Mass Collaboration Problem Solving: A New Approach to Wicked Problems

Andrew Potter
Sentar, Inc.
andrew.potter@sentar.com

Melody McClure
Sentar, Inc.
melody.mcclure@sentar.com

Ken Sellers
Sentar, Inc.
ken.sellers@sentar.com

ABSTRACT

A system that would harness the brainpower of vast numbers of humans and orchestrate their efforts could be used to solve hard problems that are beyond the reach of computational methods. This paper describes such a system. This system will use a recursive problem solving life cycle model based on a continuously evolving distributed supply chain infrastructure. The problems to be addressed using this system are "wicked"—that is, they have no definitive formulation, no absolute answer, and objective definition of equity. The problem solving infrastructure must continuously evolve and adapt, as does the community of stakeholders that enact it, concomitant with their developing understanding of the problem and its solutions. The system defined here provides an effective, repeatable mechanism for using mass collaboration to address wicked problems. This paper discusses the motivation, system architecture, and future directions.

KEYWORDS: Crowdsourcing, Distributed Systems, Mass Collaboration, Problem Solving, Social Networks.

1. INTRODUCTION

Mass collaboration problem solving is an idea whose time has come. This has been brought about by an unprecedented convergence of technologies and social phenomena that have more fully accomplished the global nature of the Internet. Richly featured Web 2.0 technologies have enabled the development of distributed collaboration tools like weblogs, wikis, and multi-media discussions that are highly interactive, easy to use, and easy to implement. That large numbers of people are eager to participate in mass collaborative activities has been demonstrated through the success of a variety of social networking phenomena such as LinkedIn, Facebook, Flickr, and YouTube. That, given the right circumstances, large numbers of people are eager to work quite hard to collectively solve difficult problems is proven by the emergence and sustainability of the open software movement. The development of scalable distributed service-oriented architectures, grid computing, and multi-agent technologies has made it possible to design systems which can orchestrate and coalesce the efforts of large numbers of participants, whether they be computers or humans.

And yet we have only just begun to realize the possibilities. Little is as yet known about the full range of problems that could be solved using mass collaboration. If the attestations of the popular press are any indication, the possibilities are unbounded. For example, according to Tapscott and Williams [1], today's organizations are redefining their business models to utilize complex ecosystems of collaborative activity that extend well beyond the boundaries of the organization or its affiliates; collaborative communities are joining open market and hierarchical firms as an alternative competitive strategy and way of organizing work; traditional in-house R&D organizations are being eliminated or reduced in favor of reliance on the open community for new ideas and innovations; increasingly customers are being treated as extensions of the organization, giving customers an active decision-making role in designing products; the relationships between the public foundations and private enterprise are being reexamined, with implications for intellectual property that have yet to be fully grasped; and open platform business models are arising to capitalize on the emergence of collaborative communities, continuous innovation, customer integration, and evolving intellectual property issues.

Even if, as some have argued, such claims are exaggerated [2], mass collaboration is clearly an emergent trend, and we need to be prepared to reap the benefits. However, there has as yet emerged no clear process for managing mass collaboration in an effective, repeatable way. For all the high profile successes, there are numerous lesser known failures. For example, open source startups such as SugarCRM, Alfresco, Jasper,

Pentaho, and ActiveGrid were expected to bring open source software into the mainstream [3], but their efforts seem to have stagnated. Instead, it is the highly successful open source projects, such as Linux, MySQL, and Jboss that receive publicity.

And so the question arises, how can we take mass collaboration problem solving to the next level? What technologies are needed? What kinds of organizations will be required to manage mass collaboration problem solving? While there has been considerable discussion about the notion that mass collaboration is a self-organizing phenomenon, if we take this idea seriously, it is necessary to introduce mechanisms that will reduce risks, assure some efficiency, and promise a pay-off downstream.

Like the public-resource computing systems which use the spare processing time of millions of computers in homes and offices around the world to address computationally intensive problems, the Mass Collaboration Problem Solver would utilize the brainpower of large numbers of humans and orchestrate their individual efforts to solve hard problems that are beyond the reach of purely computational methods. These are problems that are often considered insoluble, wicked, or too ill-defined to really solve. How can we secure our nation's borders? How can we foster democracy in developing nations? How do we feed a planet of 9.2 billion people? What are the social, economic, and military consequences of failing to do so? The usual approach to such problems is to appoint a special commission, send in a high profile envoy, or simply to hope for the best. But if it is reasonable to use the SETI public-resource computing system to search for extraterrestrial intelligence [4], then perhaps it is also reasonable to use mass collaboration to answer important questions closer to home.

2. A SUPPLY CHAIN APPROACH

A global supply chain is a highly complex network for managing a massively distributed system for producing, transforming, and moving products to customers. It is, by definition, collaborative. If we think of problems as demands and solutions as products that would satisfy those demands, it would seem possible to repurpose the concept of agile global supply chain networks to massively distributed networks for problem solving.

