Impact for Agents *

Tom Wagner† Les Gasser‡ Michael Luck§ James Odell¶ Todd Carrico ||

1. Introduction

Impact for agents. Most of the agent research community has been predicting greater impact for years and many of us have been working to help the process along. Yet the tremendous growth on the research front has not been met with a corresponding growth in commercial applications. This paper contains a diverse set of thoughts on the issue of developing impact for agents. The authors were not given a set of questions to address but were asked to focus on the problem of developing greater impact for agents from their particular vantage point. Sections range from those that are very applied, e.g., how to "sell" a concept to those that examine the issues from a more intellectual level. Each section is prefaced with a small biography on the author of that section to properly place his/her comments in context. While the material is diverse, it is our hope that these thoughts are either directly useful or help to articulate issues of interest.

2. Motivating Investment – Tom Wagner

Tom Wagner is currently serving a four year term as a Program Manager for the Defense Advanced Research Projects Agency (DARPA) in the IPTO office. Prior to this, Wagner was a principal research scientist at Honeywell Laboratories where he led the agent research group. Previous positions include being an assistant professor at the University of Maine and having leadership roles in several entrepreneurial efforts.

Most of our research community believes that the inherently distributed nature of modern computing mandates agent-based solutions and that with the increasing proliferation of wireless and mobile computing devices, of varying capacities and configurations (i.e., heterogeneous), this will only be more true in the future. However, even with these often uncontested visions for the future, motivating investment in agent technology has been an uphill battle. Without investment, research will be hampered, commercial relevance will be slow in coming, and impact will be muddied at best.

The expression "motivating investment" means getting someone to invest in your technology. This section focuses on that topic in a fairly applied fashion – it is my experience that often we have many of the critical elements needed for impact but that we aren't business people by trade and often do not know how to motivate investment. Where you look for such investment and how you approach it depends on your specific goals and how near term you are focused. There are many involved ways to classify work in terms of time to deployment – for simplicity let us simply state that these remarks tend to be more relevant for those interested in mid to near term deployment, i.e., those interested in having *impact* on the world of computing in the near term. Note that these remarks are *not facts and should not be confused with such*.

Know your customer and understand his/her problem. Pitches that focus on general technology or "my technology can do these neat things" tend not to get you what you want. Most often a given stakeholder has a particular set of problems that he/she will invest in because there are business drivers behind those problems. You may not be able to structure a complete business case for your idea but it is very helpful if you can structure your remarks to be as close as possible to the area of interest for the stakeholder.

If you don't know your customer's problem, create a vision. It is rare for a customer/stakeholder to come to a researcher with a ready made problem. If you don't know their problem, or are seeking more mid-term grade investment, create a vision or concept that is in your customer's interest area and speak to that. Note that this process is inherently more difficult because then you are in the position of having to sell both the problem and sell the solution [5]. However, you may also be able to grab someone's interest with a particularly compelling and novel idea.

Understand your customer's product/market mix. An important tool when developing a new concept is Ansoff's product-market-expansion grid, shown in Figure 1. This is a basic tool that business people use to make sure they expand from a position of strength. In the matrix, expanding right or down from the current state is desirable while expanding diagonally is not. As a researcher, you are generally limited to the one downward transition.

Know your value proposition. You must be able to articulate why someone should invest in your technology. What will they get? Better, faster, cheaper are all common terms. If you can tie numerical values to these (e.g., how much bet-

† Information Processing Technology Office, Defense Advanced Research Projects Agency, twagner@darpa.mil

‡ Graduate School of Library and Information Science, University of Illinois, gasser@uiuc.edu

§ Computer Science, University of Southampton, mml@ecs.soton.ac.uk

¶ Agentis Software, mailbox@jamesodell.com

|| Cougaar Software, tcarrico@cougaarsoftware.com

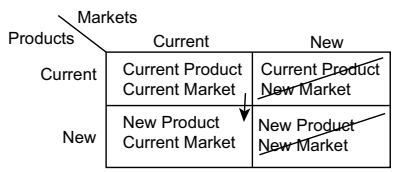


Figure 1. Product/Market Expansion Grid

ter) they are more meaningful.

