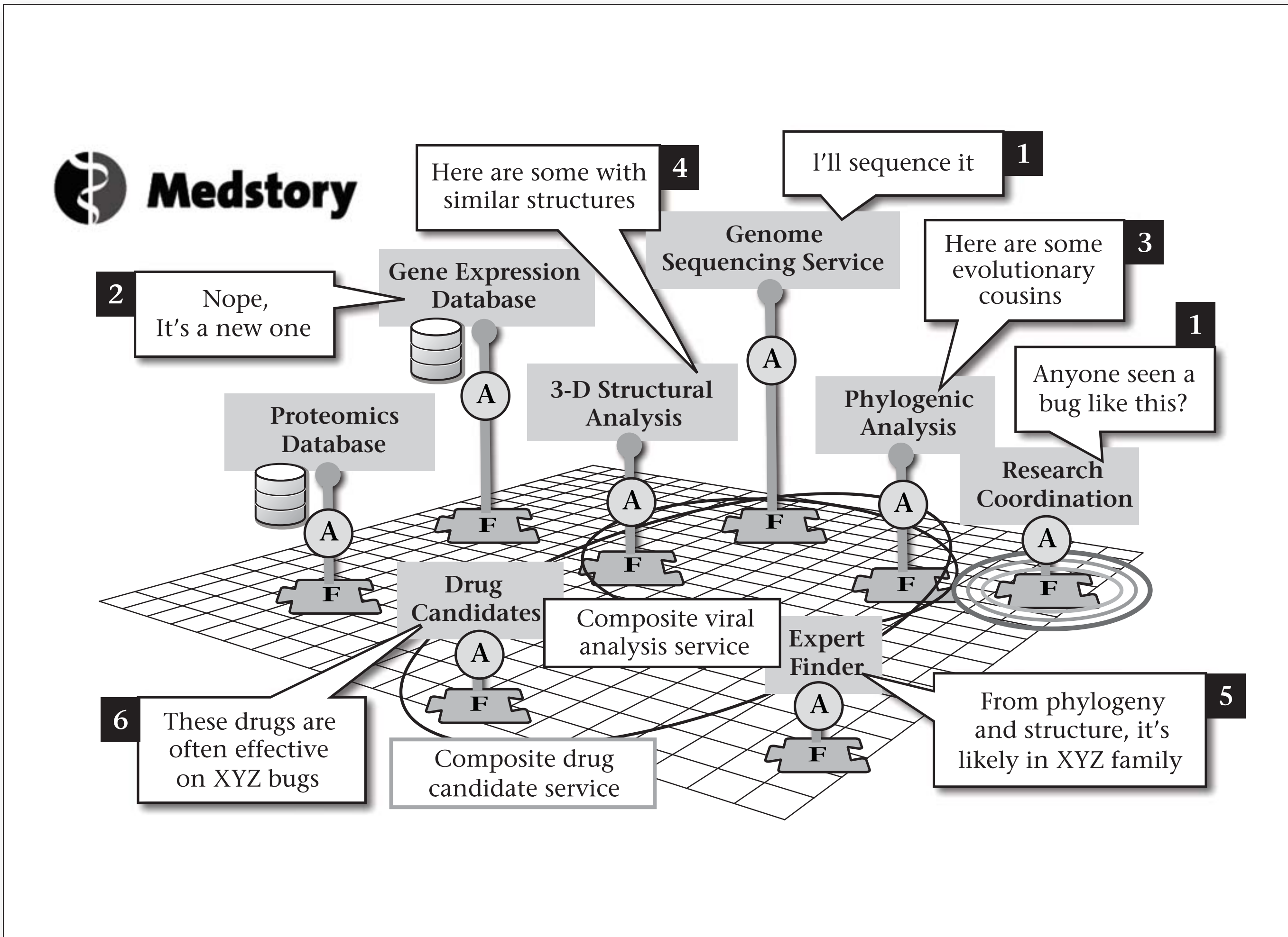

Figure 12. Service-Oriented Science.

class (evolutionary neighbors) and three-dimensional molecular structure (steps 3 and 4). Experts on these related viruses are alerted. One of those experts creates a knowledge service that synthesizes the phylogenetic and structural analyses to determine the precise family of known viruses the new bug most closely resembles (step 5). A pharmacology expert can now combine this composite virus classifier with a service that provides access to a database of antiviral drugs, creating a knowledge service that can take a virus's DNA and suggest the best drugs to defeat it. (step 6).

