A Self-Scheduling Model Using Agent-Base, Peer-to-Peer Negotiation, and Open Common Schema

Cheng Hsu
Decision Sciences and Engineering Systems, hsuc@rpi.edu
Christopher Carothers
Computer Science, chrisc@cs.rpi.edu
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

ABSTRACT
Research has shown that market-style self-scheduling is a promising approach to achieving real-time online resources allocation. However, the field still lacks sufficient results in areas such as control, inter-operation of heterogeneous data semantics, and qualitative performance measurement, to satisfy large scale distributed operations, especially today's information enterprises.

We propose a new agent-based, peer-to-peer publish and subscribe model for enterprise resource allocation under the above conditions. We use the proven industrial exchange technology to develop a full-fledged artificial market to self-schedule information tasks (resources offerings and requests) across a potentially extended enterprise, with a comprehensive performance-feedback-re-allocation mechanism. New results also include an Agent-Base for creating and managing large number of custom agents online, a peer-to-peer negotiation method, and an open common schema design, to enable the model. It also provides an implementation scheme for developing the basic architecture, the pricing model, and the data interchange required of the artificial market.

Keywords: market-style scheduling, agent-base, peer-to-peer negotiation, common schema, Metadatabase

1. MARKET-STYLE SCHEDULING

Information enterprises include virtual enterprises, extended enterprises, and enterprises that feature information production and integration. For the purpose of this paper, they feature conspicuously the use of Internet technology to reach out and integrate, and thereby improve their performance. The intelligence community, news organizations, the ASP (application service provider) model of e-business[1], industrial exchanges (e.g., Covisint and CommerceOne)[2], and the service business of industrial equipment manufacturers (e.g., Boeing and GE Industrial Systems) are representative examples. At the heart of these enterprises is the scheduling and control of their resources - i.e., the databases, networked computers, and the like; which tend to be widely distributed and heterogeneous in their technical design, and may also require openness and scalability of the regimes inter-operating them. Without enterprise-wide management of enterprise resources, an information enterprise cannot operate at high level of integration and hence can hardly capture the full benefits of extended enterprising. However, resource allocation for (Internet-based) information enterprises can be more involved to design than the scheduling regimes for traditional enterprises such as manufacturing and transportation, because of the nature of information production. In fact, we can characterize it as a resources allocation problem under the conditions of globally distributed resources and users (providers and requesters), heterogeneous information models, and real-time assignment with online performance measurement and adjustment – or, the enterprise resource allocation problem.

The enterprise resource allocation problem defies many premises of classical scheduling theory [3] and online scheduling [4, 5]. The classical paradigm focuses on optimizing the supply (resources) with respect to a given demand (tasks), subject to workflow precedence and other job constraints. This leads to the dichotomy

of resource versus user in the tradition of manufacturing, where machines and jobs are two orthogonal genres and it does not consider the possibility that a job could be a resource, nor a resource be a job. Therefore, the instances of each genre are homogeneous in their technical nature; and both genres can be characterized in a unified set of definitive terms such as machining capacity, classes or functions, and processing times. The matching of a job to a machine in this context is never ambiguous and the objective function can be neatly analyzed with respect to throughput, make-span, tardiness, utilization rate, and other physical performance measures. When necessary, such as in online scheduling for computers, an assignment can even be moved around as in bin packing, in order to optimize the overall performance within a single migration round. Finally, a scheduling regime is designed to be a planner rather than an executioner, and it does not consider real-time conditions nor online feedback from the system that executes the schedules when determines the schedules. The literature assumes that either the system's controller will adjust the schedules to accommodate real time conditions, or the scheduler will re-run itself with new conditions to produce a new result in the next planning window.

For Internet information enterprises, the resources allocation regime would have to do better since the environment includes not only physical facilities but also information resources such as databases and personal information assets. The regime must produce maximum quantity of information to maximum suitable users with maximum relevance and quality, with minimum delay. Thus, the regime has to **encourage** information sharing, respond to real-time conditions online, and re-allocate resources according to performance feedback. That is, it must consider both the supply and the demand. The regime must allow for resources providers to also be users; e.g., a field officer could both provide and request information and an automated analysis system or a database might request input or even co-processing from other facilities as well as produce output. In addition, these providing or requesting tasks might use different data semantics to describe their information contents and requirements. When information exchange is involved, the semantic uniqueness makes tasks heterogeneous and requiring **individual, custom** representation, attention, and processing. Furthermore, Internet information enterprises involve extended organization (inter-organizational tasks), globally distributed resources and users, and potentially very large number of participants. We submit that these characteristics fit best with those of an artificial market, such as stock exchange or industrial exchange in e-business.