Consider the following example: The manufacture of a simple item such as a pencil requires a complex supply chain involving extraction and production processes from all over the world [5], as shown in Figure 1. The process is sufficiently complex that no single individual possesses all the skills and knowledge necessary to mass produce a pencil. In other words, collaboration is not only required, quite a lot of it is required. Obviously, a pencil is a relatively simple item compared with many other products routinely brought to market. If supply chain management technologies are able to address such production and marketing challenges, then perhaps a similar approach can be used to solve complex hard-to-define problems, such as securing national borders or managing global climate change. Figure 2 illustrates, in broad strokes, the diversity of elements that would constitute problem solving supply chain for global climate change. Reaching political, economic, and national security consensus on problems such as this requires a dynamic workflow that reflects these interdependencies. So many are the stakeholders, so diverse are their interests, and so specialized is their expertise, a supply chain infrastructure would be an essential element in going forward effectively and efficiently. By this means, problem decomposition entails both vertical and horizontal integration.

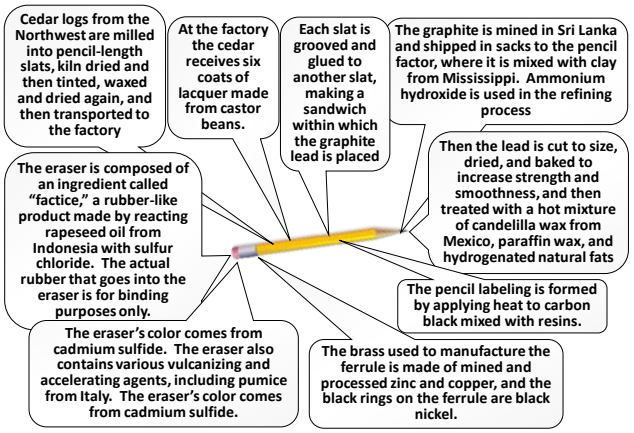

Figure 1. Even the manufacture of a pencil requires a global supply chain [5]

In traditional supply chain management, decision making occurs in four areas: 1) location, 2) production, 3) inventory, and 4) distribution [6]. There are analogs for each of these in the distributed problem solving network. *Location* refers to the decomposition of the problem. For our purposes, locations are logical rather than geographical. As noted above, problem decomposition is both horizontal and vertical. Horizontal decomposition refers the functional breakdown of problems into sub-problems; vertical decomposition refers to the allocation of sub-problems among stakeholders. *Production* decisions determine the paths through which information flows through the supply chain. These decisions define the decomposition of the problem into solution spaces which correspond to problem solving processes within the logistical workflow. *Inventories* are of two kinds. The

first refers to the availability of collaborators to support each step of the supply chain; for any sub-problem within the overall process, it is necessary to determine which collaborators will be involved and ensure that they will be ready when needed. The second kind of inventory is *Information*. Although information is sometimes treated as a non-depletable resource, before it can be disseminated, it must first be created. Thus the information inventory is critical to the efficiency of the problem solving supply chain. *Distribution* of information inventories is critical to the flow of information through the supply chain. In a problem solving supply chain, information flow is both a management and content function. That is, information is both the goods supplied through the supply chain, and it is the workflow structure that manages the flow. Management distribution decisions determine selective dissemination and control, scheduling, processing, overload mitigation, conflict resolution, and security of content. By defining mass collaboration problem solving as a distributed supply chain process, we can design systems that address requirements for scalability, decomposition, interdependency, evolvability, selectivity, reusability, and measurability necessary for an effective and efficient outcome.

3. AN EVOLVABLE PROBLEM SOLVING SUPPLY CHAIN

The system will constitute a practical distributed human problem-solving system, using collaborative human brainpower to tackle problems that are too big and complex to be handled by conventional organizations and too ill-defined to be solved effectively through computational methods alone.

The supply chain infrastructure will enable collaborators to define the solution space in terms of evolvable workflow graphs. The problem decomposition will be used as input to this activity. The resulting model is then deployed into the collaboration environment wherein collaborators can match their capabilities with advertised requirements of the workflow nodes. Workflow logic will be used to enable work to proceed on a given node as its supply chain antecedents are resolved. Ongoing reviews will be incorporated as collaborative tasks to ensure that the graph accurately reflects current understanding of the problem and its prospective solution.

Just as problems may be decomposed into smaller more manageable pieces, composable workflows can be created bottom-up from the problem elements and integrated to form the overall solution supply chain. In this way, supply chain construction may be undertaken as a collaborative activity, just as any other problem solving activity within the overall infrastructure. In addition, this supply chain mapping of the problem may be used to aid sub-problem synthesis. This would occur progressively; as each workflow process is completed its output solution would be synthesized by the downstream process node.