The circles associated with steps 5 and 6, and thousands more like them, represent composite knowledge services. Most will be created by scientists for their own use and then made available for others to build on. Collectively they will transform how science is done. Knowledge services are already impacting how drugs are developed (and not just for bioemergencies). Increasingly, drugs are developed collaboratively by academic researchers, biotech scientists, and big pharma marketers, supported by a vast ecosystem of third-party services—combinatorial chemistry, toxicity assays, proteomic analyses, clinical trials management, and so forth. Knowledge services help facilitate the sharing of information and expertise, the allocation of resources, and the coordination of workflows across organizations.

E-Life

My final example is an e-life scenario: planning a dinner for tonight in Pittsburgh. The task breaks down into a series of subtasks (see figure 13): Determining who is in town that I might like to spend some time with. Are they available and interested in meeting for dinner? Finding a restaurant, booking reservations, making arrangements, and coordinating any last minute changes in plans. These are precisely the types of tasks the web excels at.

Let's start with who's in town that I might

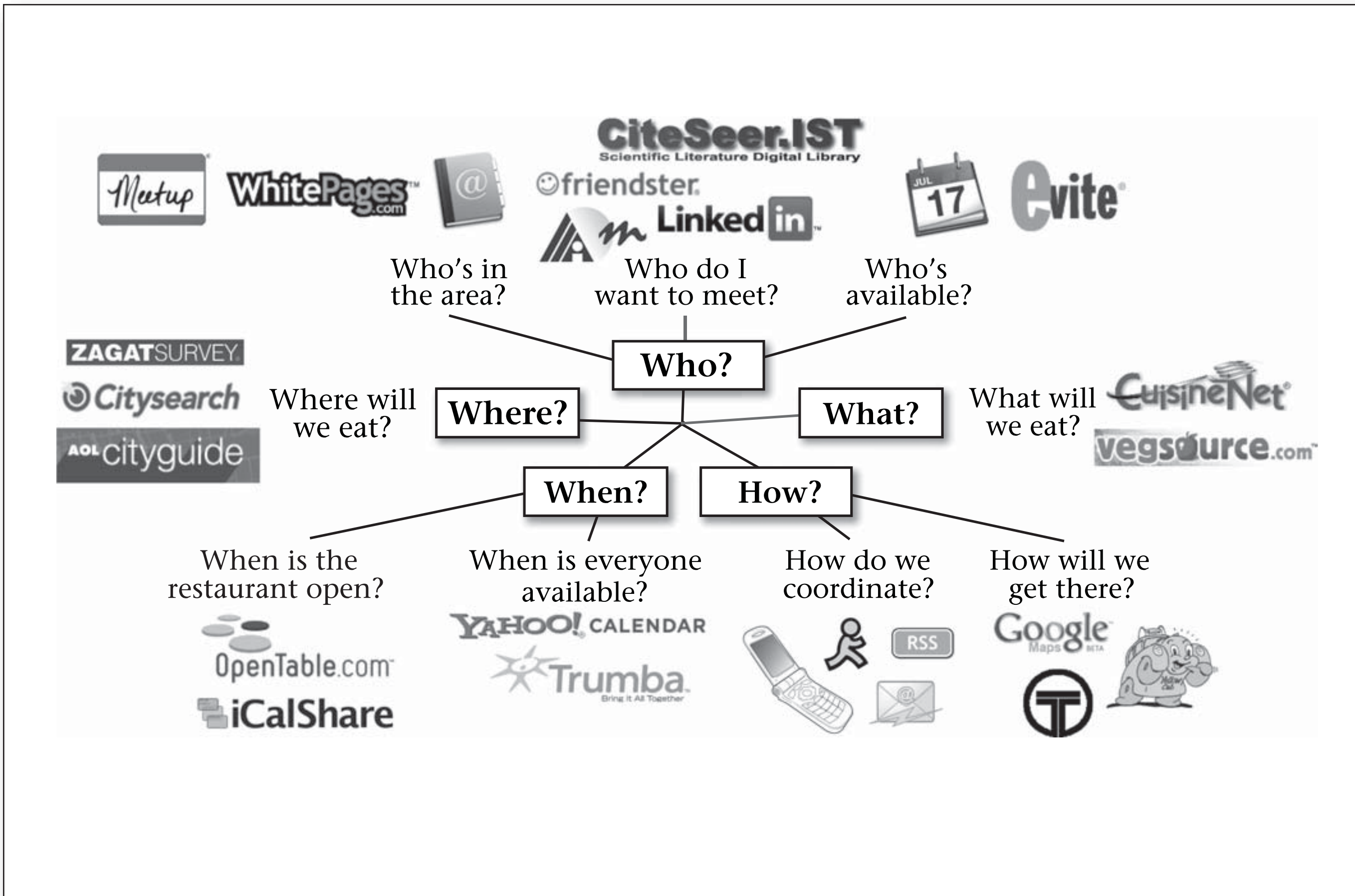

Figure 13: E-Life.