You have 3.5 minutes to make your point. The number may be 5 minutes but it is not 30. If you cannot clearly articulate, in short order, what you offer, why they should care, and what resources you need to accomplish your task, you should go back over your materials. Make your point at an executive level but be ready with backup.

Shooting too near term is problematic for researchers. By definition, we aren't commercial software developers and we don't generally manage commercial projects. Accordingly we tend to be expensive and not very credible at commercial efforts. We also generally still have a research issue or two that we want to work and this will invariably come out during conversations with your stakeholder – such items hinder very near term investment.

There is always an opportunity cost. Resources are limited. Investing in the maturation of research ideas must have a high estimated return to compensate for the high risk involved. (Even then the dollars may go to lower risk/lower reward activities such as enhancing existing products.)

Early adopters versus originators. Unless some idea is right on the corporate vector, e.g., new brewing technology for Starbucks, many business would rather let someone else incur the initial product development risk and then copy the ideas (or license them) if they turn out to be good ideas.

The term "agent" may not buy you anything. When using the word "agent" I have had reactions that range from (a) "Agents? I already have them" to (b) "Isn't that all just more AI hype?" to (c) "I've never heard of an agent. What do you mean?" The first reaction might imply that the person thinks of agents as distributed objects (this may be a good thing if that is how you view agents also). The problem here is that if they already have them, or perceive them as a simple programming tool, they don't need your (costly) expertise. They can just download an SDK and away they go. The second reaction stems from how overloaded the term is and its unclear semantics (and yes, some hyping of "agents"). The third reaction might seem like an invitation but often that conversation is a red herring. You might wind up discussing agents in general terms and the unclear semantics / wide range of possible topics might leave your listener with a wrong or confused impression. You will also invariably be asked who else is using agents and how it is benefiting their business. Data on that front is sparse at best. Because of these reactions, I no longer use the term "agent" in certain kinds of talks. I'll refer you back to knowing your customer (above) and solving his/her problem (also above). Don't sell agents – sell a solution.

Autonomy is not desired in many domains. Autonomy can be a frightening word in some contexts. Consider using a different expression and for more critical applications keep the human in the loop, i.e., think "decision support."

Distributed computing is not universally recognized as a "good thing." Be ready to make an argument for why a given problem cannot simply be solved in a centralized or monolithic fashion.

If they ask about software issues, you have already sold them on your overall concept. If your customer starts asking about how bullet-proof your agent software is, they are interested enough in the business concept or overall idea that they are trying to assess second level feasibility and risk. Generally this is a very good position from which to operate.

Selling frameworks or middleware seems to be inherently difficult. This is because the value proposition is not clear to anyone who isn't in the middleware business, i.e., first they have to pay for it then they have to figure out what applications to build with it.

The world may be more primitive than you think. As researchers we are used to focusing on future visions – take a look at Windows Notepad before going into your meeting to pitch a non technical person.

You are asking for someone else's money. Whether it is corporate funds, venture capital, or the funds of your government, you are asking for someone else's money. In most cases there isn't a line item in the budget that says "fund agent research" or even one that says "fund AI research" (and possibly not one that says "fund research" or anything close). Be aware that your potential customer probably has no compelling reason to give you their money but must be motivated by you to do so.

Leave the "snake oil" at home. Identify realistic pursuits for your efforts and engage them. If you have to bend and twist your interests too far you either won't be compelling or you won't be happy with the work even if you obtain the resources.

Even if you do it perfectly, you may not get anything. There are barriers to agents, and any leading edge technology, that are beyond your control in the near term. (In the long term, the very act of you trying to obtain investment for a more near term project will help remove these barriers.) The best you can do is present your material well and try-try-again if you do not get traction the first time around.

I continue to believe that the future of computing and "agents" have many things in common. For our community to help shape that future, and for that future to arrive in a more timely fashion, continued investment is required. Motivating said investment may well be as large a challenge as doing the work itself. It is, however, important that some per-

centage of the community recognizes the value of this work and engages in the challenge.

3. Build on strengths and remove barriers – Les Gasser

Les Gasser is an Associate Professor of Library and Information Science, with joint appointments in Computer Science and Computational Science/Engineering, at the University of Illinois at Urbana-Champaign. He has published over seventy technical papers and five books in multi-agent systems. He is currently Past-President of the International Foundation for Multi-Agent Systems (IFMAS), and is one of the founders of the field. A former Program Director for the National Science Foundation handling MAS areas, he has also been a principal or advisor with a number of technology startup firms.