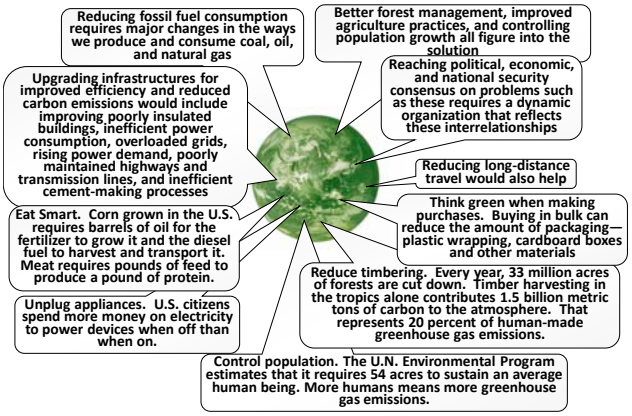

Figure 2: The processes required to solve more complex problems, such as managing global climate change, are even more challenging [7]

Mass collaborative activities are often characterized as bottom-up self-organizing activities [8, 9]. However, taking a look at the technology used to host these activities tells another story. Anyone contributing to Wikipedia will find themselves obliged to work within a highly structured authoring environment, subject to well defined technical constraints and social norms. The same can be said for any of the social networking environments, such as Facebook, YouTube, or MySpace. Open source projects must standardize around a minimal set of tools for version control, bug tracking, and communication channels, and increasingly the development environment itself is subject to standardization. Users contribute content, but always within a well defined information technology architecture. For activities that entail a specifiable life cycle, it is necessary that the architecture and tools support the successive phases of problem solving.

4. THE PROBLEM-SOLVING LIFE CYCLE

The problem solving life cycle is recursive. As problems are decomposed into sub-problems, each of these sub-problems takes on a life of its own, eventually feeding its results back up to the upper level problem. It is necessary that the system be able to handle these nested interrelationships. As shown in Figure 3, the general problem solving life cycle begins when a problem is identified. As the problem is being identified, it will also be assessed to determine whether it is suitable for a mass collaboration approach. Concomitant with the problem

selection process is the Research Review, which will determine what work has already been accomplished, or is in progress, relating to the identified problem. Depending on the outcome of this review, it may be appropriate to cease activity, to merge with an ongoing project, or to proceed with the problem solving cycle. If the decision is made to proceed with the problem solving cycle, the next step will begin detailed planning and recruiting for the project. Recruitment will continue throughout the cycle, matching problem solvers with the needs of the problem. This may include forming strategic alliances with co-interested organizations and informal communities.

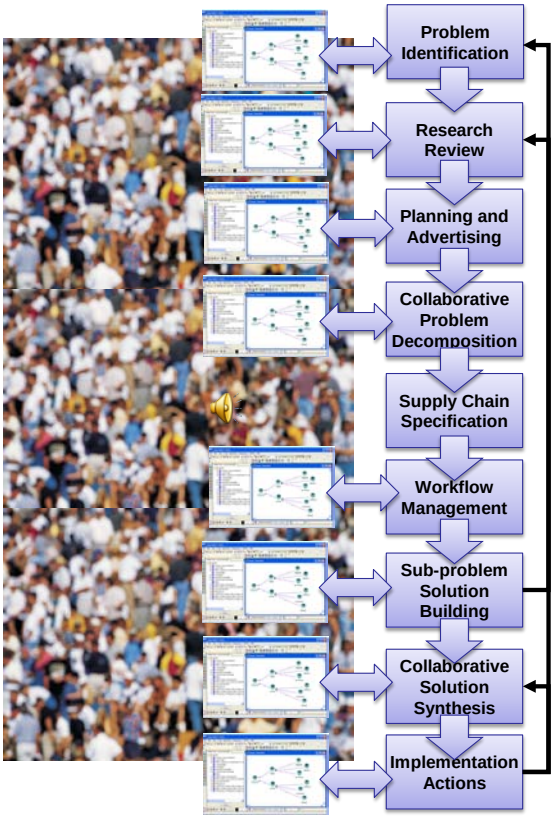

Figure 3: Problem Solving Recursive Life Cycle

As planning proceeds, it will phase into problem decomposition, breaking the problem down into sub-problems. This will be undertaken collaboratively and iteratively, enlisting the support of the subject matter communities active in the problem domain and working in tandem with ongoing effort to recruit contributors. The next step, called supply chain definition, is a detailed planning activity, in which the sub-problem dependencies are identified and a workflow is constructed. Once the supply chain workflow is populated, workflow management will begin executing it, matching problem requests with registered capabilities and dispatching problems to problem solver teams and individuals. Sub-problem resolution will be approached recursively, with each sub-problem following an instance of the problem-solving life cycle. As the results of these activities come in, they will undergo solution synthesis. Solution synthesis is also a collaborative activity, in which sub-problem solutions are assessed and integrated into a solution model.

What does the system produce? For complex problems the problem solving life cycle will be performed over five or more iterations, beginning with a conceptual phase in which basic insights and innovations will be developed, and followed by a theoretical investigation to understand the problem and potential solutions in greater detail, reduction to practice for prototype development, and then solution building in which the solution is formulated [10]. But the process cannot be permitted to stop here. To be of value, the technology cannot simply solve problems and then leave it up to the stakeholders to implement the solutions. After all, for real-world problems, the implementation is part of the solution. Thus the details for how the solution will be implemented are imbedded in the solution itself. The problem solving process continuously evolves and adapts, as does the community of stakeholders that enact it, along with their understanding of the problem and its solutions. In short, *the stakeholders are the solution*.