like to meet. That list would include people I know who live in Pittsburgh, as well as conference attendees with whom I have something in common—perhaps authors of papers I've liked or colleagues who liked the same papers I did, or those who fund research in my area. I might start by asking my personal agent to compile a list of the people I know who live in Pittsburgh, using contacts from my address book and social networking sites. To compile a list of conference attendees, someone would have to create a service that scraped the conference site for speakers, session chairs, and the like, and published the results as a microformatted list; in the future, conference organizers might routinely publish structured lists of registered attendees. Someone else might build a collaborative filtering service for CiteSeer, where users could share ratings of papers. Given such services, a simple agent could correlate the list of conference attendees with the authors of papers I liked, to suggest possible dinner guests.

Given a list of potential dinner guests, an agent could use e-mail or a service like Evite to see who's available. An incremental service, built on Evite, might offer responders the opportunity to fill out a microform, indicating preferences regarding cuisine, location, and so on. Next up is selecting a place to eat. There are numerous sites on the web that rate and review restaurants. A helpful knowledge service could use these existing sites, together with guests' indicated preferences, to recommend restaurants. Another knowledge service could then check availability at OpenTable, book the reservation, and notify the guests.

Each of these knowledge services is helpful in its own right, so it's a good bet that someone will create services like them. Once they exist, it won't take long before someone drops in a constraint-based planner and offers a completely automated dinner planning service.

Conclusions

I'm convinced that the time has come to rethink how we do AI applications in the context of the web. This is not an academic exercise. It's not a question of if or even when this will happen. It's already happening in the web community. We just need to insert a little AI! And if the AI community doesn't step up to this opportunity, others will reinvent AI for us.

Stripped to its essence, I'm talking about (1) using microformats and agents to transform ordinary web resources into knowledge services; (2) enabling these knowledge services to communicate directly and in a content-directed fashion, through events, to create new compound knowledge services, and (3) providing millions of end users with personal agent tools to do this on an unprecedented scale, all building upon each other to scale exponentially like the web.

The result would be millions of simple, interacting, Internet-scale knowledge systems, built through mass collaboration, which support how people actually work and live. And because they're human-machine systems, they could utilize any web resource. Such systems would stand in stark contrast to contemporary AI and semantic web applications, which are typically built by a handful of researchers to perform deep reasoning on a tiny subweb of formally structured information. Internet-scale knowledge systems, by contrast, are developed pragmatically: Start with the most basic functionality that does something useful. Get it out quickly so others can build on it. Automate complex tasks incrementally; and never let a lack of standards get in the way of connecting things.

The resulting systems are not incompatible with the semantic web. Feel free to use the resource description framework (RDF) if you like—it's just another microformat, one particularly good for deeper reasoning. Indeed, microformats are a great way to bootstrap the semantic web and get traction on problems that people actually care about. Some in the Web 2.0 community have begun referring to their microformat-based applications as the lowercase semantic web. I prefer semantic web 2.0, which appropriately acknowledges the contributions from both the semantic web and Web 2.0 communities.

But Is It AI?

Internet-scale knowledge systems are clearly useful. But are they intelligent? Recall Newell's criteria for intelligent systems (table 1). Most websites exhibit at least a few of these characteristics. Moreover, each of them can be found somewhere on the web. Newell, however, was adamant that all characteristics must be exhibited simultaneously.

A few megasites, such as Amazon, come close to meeting Newell's condition. Amazon has adaptive goal-oriented behavior—it tries to sell you books. It certainly learns from experience—what to recommend. Someone has built a speech interface using Amazon's published APIs. And Amazon even has a modicum of self-awareness—the ability to recognize when servers go down and take corrective action. So is Amazon intelligent?