For over 25 years the MAS community has been debating how to make apparent and increase its impact in the AI and computing communities. Twenty-five years ago, F. Hayes-Roth famously justified a new multi-agent viewpoint on AI, saying "All real systems are distributed" [6]. In 1980 *he had to point it out*. Today this is taken for granted. Because the distributed nature of knowledge and action are now so pervasively natural, the impacts of three decades of MAS research effort often seem to have vanished.

Nonetheless, MAS research and technology *has* built two strong pillars of progress. First, it has clarified many distributed, contingent, and interdependent features of knowledge and action. Today we *know* that smart machines can't act alone, and thanks to MAS research we have much deeper models that explain why managing interdependent activity is so hard. Second, MAS research has actually built technologies that provide generalized solutions to hard distributed problems, such as:

- Tractably creating, monitoring, and adapting joint plans that are sensitive to resource and quality constraints.
- Deciding when, with whom, and what to communicate to meet collective goals
- Designing constraints on interactions so that agents' behaviors converge on collectively useful solutions that fall within desirable bounds.

MAS theories and technologies have already provided significant benefits even with partial adoption in commercial settings [9, 8]. In commercial setting, as Parunak said (2000) "*The question is no longer to prove whether agents can solve realistic problems.*" Two MAS application domains where adoption has been more than partial are /em systems modeling and /em entertainment. A major and quite evident successes of MAS in academia, business, and in government/policy arenas is the use of MAS for modeling social, economic, environmental, and governmental systems. Multi-agent models are proving to be critical and high-leverage underpinnings for policy discussions, negotiations, and science in many of these areas. They are critical partly because of the types of problems faced (complex, analytically difficult); partly due to MAS tools that provide easy entry and worked examples; and partly because of human-in-the-loop features of modern MAS models. Another area of major current impact for agents is in entertainment and games, including film production (e.g., realistic computer-generated scenes involving multiple agents), massive multiplayer games (MMPORGs), and the like.

Perhaps more importantly, the basic landscape of AI has changed as research into agent-based systems has grown. Today, the ultimate goal of artificial intelligence is widely seen as understanding how to build active, embodied, integrated agents that behave in socially-realistic ways (i.e., can integrate with people's activities, attend to social conventions, and operate under people's oversight), and that can reason and act collectively and in coordination with others. Research continues into component theories and technologies, such as learning, planning, memory, perception, and representation. But all of these are now seen as components of a larger, integrated picture. This new picture always accommodates social realism and always includes multiple participants, with different representations, values, and life histories—in short, diversity. The AI tide has turned strongly toward multi-agent systems, and this sea-change is itself a major impact of the MAS perspective on the science of AI and on many other sciences.

Nonetheless, several critical barriers seem to prevent the partial successes of MAS from growing into more complete acceptance, and offering even greater impacts.

Parallel and invisible MAS origins: It often seems that many more "agent-like" systems exist than agents researchers and communities give credit for (that's "*give* credit for," not "get credit for"). Emerging web services, sophisticated browsers and web (applet/application) servers [2, 11], many kinds of smart network technology [13], not to mention PDAs, cellphones, and so on often don't draw on explicit agent theories, languages, practices, and methods, but they *are* achieving many of the same ends envisioned by agent researchers and developers. Also, technologies and theories underpinning these non-agent agent systems are developed in universes parallel to, and only minimally overlapping, the explicitly named agents and multi-agents communities. The agent communities are caught in a bind between dissolving out into specific application areas or into parallel, existing, subareas of computing such as distributed computing or networks, or maintaining a separate identity and struggling for recognition in them. Is MAS any more than a minor twist on infrastructure technology? Does MAS provide fundamental enabling advances that cannot be made elsewhere? These are cases that must be made.

Solve the problems of others: An accomplished theater/TV director and teacher once said that the quickest route to a sustainable career in the entertainment industry was "make money for other people" [10]. If MAS technolgies principally address issues that are defined by MAS researchers, that appear in MAS-based applications, and whose visible payoffs accrue to actors in MAS communities, their ability to create feedback loops that introduce new problems and resources into MAS communities will be lost. Viable economic progress depends on profitable trade with an outside world. To grow, MAS experiments and tools have to address critical problems in the domains of others, that those others can't efficiently solve themselves.