We believe that there will be many variations on this general approach. Some problems may require that certain portions of the effort be classified. For example, if the system were used to address the problem of securing the nation's borders, participation from interested citizens, particularly those living in areas near borders, as well as the business community, economists, anthropologists, law enforcement, and environmentalists, could be used to gain insights and alternatives for possible solutions. But other aspects of the problem solving effort could involve the handling of classified, export controlled, or other sensitive information, e.g. designs for advanced monitoring equipment or sensitive diplomatic discussions with representatives of foreign countries. For this reason, not all decomposition branches within a problem solving workflow will be visible, nor will they be managed on the same computer networks. Such tradeoffs between openness and security are essential to the ability of the system to address a full range of significant problems.

5. MOTIVATING MASS COLLABORATION

Motivation to participate and perform occurs at two levels—the individual and the organizational, and while their interests in the collaborative ecosystem may be

aligned, they remain separate. In their study of programmer motivation in open software projects, Lakhani and Wolf [11] found that enjoyment-based intrinsic motivation is the most pervasive motivation, with user need for the software, intellectual stimulation, and improving skills as other motivators for project participation. Nevertheless, as Fogel observed, "groups engaged in cooperative activities must evolve norms of behavior such that status is acquired and kept through actions that help the group's goals" [12].

Individual motivation is only a small part of what is needed for effective mass collaboration. *The economics of collaboration are based not just on individual motivation, but on the interests of the individual's organizational sponsor in the fruits of the collaboration.* For example IBM is willing to assign a team of programmers to support Linux, not because it believes in community service, but because it expects to get a superior product at lower costs as a result. This "volunteer" collaborative community becomes a shared resource among separate self-interested organizations—and these organizations may have few ties to one another except their common interest in the products of the collaborative community. To this extent, a qualification for what makes a good collaborative problem is one in which diverse organizations have a stake in the shared outcome. As Tapscott observes, the "free" software of open source is an integral part of a much larger multi-billion dollar ecosystem [1]. The ability of a business to provide products and services depends on the reliability of its supply chain. In a traditional supply chain, the contributing entities are assumed to be acting out of self-interest [13], and the supply chain infrastructure must account for this. Without appropriate checks and balances, there are a number of ways in which self-interest can lead to supply-chain breakdown. This interdependency can have undesirable effects; for example, an organization within the supply chain may emerge as a competitive threat to other members of the chain [14, 15]. Thus it is not only important to provide incentives, but it is necessary cultivate loyalty, manage trust, and provide redundancy, as in any other organizational federation.

6. MASS COLLABORATION PROBLEM SOLVER DESIGN

The recursive, evolving nature of problem solving is reflected in the supply chain approach to organizing information flow. As shown in Figure 4, the underlying agent architecture [16-18] will use a goal-driven mediated model, enabling it to allocate tasks to problem-solvers based on their registered capabilities. Working cooperatively with the workflow manager and other core agents, the mediator agent marshals capabilities as they are requested by the workflow and other agents, it maintains an agenda of pending allocated activities and their interdependencies, and it maintains a data store reflecting progressive work towards problem resolution. As tasks are created, problem-solver agents are allocated. These problem-solver agents will provide a principal interaction point for information storage and retrieval for the human problem solver.

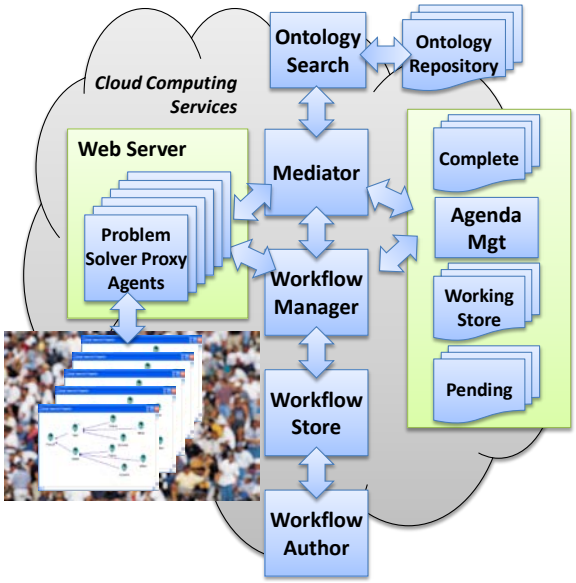

Figure 4. Basic Problem-Solver Architecture

The system architecture handles the complex logistical message management necessary to address complex problems using a workflow. This approach, as detailed in the following subsections, enables workflow authors to specify problem solving logistics at an ontological level, while the underlying agent architecture handles marshalling of specific supply chain information flows.