Begging that question, the real issue is whether a decentralized web of knowledge services, each of which may satisfy at most a few of the criteria, can collectively achieve true intelligence. The jury is still out. However, for several reasons, I'm sanguine the answer will be yes. First, as Google and others have demonstrated, some of Newell's criteria become simpler at Internet scale—statistical learning and using large amounts of knowledge, for example. Second, like open source software, decentralized knowledge systems tap the collective wisdom and experience of large numbers of developers (potentially millions), rather than a handful of researchers. Although most of the resulting knowledge services likely will be simple compositions of existing building blocks glued together with end-user scripts, the programming can be arbitrarily complex. Even simple composite knowledge services can be arbitrarily powerful when their components can include anything from a FedEx-class logistics planner to a Newell-class general problem solver. Clearly, conventional AI systems provide a lower bound to the collective intelligence achievable by an ecosystem of cooperating knowledge services.

Neil Jacobstein suggested to me an intriguing third argument: in an ecosystem of millions of diverse knowledge services, which evolve through composition, amid intense competition, it seems likely that natural selection will apply—selecting for those with greater "fitness" to their business, scientific, or personal environments.

For all these reasons I'm confident we're on the right track. We're finally building software that works the way a free society does—decentralized, evolutionary, democratic, and tolerant. Markets aggregate the collective intelligence of humans and computers, and rapidly disseminate the best ideas and knowledge services. Mass collaboration bootstraps the resulting components into systems with unprecedented intelligence and flexibility. These ideas are reminiscent of Marvin Minsky's *Society of Mind*.[17] As AI meets Web 2.0, they are ideas whose time has come.

I invite you to contact me or visit the commerce.net website[16] to learn more about oppor-

AAAI Special Award Nominations for 2007

AAAI is pleased to announce the continuation of its two special awards in 2006, and is currently seeking nominations for the 2007 AAAI Classic Paper Award and the AAAI Distinguished Service Award.

The 2007 AAAI Classic Paper Award will be given to the author of the most influential paper(s) from the Seventh National Conference on Artificial Intelligence, held in 1988 in St. Paul, Minnesota.

The 2007 AAAI Distinguished Service Award will recognize one individual for extraordinary service to the AI community.

Awards will be presented at AAAI-07 in Vancouver, Canada. Complete nomination information, including nomination forms, is available at www.aaai.org/Awards/awards. php.

The deadline for nominations is March 15, 2007.

For additional inquiries, please contact Carol Hamilton at hamilton@aaai.org.

tunities to get involved and help realize the vision.

Acknowledgements

This article is based on an invited talk delivered at IAAI-05. I'd like to thank the many colleagues who helped with that presentation—Alain Rappaport, Tantek Çelik, Randy Davis, Dan Gould, Chris Hibbert, Rohit Khare, Greg Olsen, Barney Pell, Raj Reddy, Adam Rifkin, Manoj Saxena, Allan Schiffman, Kragen Sitaker, Jay Weber, and especially Neil Jacobstein, who persuaded me that these ideas were worth talking about, and Kevin Hughes, whose illustrations and collaboration helped crystallize them.

Notes

1. movabletype.org.
2. blogger.com.
3. wikipedia.org.
4. flickr.com.
5. del.icio.us.
6. CommerceNet Workshop on Decentralized Commerce, Wharton West, San Francisco, 20 June 2005.
7. eventful.com.
8. Available at suda.co.uk/projects/X2V/.
9. medstory.com.
10. apple.com/macosx/features/automator/.
11. craiglist.org.
12. ftrain.com/google_takes_all.html.
13. supplyframe.com.
14. webifysolutions.com (Webify is now part of IBM).
15. *Science* 6 May 2005, 308(5723): 809, 814–824.
16. commerce.net/semweb2.
17. Marvin Minsky, *The Society of Mind*. Simon and Schuster, New York, 1986.

Jay M. "Marty" Tenenbaum spent the 1970s at SRI's AI Center leading vision research, the 1980s at Schlumberger managing AI Labs, and the 1990s founding a string of successful Internet commerce companies, ultimately serving as chief scientist of Commerce One. He now splits his time between Internet startups Efficient Finance, Medstory, and Patients Like Me, and the nonprofit CommerceNet, where he's using AI and the Web to improve healthcare by tapping the collective intelligence of doctors, patients and medical researchers.

Kevin Hughes is an internationally-recognized pioneer in web design and software. One of only six members of the World Wide Web Hall of Fame, in 1994 Hughes was presented the award by Tim Berners-Lee at the First International World Wide Web conference in Geneva. He has worked with Tenenbaum at EIT, CommerceNet, VeriFone, and CommerceOne and is currently the CTO of ChipIn, a Web 2.0 startup based in Honolulu, Hawaii.