Focus on fundamentals: Many issues that arise in multi-agent worlds are fundamental problems of distributed systems of information. Indeed, the prospect of impacting fundamentals is a principal motivation for many serious MAS researchers. Fundamental problems being fundamental, they appear in many guises in many fields—no field "owns" them. For instance, *coordination*, possibly the most ubiquitous long-term issue in MAS, is also a salient problem in economics, in ecology, in sociology, in distributed operating systems, in mechanics, and in biology to name just a few. To the extent that MAS technologies give good models and results for fundamental problems, the broader scientific and technological communities benefit, and both impact and awareness will grow. MAS researchers should reach for these connections.

Win cost-effectiveness and elegance races: When problems are fundamental and critical, they appear in many disciplines and many people address them. MAS researchers need to create approaches and solutions that are provably *more* elegant, insightful, cost-effective, and integratable than those appearing elsewhere—not just equal, but better. In some cases this has occurred. MAS researchers' computability and tractability analyses of equilibrium states in games and markets has led to new algorithm designs, where tractability wasn't yet a concern in other fields. In other areas such as programming languages and communication systems, MAS techniques haven't yet shown clear advantages of effectiveness and elegance. What are the make-or-break practical advantages of using a specialized agent communication language and infrastructure such as FIPA-ACL over CORBA, WSDL, SOAP, UDDI, HTTP, or a protocol specific to an application area such as Z39.50 or the OAI protocol for information services? MAS developers may not yet take the interdisciplinary race seriously enough.

Create identity for results, and exploit it: Product reputations grow through naming and branding. The lasting currency of MAS, like most science and technology, comes in two forms: surprising, entertaining insights or phenomena, and efficiency- or effectiveness-producing techniques. These are what need naming and branding. Some MAS brands are loosely established within the field, but not usually propagated outward; examples include the *Cohen-Levesque joint intentions framework*, the *PGP* and *GPGP planning protocols*, and the *adjustable autonomy* concept. But many more MAS ideas need clear identities that get propagated outside the MAS community.

Reinforcement politics: Most of the impact-enhancing strategies discussed above are rationalist: they imagine raising impact by linking MAS to better/faster/cheaper solutions to recognized, critical problems. In the 1970s, studies of computing adoption posed "reinforcement politics" (RP) as an alterative to economic or technical rationalism in the adoption of new computing models [3]. Under this RP model, computing arrangements are chosen that reinforce dominant biases and structures of control. There is currently little sociotechnical analysis or research for MAS; nonetheless, from the RP perspective, to the extent dominant industrial and commercial bureaucratic models become decentralized and autonomic, MAS should gain impact. So the place to act would be in transforming bureacracies and ideologies, rather than selling MAS *per se* on rationalist grounds.

4. Where's the Agent? – Michael Luck

Michael Luck is a professsor of Computer Science who has worked in the area of agent-based computing for over 10 years. He was Director of AgentLink II, and is an Executive Director of AgentLink III, both tasked with promoting take-up and deployment of agent-based systems. He has instigated the development of several industry-facing activities and events to try to reach out to the broader IT community.

The rise of research into agent-based computing has not yet been shadowed by similar success in industry. In this section, I examine some of the reasons for the apparent lack of take-up of agent-based computing, and explore why recent, more general, developments in computing will likely provide the missing impetus, and why industry will need to move quickly to catch up.

A killer application? In their Scientific American article on the Semantic Web [1], Hendler, Berners-Lee and Lassila rightly oppose the search for a killer application for the Semantic Web. The Semantic Web *is* a killer application *itself*, they argue. In the same way, agent technologies do not need a killer application. First, the diversity of agent technologies makes it hard to identify a single (or even a set) of such applications, since each represents only specific instances of techniques or paradigms, not generality. Second, agents provide a way to understand and conceptualise a system, and to manage complexity in a very general sense, rather like object-oriented computing. Indeed, some argue that agents themselves create problems, not solutions, because modelling systems as agent collections requires a whole set of

specific new techniques for managing interactions, coordination, negotiation, and so on.