6.1 Agent Infrastructure

As shown in Figure 5, the lower level agent runtime infrastructure consists of an Environment, Environment Configuration, Controller, Behavior, and Agent Configuration. The *Environment* implements operations for instantiating and managing a collection of agents and communicating with remote environments. During initialization, the Environment accesses *Environment Configuration* to identify startup agent configuration, remote environment interfaces, and other initialization parameters. The *Controller* supports the environment, performing operations for sending and receiving messages. Each agent is associated with a specific *Behavior* and an *Agent Configuration*. The Behavior contains the application specific functionality of an agent, and the Agent Configuration defines the agent functionality and is loaded by a behavior during initialization.

An Environment can interact with other environments using *Channel Agents*. Channel agents establish capabilities-based pathways through federations of systems. When a registered capability resides with a channel agent, the search for a solution follows its registered capabilities into another system. A channel agent looks and behaves like any other agent. Its registered capabilities implement the interface of one environment to another. Once the channel agent has registered these capabilities, any messages dispatched to it are forwarded to its counterpart in the remote environment.

A key agent in the Environment is the *Mediator Service*. The Mediator implements cooperative reasoning among the other agents participating in the environment. When an agent joins an environment, it declares its capabilities to the mediator, and the Mediator enters the capabilities in its registry. Registration identifies the services the agent can respond to and the domain ontology within which it can provide its response services. Any agent can initiate a request. The mediator uses an agenda to keep track of what it is doing and for whom, and it maintains a context of asserted propositions associated with the agenda. This may result in deeply nested contexts, depending on the complexity of the problem and the number of agent interactions necessary to satisfy it. Thus the agents act as proxies for the human collaborators, representing their abilities and their interests. The Mediator provides automated management for the chain of interactions required to resolve a problem.

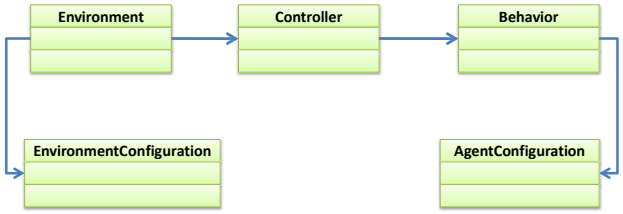

Figure 5: Runtime Infrastructure

6.2 Workflow Manager Design

The Workflow Manager is an agent like any other. It consists of a workflow behavior and processing elements required to implement a workflow. Figure 6 shows the components of the Workflow Manager. The workflow *Behavior* is responsible for processing a workflow. The behavior maintains a context for each active workflow instance, including workflow paths created by forks. When a message is received the workflow locates the associated context and allows the workflow to continue processing. If an active context is not found then a new context (instance) is created and the workflow begins at the workflow's start node. The workflow *Context* provides a way for the workflow to be put "on hold" while it is waiting for a response to some activity. The context also provides a mechanism for multiple workflow instances to be active simultaneously. An active context has a unique identifier and contains a collection of conclusions for completed activities. The context is initialized with a reference to the behavior's agent controller. The *Workflow Node* defines an interface that all workflow nodes support. The interface provides a mechanism for processing workflow nodes without having to know the details or interface of individual node types.

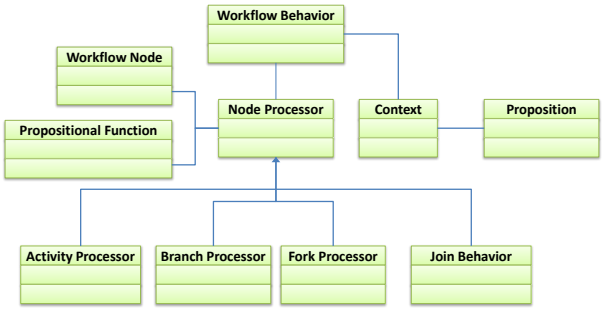

Figure 6: Workflow Manager Components

Like other agents in the system, the Workflow Manager registers its capabilities with the Mediator, and when called upon to fulfill those capabilities, it will execute the appropriate workflow. The workflow may generate its own requests and post them to the mediator, which will dispatch them to other agents registered with the system. If these agents are Proxy Problem Solver Agents, then the request is effectively dispatch to a member of the collaborative team. However, the agent could also perform some automated function, such as executing an algorithm or checking a sensor.

A Workflow Editor, like the prototype shown in Figure 7, can be used to author workflows for use by the Workflow Manager. These workflows, once edited can be loaded by the manager as part of the agent configuration. The workflow editor can be used to define the steps and flow necessary to solve a problem, when explicit flow control is desired. We do not regard the current prototype as a final product, ready to use in a collaborative environment, but it is a step in that direction.

6.3 Problem Solving Life-Cycle as Multi-Agent Supply Chain

The agent infrastructure implements the problem solving life cycle. When an individual submits a new problem,

the system manages construction of the dynamic supply chain that will solve the problem. A new problem is submitted when an initial problem statement is submitted, via proxy agent, to the mediator. The problem statement is formal, consisting of an ontology and a problem specification. We expect that this initial ontology will be either rudimentary, or it will be based on an existing ontology. In either case we do not expect that these ontologies require the rigor necessary for full scale knowledge representation systems, but with sufficient granularity to enable the mediation and workflow processes. We expect that they will consist of both structured and unstructured information.