If agents do cause these problems, why should they be important? The answer lies in the increasingly dynamic, decentralised and open nature of modern computing and infrastructure. New paradigms for developing distributed systems such as Web Services and peer-to-peer computing possess agent-like characteristics, regardless of whether designers articulate them as agent-based systems. When most systems are closed systems the commercial case for agents may be debatable. But visions for the future of computing (including Autonomic Computing, Ambient Intelligence, Grid Computing, Semantic Web, etc) all take openness for granted, and eventually the two most important attributes motivating the agent approach, dynamism and openness, will prevail.

My point is simple: Next-generation computing systems will exhibit the same characteristics that motivate the agent approach. Agent technologies (rather than the agent paradigm itself, which may or may not gain traction) are needed to realise these kinds of system, and are beginning to gain increased visibility across a range of domains.

Where are the case studies? The current problem is how to focus the commercial sector on these emerging areas and the available solutions. From my own interactions with a range of commercial organizations, both inside and outside the agent community, it appears that case-studies with real articulated business benefit will best encourage deployment of agent systems. However, we don't yet possess a sufficient range of successful commercial deployment examples for agent systems. Demonstrators, proof-of-concept systems and prototypes are all valuable, but they alone don't overcome naturally risk-averse commercial attitudes toward new technologies.

Too few agent companies and commercial agent systems exemplify full deployment with clear business benefit. Many successful deployed systems occupy *niches*, making them limited drivers for the full range of agent technologies. For example, in the US, Nutech (which recently acquired Bios Group) has gained recognition for work on optimisation problems using agent-based simulations. Similarly, Magenta in the UK has developed systems for routing oil tanker fleets. Other high visibility examples include the use of agents in *The Two Towers* film, and on the factory floor at DaimlerChrysler, but these demonstrate value in specific sub-areas, not across the range of agent-based computing.

The answer includes developing an application case study catalogue spanning the range of agent technologies, that shows business benefit for a spectrum of applications. To do this, AgentLink (www.agentlink.org) is enumerating specific applications in different domains, across a varety of techniques, in different geographical locations. As ever, though, more is needed.

Take-up, drivers, and opportunities Why have agent *systems* achieved such limited commercial visibility? Though real benefits have been delivered, business case focus has often been limited and market entry has been sometimes premature. Until recent Web Services developments, for example, many agent solutions required a whole new infrastructure, which required too heavy a cost in disruption and enterprise re-engineering. The Web Services substrate provides two key benefits for the ready deployment of agents. First, it enncourages designers and users to naturally view systems as independent interacting entities, entirely consistent with the agent view. Second, effective, standard infrastructures (like Web Services) facilitate more sophisticated agent techniques such as coordination and negotiation.

The Semantic Web (SW), Grid Computing (GC), and other *drivers* produce similar effects. The credibility of the World Wide Web consortium backs SW, and natural developments in internal company management efforts increase its industrial currency. Companies clearly see SW as an important future step for large-scale information and knowledge management, and bringing relevant technologies such as agent-based computing into focus too. Less mature, GC is also visible: Sun recently launched a *pay-as-you-go* service whereby customers rent computing services hourly. This kind of infrastructure and utility computing will foster agent-based application development, building virtual organisations that coordinate and deliver composite services, in ways not otherwise possible. In both cases, agents (individually or through the realisation of virtual organisations) release these driving visions' power [4, 1].

In summary, the rise of Semantic, Service, and Grid technologies will enable new dynamic and open application systems. Such systems demand new technologies, and agent-based computing provides many relevant solutions. The benefits include new processes heretofore unrealizable, as well as increased flexibility for existing processes. Companies such as IBM focus on areas like autonomic computing for just these reasons: a step change in computing will only arrive by facilitating autonomous components that dynamically respond to open and changing circumstances. Managing the resulting complexity will require agent technologies.

5. Commercial settings: What is needed to get traction? – Jim Odell

James J. Odell is a consultant, writer, and educator in the areas of agent-based and object-oriented systems, methodology, business process management (BPM), and complex adaptive systems. Throughout most of his thirty-year career, Mr. Odell has been heavily involved in developing better methods to understand, communicate, and manage system requirements. He was one of the early innovators of information engineering methodologies. Recently, he participated in

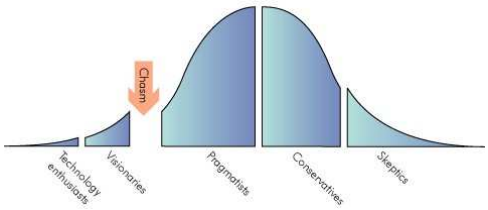


Figure 2. Technology Adoption Life Cycle[7]

the development of the UML and UML 2.0, and remains co-chair of the OMG's Object Analysis and Design Task Force. Most recently, Mr. Odell is involved in agent-based and complex systems and their application to business systems (including an agent-based UML, called AUML). He is on the board of FIPA and the chair of its Modeling Technical Committee, as well as the chair of the OMG's Agents Special Interest Group. His consulting clients represent many business sectors and include major companies, such as Netscape, Amazon.com, DHL, Oracle, IBM, Hewlett-Packard, and others, spanning 17 countries. Most recently, he has joined Agentis Software, where he directs the methodology and modeling efforts.

Throughout most of my thirty-five year career, developing commercially successful software systems has been a primary activity. Over that period, software development has gone through a number of technology life cycles, including relational, object oriented (OO), and now agent-based technologies. While these approaches differ, their adoption life cycle is the same. In Crossing the Chasm, Geoffrey Moore [7] looks at the challenge of marketing high technology and distinguishes between successive groups of adopters (Fig. 1). Agent technology is currently in the chasm period of adoption: the time gap between the visionaries and the pragmatists (early adopters). Pragmatists are the solid citizens who avoid the risks of pioneering but readily see the advantages of tested technologies. They are the vanguard of a mass market.

An agent approach can only get traction if it provides significant advantage with a reasonable investment return. But how can agent technology foster that competitive edge cost effectively? For early-adopter companies, agents must:

- Enable a solution that can not be done practically using any other kind of approach.
- Promote better developer productivity.
- Leverage existing technology.

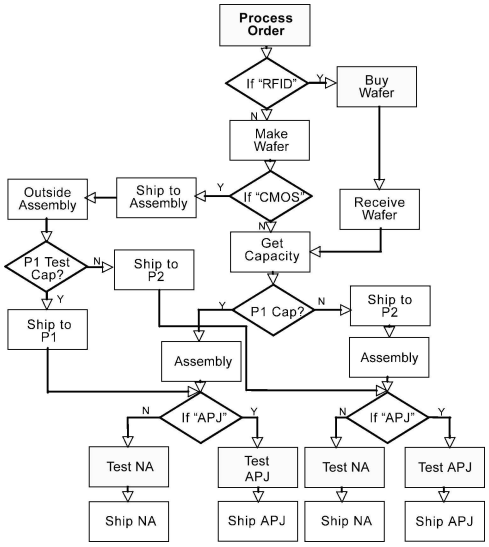
Enabling a solution. Agent enthusiasts and visionaries suggest that almost every application is appropriate for agent technology. But let's be honest: other options exist. Any claim that agent solutions are unique is an exaggeration. The IT pragmatist knows that a top-down, centralized OO approach to business process management (BPM) is possible, because most companies now use it. However, many BPM practitioners realize that processing bottlenecks and scalability problems plague these conventional implementations. In contrast, agent-based decentralized approaches enable a more efficient solution. To cross the chasm, the IT pragmatist needs to understand which applications can most benefit from little centralized control. BPM is just one compelling application in which early adopters are employing agent technology. Others include order processing, item tracking, equity trading, and supply chains, for example. As more of these applications successfully employ agents, commercial IT enterprises will more easily accept them.

Agents' processing autonomy is also catching early adopters' attention. With conventional approaches, systems must explicitly micromanage every decision (Fig. 2a). However, BDI-style agents provided by companies such as AOS and Agentis can dynamically choose and execute their own plans based on changing environment states (Fig. 2b). Goal-based applications that require contextual processing increase the effectiveness of agent-based approaches over conventional ones. Common application areas here include insurance policy rating, pricing, and item tracking.