The Mediator will attempt to find a registered Problem Solving Proxy Agent or workflow with problem solving initialization capabilities matching those of the problem statement. Capabilities matching may be direct if there is a Problem Solving Proxy Agent or workflow registered that directly matches the problem specification. Otherwise the Mediator uses the Ontology Matching agent to attempt to find a proxy problem solver that will accept the problem. And if that still fails, the Mediator seeks a proxy that will accept unmatched problems. When there is a workflow with registered capabilities matching the problem, the Problem Specification is dispatched to a Workflow Agent that then executes the workflow. As the workflow executes, it will submit sub-problems to the Mediator the Mediator will attempt to solve them in the established manner. This will result in sub-problem dispatch to other proxy and workflow agents as needed, thus dynamically extending the supply chain vertically and horizontally until problem decomposition is complete.

When a Problem Solving Proxy Agent accepts a problem, it can either solve it or it can decompose it into smaller problems and solve some parts while initiating problem solving processes for the other parts. This is accomplished by creating a problem solving specification and submitting it to the mediator. These problem solving processes follow the same path as the original problem. When we say that a proxy agent solves a problem, generally what actually will occur is that a human will solve the problem, with the proxy agent acting in the human's behalf. The proxy provides the human-computer interface between the human and the problem solving mechanisms. From the human's perspective, these mechanisms are invisible. The human sees problem solving opportunities that match his or her capabilities as registered with the mediator, and the proxy provides the means for accepting problems, submitting sub-problems, submitting new problems, and submitting solutions. The proxy also provides the human with the ability to author and submit workflows. It is expected that these would be authored as canonical solutions to problems that tend to recur with minor variations. As the sub-problems are satisfied their results are fed back to the Proxy Agents of origin, where the human can synthesize them to create the higher level solution and submit it to the Mediator. This step completes the Sub-Problem Building activity shown earlier in
Figure **3**.

While it would be possible to limit the scope of human's access to the problem to the immediate problems waiting to be solved, we expect that for complex problems greater visibility into the collaborative problem-solving context will be useful. This would enable the human problem solver to see the horizontal decomposition of problems into sub-problems and the vertical allocation of sub-problems among other participants. Using this contextual representation the human can pick problems for assignment and communicate with other members of the ad hoc team.

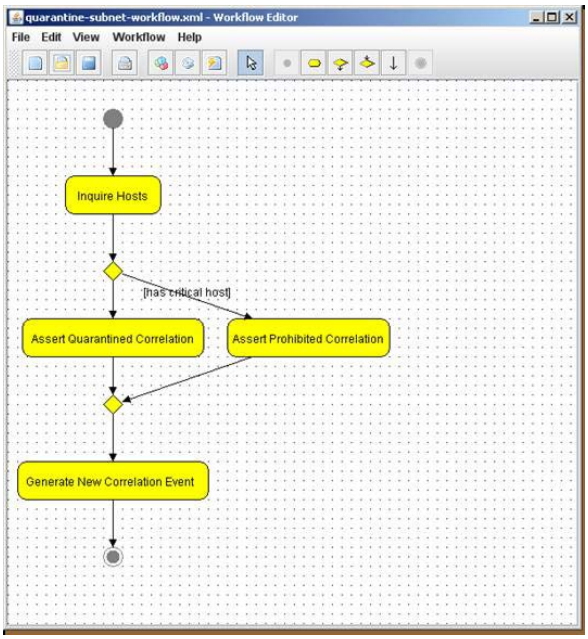

Figure 7: A Prototype Workflow Editor

The system will provide an open architecture for selecting the interaction and problem solving techniques for the particular problem. A wide variety of techniques have been shown to have value, such as dialogue mapping [19], object oriented collaborative analysis [20], anchored discussion [21-27], synchronous and asynchronous forums [28, 29], virtual worlds [30], scale-free virtual communities [31-33], simple task listings [10], and serious games [34-36]. Dialogue mapping seems particularly suitable since it incorporates a process for collaboratively constructing visual renderings of wicked problems (albeit for relatively small groups) and is

supported by a well defined iconography and conceptualization [19]. This mapping can be enriched using the supply chain metaphor, such that logical locations indicate the horizontal and vertical linkages among problem solvers; production decisions define the decomposition of the problem into solution spaces corresponding to problem solving processes and workflows; inventories define the available relevant human resources who may or may not be currently participating, as well as information resources that may enable problem resolution; and management distribution decisions that determine selective dissemination and control, scheduling, processing, overload mitigation, conflict resolution, and security of content.

7. MEASURING & VALIDATING MASS COLLABORATION

As we look into the possibilities for measuring and validating mass collaboration, we first encounter problems in identifying precisely what needs measuring. Some possibilities include the quality of the collaboration, the quality of individual contributions to the collaboration, the usability of the system, and the performance of the system in achieving its goals. For our near-term concerns, the more interesting of these possibilities concerns the latter, the ability of the system to achieve its goals—in other words, how well it performs in enabling large numbers of people to solve difficult problems.