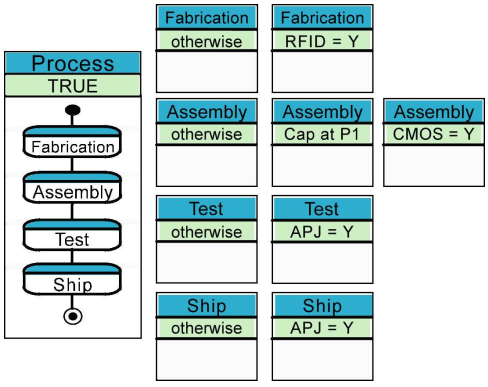
Promote better productivity: Pragmatic IT managers will estimate the potential development productivity of agent technology by examining items such as methodology, modeling notation, and design effort. Without an accepted methodology and modeling language, they won't cross the technology chasm. Even *with* these, developer-retraining costs become a consideration. The greater the difference between agent and conventional approaches, the greater the perceived cost and the lower the pragmatist's adoption probability. To succeed, the agent approach and convention must not differ greatly. Based on current market sentiment, agent extensions to UML, RUP, and MDA are being proposed, developed, and accepted by many early-adopter IT departments. While this may distress some agent researchers, agent technologies won't be accepted without it.

Furthermore, any higher cost for agent-based design and implementation effort over the conventional will impede chasm-crossing. Without productivity-enhancing tools, most pragmatists will never adopt an agent approach. Happily, commercial-grade agent tools are just now appearing and leading-edge IT departments are authorizing expenditures for them.

Leverage existing technology: Totally new and completely standalone applications are scarce. Commercial IT systems, by their nature, extend existing systems and databases. "New systems" are rarely new. They rely on, reuse, or "wrap" legacy code. Therefore, agents must interface with many other technologies. Resulting applications will combine technologies: agent, object, rela-



(a) Explicit Specifi cation



(b) Plan Spec. For Implicit Execution

Figure 3. Explicit Versus Implicit Approaches

tional, component-based software packages, and so on. Finally, agent developers need to integrate existing standards and, where appropriate, their extensions. Current standards emanate from the OMG, W3C, and Oasis.

6. Agent Business Case – Todd Carrico

Todd Carrico is the President of Cougaar software and a former DARPA program manager. Cougaar Software is a fast-growing startup company focused on commercializing the Cognitive Agent Architecture (Cougaar) technology , the result of 7+ years of DARPA investment in applied intelligent agent development performed by a world-class team of contractors from across industry and academia.

Like many companies struggling to make agent technology real in the marketplace, we are often faced with constructing an effective 'business case' for agents. In this brief section, I thought I would share some insights from our experience in building such arguments with our customers in hopes that it might help others to advance the adoption and legitimization of agent technology.

First, technologists must realize that the business world does not care about your technology. There may be some early adopter visionaries who are champions for the sake of the technology, but this is a small and specialized group. Business decision makers want solutions to hard problems in the form of operational capabilities at price points and on performance envelopes comparable to or better than their alternatives. The question we should be asking ourselves is 'What problems can agents solve exceptionally well that other technologies cannot?'

Unfortunately the truthful answer to that question is that there is nothing, in the strictest sense, that cannot be done with any other architectures. This unfortunate fact is frequently pointed out by the IT groups of potential customers. Since it is understood that any program can be reduced to a Universal Turing Machine[12], this simple truth causes us no end of grief. The more discriminating question might be 'What problems can agents solve more naturally then other technologies?'

These questions, from our experience, yield a few interesting answers that can serve as the foundation of a business case. Your mileage may vary, but I recommend a similar exercise to construct a business case for your potential customers to convince a potential customer of the virtues of your technology.

The Value of Situated Reasoning. Business people understand the value of agility, and know too well the lack of agility provided by today's Enterprise Resource Planning (ERP) systems. A strong business case can be built around the value of having agent-based planning and reasoning that can situationally aware. The subtle ability to adjust plans, processes and behaviors in response to the current state of the world affords a number of efficiencies and avoids several common problems. It ensures the plan is efficient with respect to the operational parameters and available resources, while attempting to work around equipment failures, resource shortages and facility constraints. Factors like weather can effectively be represented in the situation and reasoned about through agents more naturally then can legacy ERP systems. Whenever resources can be used more flexibly and appropriately in the execution of business operations, real value is created.

Capture Knowledge in a Reusable Form. A problem most companies face is how to capture and share knowledge across the enterprise. Typically agent systems can capture knowledge in a number of ways, such as rules, workflow, ob-

ject models, ontologies and domain behaviors. The explicit capture of this knowledge, especially those forms that can be produced by the domain operator, is typically the foundation of an agent application. A subtle but critical reality is that today, far too much of what makes a business effective is the result of human leadership. Two stores, identical in every physical way, can have two dramatically different profit lines due solely to the subtle differences in how the stores operate. Successful companies go to great lengths to understand what makes one store successful and impart that through policy and training to the other stores. Through agent technology, knowledge can be captured, utilized, evaluated and 'good' knowledge shared across the enterprise. The capture and reuse of knowledge provides significant business value to the enterprise.

Utilizing Humans as Decision Makers. With the complexity of today's information space, people tend to spend a great deal of time performing information gathering and processing, and less time analyzing and deliberating decisions. Agent technology is extremely well suited to providing automation of complex information processing activities, even when operating over vast amounts of data, on behalf of the user. As agents are capable of taking up more of the information gathering and processing burden, it frees humans to focus on what they do best, and agents do poorly - drawing insightful conclusions from which to base a decision. To show the expense and inefficiency in having people perform routine information activities versus agile and adaptive agent systems automating the same activities under user direction demonstrates real business value over conventional infrastructures.

These are three areas we have found to be good toeholds on which to form a solid business case. While these are by no means sufficient to justify a business investment in agent technology, they are often sufficient to motivate further dialog and exploration. The key to building a good business case for agents is to make it about the problems that can be solved and the value those solutions provide, not about the technology being employed. The technology is part of the 'how' of the solution, not the 'what'. I hope these simple points on building an effective business case for agents serves you well. As a community, we need to do a better job on helping our business benefactors to understand the value and potential of our technology, and in doing so we increase our opportunities for application, adoption and support for future research.

7. Conclusion

It is interesting, given the diversity of commentary on developing *impact for agents*, that common themes appear in some/all of the sections. In particular, the issue of applications or problems on which to apply agent technologies repeats with high frequency. One consistent message is *solve a problem* in contrast to focusing on general purpose technologies for which there is no direct need or pull. Note that a given problem may be an instance of a class of problems so that a technology developed to solve the class may be directly useful on many different instances. The question then is how broad or general a given approach should be – determining that balance is dependent on the technologies and your impact goals.

Developing impact for agents is clearly a much larger potential topic-of-conversation and a conversation that the community needs to have on a regular basis. The ideas contained in this paper are articulated by us but in some sense the community as a whole has contributed to them. It is our hope that these ideas assist in the sustained growth of agent research and in connecting-the-dots so that our research pays the dividends that we believe it can pay.

References

[1] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientifi c American*, May 2001.

[2] Adam Bosworth. Microsoft's vision for xml. In *SGML/XML Wurope '98*, 1998. http://www.infoloom.com/gcaconfs/WEB/paris98/bosworth.HTM.

[3] J. N. Danziger, W.H. Dutton, R. Kling, and K.L. Kraemer. *Computers and Politics*. New York: Columbia University Press, 1982.

[4] I. Foster, N. R. Jennings, and C. Kesselman. Brain meets brawn: Why grid and agents need each other. In *Proc. 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems*, New York, USA, 2004.

[5] Les Gasser. Private conversation with Tom Wagner, 2004.

[6] Frederick Hayes-Roth. Towards a framework for distributed ai. *SIGART Newsletter*, 1980.

[7] Geoffrey Moore. *Crossing the Chasm*. HarperCollins, NY, 1991.

[8] H. Van Parunak. Practical and industrial applications of agent-based systems. Technical Report, Altarum, 1998. Online at: http://agents.umbc.edu/papers/apps98.pdf.

[9] H. Van Parunak. A practitioners' review of industrial agent applications. *Autonomous Agents and Multi-Agent Systems*, 3(4), 2000.

[10] Don Richardson. Private communication with L. Gasser, 1985.

[11] Adam Rifkin. Relax, everything is deeply intertwingled: Weblications. ifi nd-karma.typepad.com/relax/2004/12/weblications.html.

[12] A.M. Turing. Computability and lambda defi nability. *The Journal of Symbolic Logic*, 2, 1937.

[13] Mark Weiser and John Seely Brown. The coming age of calm technology. *Xerox PARC*, 1996. http://www.ubiq.com/hypertext/weiser/acmfuture2endnote.htm.