The kinds of problems to be solved using mass collaboration require rethinking our assumptions about completeness, consistency, and correctness. One of the characteristics of wicked problems is that there is no stopping rule [37]. As Rittel remarked, when working on such problems, one does not stop working because the work is complete—one stops for outside reasons, such as running out of money, time, or patience. While that may have been true for urban planners circa 1973, in the mass collaboration context, there need not be such external constraints—successful open source projects such as Apache and Linux, or crowdsourcing initiatives such as Wikipedia undergo continuous improvement. So there are reasons why, as Kazman and Chen have observed, concepts such as completeness, consistency, and correctness are anathema to mass collaboration systems [38]. If the solutions to the problem are subject to continuous improvement, then the problems must be wicked—they have no definitive formulation, no absolute answer, no objective definition of equity, and no immediate and no ultimate test of a solution. Were this not the case, the problems would be amenable to more conventional treatment, and there might be no need to resort to mass collaborative approaches. Thus, as we go beyond initial laboratory prototypes and begin solving real problems, we must look to conditions of *sufficient correctness* and *continuous evaluation* for measuring and validating the ability of the system to solve problems. Continuous evaluation is built into the problem solving process, with loopbacks at several stages of the process, as discussed earlier in Section 4. These loopbacks may occur as a result of recursive problem decomposition, or they may be iterative, as the problem becomes better understood.

7. CONCLUSION

The approach described here constitutes a practical distributed human problem-solving system, using collaborative human brainpower to tackle problems that are too big and complex to be handled by conventional organizations and too ill-defined to be solved effectively through computational methods alone. To date, we have implemented a prototype agent runtime infrastructure, workflow agent and editor, and other supporting utilities. We have begun developing concepts for the proxy agent design based on dialog mapping. Using dialog mapping, collaborators would jointly construct an integrated collection of visual diagrams that capture all aspects of a problem, as exemplified in the Compendium system [19]. Whereas the Compendium system is intended for relative small groups, our approach scales up, in team size and in problem size.

Mass collaboration problem solving is an idea whose time has come. Shaw [39, 40] argued that while the gold standard for software systems has long been functional correctness and that for many systems, particularly non-critical systems, sufficient correctness would suffice; however, when it comes to wicked problems, sufficient correctness is the best we can hope for. Our best bet then is to develop technologies that will enable us to push our problem solving abilities to the upper limits of sufficiency.

REFERENCES

[1] D. Tapscott and A. D. Williams, WIKINOMICS: HOW MASS COLLABORATION CHANGES EVERYTHING. New York: Penguin, 2006.

[2] N. G. Carr, "The Ignorance of Crowds," *Strategy and Business*, vol. 47, pp. 36-41, 2007.

[3] J. Surowiecki, THE WISDOM OF CROWDS. New York: Anchor, 2005.

[4] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, "SETI@home: an experiment in public-resource computing," *Communications of the ACM*, vol. 45, pp. 56-61, 2002.

[5] L. E. Read, I, Pencil: My family tree as told to Leonard E. Read. Irvington-on-Hudson: Foundation for Economic Education, 1958.

[6] R. Ganeshan and T. P. Harrison, "An introduction to supply chain management," Pennsylvania State University, University Park, PA 1995. Available: http://lcm.csa.iisc.ernet.in/scm/supply_chain_intro.html.

[7] D. Biello, "Ten solutions for climate change: Ten possibilities for staving off catastrophic climate change," 2007. Available: http://www.scientificamerican.com/article.cfm?id=10-solutions-for-climate-change.

[8] J. H. Panchal, "Co-evolution of products & communities in mass collaborative product development – A computational exploration," *International Conference On Engineering Design (ICED'09), Stanford University, Stanford, CA, USA*, Stanford University, Stanford, CA, USA, 2009.

[9] S. Valverde, G. Theraulaz, J. Gautrais, V. Fourcassié, and R. V. Solé, "Self-organization patterns in wasp and open source communities," *IEEE Intelligent Systems*, vol. 21, pp. 36-40, 2006.

[10] Innocentive, "The complete innovation solution from the big idea to the final product," Innocentive, Inc. 2008.

[11] K. R. Lakhani and R. G. Wolf, "Why hackers do what they do: Understanding motivation and effort in free/open source software projects," *Perspectives on Free and Open Source Software*, J. Feller, B. Fitzgerald, S. Hissam, and K. R. Lakhani, Eds. Cambridge, MA: MIT Press, 2005.

[12] K. F. Fogel, *Producing open source software: How to run a successful free software project*. Sebastopol, California: O'Reilly Media, 2005.

[13] S. Haag, P. Baltzan, and A. Phillips, *Business driven technology*, 2nd ed. Boston, MA: McGraw-Hill/Irwin, 2006.

[14] M. E. Porter, "How competitive forces shape strategy," *Harvard Business Review*, vol. 57, pp. 137-145, 1979.

[15] M. E. Porter, *Competitive strategy*. New York: The Free Press, 1980.

[16] R. S. Daniel, P. A. Kiss, and J. A. Yalowitz, "KnoWeb: A knowledge web for large-scale, evolving distributed knowledge resources," *IEEE Information Technology Conference*, 1998, pp. 75-78.

[17] A. Potter and G. Streeter, "Work-centered services for the Semantic Web," *3rd International Symposium on Multi-Agent Systems, Large Complex Systems, and E-Businesses (MALCEB'2002)*, Erfurt/Thuringia, Germany, 2002.

[18] G. Streeter, A. Potter, and T. Flores, "A mediated architecture for multi-agent systems," *Seventeenth International Joint Conference on Artificial Intelligence: Workshop on E-Business and the Intelligent Web*, Seattle, WA, 2001, pp. 173-117.

[19] J. Conklin, DIALOGUE MAPPING: BUILDING SHARED UNDERSTANDING OF WICKED PROBLEMS. West Sussex, England: John Wiley & Sons, 2006.

[20] N. Avouris, A. Dimitracopoulou, and V. Komis, "On analysis of collaborative problem solving: An object-oriented approach," *Computers in Human Behavior*, vol. 19, pp. 147-167, 2003/3 2003.

[21] G. Abowd, M. Pimetel, B. Kerimbaev, Y. Ishiguro, and M. Guzdial, "Anchoring discussions in lecture: An approach to collaboratively extending classroom digital media.," in *CSCL '99: Proceedings of the 1999 Conference on Computer Support for Collaborative Learning*. Palo Alto, CA: International Society of the Learning Sciences, 1999, pp. 11-19.

[22] A. J. Bernheim Brush, D. Bargeron, J. Grudin, A. Borning, and A. Gupta, "Supporting interaction outside of class: Anchored discussion vs. discussion boards," *Proceedings of Computer-Supported Collaborative Learning (CSCL'2002)*. Boulder, CO: University of Colorado, 2002, pp. 425-434.

[23] N. Dwyer and D. D. Suthers, "A study of the foundations of artifact-mediated collaboration," in *Computer Supported Collaborative Learning 2005: The Next ten Years!* Mahwah, NJ: Lawrence Erlbaum Associates, 2005, pp. 135-144.

[24] D. Thurab-Nkhosi, "Anchored Instruction (AI) as a tool in online learning at the University of the West Indies, St. Augustine.," 2005.

[25] K. Severinson Eklundh and H. Rodriguez, "Coherence and interactivity in text-based group discussions around Web documents.," *37th Hawaii International Conference on System Sciences*, Island of Hawaii, 2004.

[26] D. Suthers and J. Xu, "Kukakuka: An online environment for artifact-centered discourse," *International WWW Conference*, Honolulu, Hawaii, 2002.

[27] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of online social networks," *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. Philadelphia, PA, USA: ACM, 2006, pp. 611-617.

[28] A. Potter, CONSTRUCTIVE CHAOS: INTERACTIONAL COHERENCE IN ASYNCHRONOUS LEARNING ENVIRONMENTS. Saarbruecken, Germany: VDM Verlag, 2008.

[29] C. Cornelius and M. Boos, "Enhancing mutual understanding in synchronous computer-mediated communication by training: Trade-offs in judgmental tasks," *Communication Research*, vol. 30, pp. 147-177, 2003.

[30] M. D. Dickey, "Teaching in 3D: Pedagogical affordances and constraints of 3D virtual worlds for synchronous distance learning," *Distance Education*, vol. 24, pp. 105-121, 2003.

[31] G. Ravid, "Scale free and small worlds networks: Studying a-synchronous discussion groups," *Association of Internet Researchers (AOIR) Fifth Conference*, Sussex, UK, 2004.

[32] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks " *Science*, vol. 286, pp. 509-512, October 1999.

[33] A. Barbabási, LINKED: THE SCIENCE OF NETWORKS. Cambridge, MA: Perseus Publishing, 2002.

[34] L. von Ahn, "Games with a purpose," 2006, pp. 96-98.

[35] L. von Ahn, M. Kedia, and M. Blum, "Verbosity: A game for collecting common-sense facts," *CHI '06: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York: ACM Press, 2006, pp. 75-78.

[36] B. Laurel, "Strange new worlds of entertainment," *Compute*, vol. 13, pp. 102-104, November 1991.

[37] H. W. J. Rittel and M. M. Webber, "Dilemmas in a general theory of planning," *Policy Sciences*, vol. 4, pp. 155-169, 1973.

[38] R. Kazman and H.-M. Chen, "The metropolis model a new logic for development of crowdsourced systems," *Communications of the ACM*, vol. 52, pp. 76-84, 2009.

[39] R. Orna and S. Mary, "An approach to preserving sufficient correctness in open resource coalitions," *Proceedings of the 10th International Workshop on Software Specification and Design*: IEEE Computer Society, 2000.

[40] M. Shaw, "Sufficient correctness and homeostasis in open resource coalitions," in *Fourth International Software Architecture Workshop (ISAW-4)*, 2000.