A number of researchers have recently proposed market-style resources allocation schemes using software agents to make conventional scheduling models more in line with real-time assignment [6, 7, 8, 9, 10, 11, 12, 13, 14]. These newer efforts tend to create a pseudo market for, e.g., shop floor scheduling and computer networks allocation, where facility agents and job agents meet and match. These works, however, tend to lack an effective market mechanism - i.e., a performance-feedback-reward loop - to measure the value of the resources in the market and thereby approach global optimality. Without this self-correcting capability, the pseudo market is more a metaphor than a complete mechanism capable of capturing the benefits of the market model. These designs tend to ignore the adaptive capacity of agents required to perform semantics matching, multi-criteria negotiation, and other dynamic tasks. They also do not consider how to create efficiently large number of agents online and effectively manage these agents when updates are necessary. These issues become critical when the pseudo market were to scale up to handling, say, hundreds or even millions of concurrent, custom tasks. Consequently, the assignment schemes developed under this paradigm do not offer sufficient feedback and adaptation to assure self-correction. Not surprisingly, some of these pseudo markets exhibit various global inefficiencies (e.g., long queues at certain resources sites caused by obsolete information at the global site, and tasks not getting assigned properly due to lack of negotiation); and others excessive overhead (imposition of a global controller to supersede the self-scheduling. All of them lack the promises to address the potential of influencing demand and supply such as encouraging information sharing.

In contrast, we develop a full-fledged artificial market model, the ***Enterprise Resources Market,*** using the proven technology of industrial exchanges to accomplish the goal of self-scheduling resources allocation for Internet information enterprises. The new model promises to resolve the above problems and

achieve balance of demand and supply, computational efficiency (linear to low polynomial complexity), information interchange (semantics match), and self-correction (performance feedback and reward) for resources allocation. The model includes new extensions to enable the previous exchange to allocate both physical and information resources that the enterprises involve, and to execute the information processing tasks at the local level on a peer-to-peer basis. The new results provide an agent model and an agent-based architecture to connect distributed resources providers and users to the market, as well as directly to each other. The new agent model allows for very large number of concurrent participants (exponentially scalable) to use software surrogates to globally publish their offers and requests of resources, match, and negotiate-auction (a global Black Board); and then connect locally with their matches to subscribe to the resources. It also provides an alternative peer-to-peer negotiation model to allow matching and auction among local sites without a global Black Board. A pricing model drives negotiation and feedback towards achieving a globally sound, self-scheduling regime. The pricing model ***encourages*** sharing of information and resources in the same time it controls the use of them. In this regime, both providers and users can initiate tasks as bids for transaction, while a global server facilitates the creation, management, and processing of their custom (task-oriented) agents. The global server also uses metadata technology to represent task characteristics, including transaction requirements and data semantics, and to match requests with offerings according to these characteristics. It subsequently conducts the negotiation, auction, and final connection of tasks to local resources. As such, the objective function is the maximization of the total (perceived) value of information and physical resources. It is worthwhile to note that the pricing model encompasses and synthesizes such criteria as minimization of delay in assignment and optimal utilization of resources, as well as provides performance-based rewards and adjustment, in a way similar to that associated with a natural market. The constraints are the transaction requirements and data semantics of tasks, which could be updated real-time and online through the task agents, based on local conditions (e.g., work load and deadline). Self-scheduling takes place at matching, the allocation-connection, and the queuing at the local resources; and hence assure computational efficiency. Finally, we should note that the Enterprise Resources Market accomplishes both resources allocation and information sharing for the extended organization. With the pricing model, it ***optimally*** allocates the published tasks and helps subscribe to the resources; without it, it still matches tasks and helps interchange of information resources.

In sum, this work contributes a self-scheduling resources allocation mode with new enabling results; which include a new agent model for managing large number of concurrent, custom agents to add to the software agent literature, a peer-to-peer negotiation model as a new result to e-auction, and a method to achieve open schema for exchange technology. Next, we discuss how the Enterprise Resources Market works in Section 2, and present the detailed methods in Section 3. Section 4 concludes the paper with an assessment of the new model and its future work.

2. THE AGENT-BASED, PEER-TO-PEER, PUBLISH AND SUBSCRIBE MODEL

We define that the user community, i. e. the organizations involved, encompasses multiple operating groups, databases and computing networks all of whom have information and can process data. Both the human and machine components of the community can both be providers and requesters of information resources. That is, all participants of the market can play both roles of sellers and buyers (e.g., an automated information system may trigger a request for information or co-processing from other sources during the execution of an analysis). The organization uses budgets to control the allocation and utilization of information resources. All users, therefore, pay from their funds (real money or fungible credit) for their requesting tasks (buy) and take revenue from their offering tasks (sell). The market is, hence, a performance and reward mechanism, as well; to which the management could complement additional adjustments of funding, as desired.

Participants use task-oriented and manageable agents to publish their requests and offers of resources, and use the same to subscribe to the resources. Software agents bring about several significant advantages: asynchronous (24/7) transaction, controllable consistency with enterprise knowledge and requirements, and

security (e.g., the participant can publish only the information slated to offer, and conceal the true nature of the tasks or identity of the requesters from the providers if necessary). The design of the agents must also include additional intelligence such as preferences and transaction rules to further automate negotiation and peer-to-peer transaction as described below.

At the global server level, a blackboard is maintained to match requests and offers based on task characteristics such as deadline, specific requirements, availability, and perceived value in terms of price. The criteria allow for non-exact or staggered match. After one or more matches are found, a pricing model will perform the assignment, which entails negotiation, including group auction and revision of terms, amongst the matched parties if alternative allocations exist. The global server maintains and makes available the market status to all participants to help them publish their bids and negotiate, as well as provide remedy to tasks having difficulty getting allocated. For instance, the server could increase the price offered by a request to find it a match just before the deadline. The loan becomes a feedback to the reward system on the initiator of the task. After finalization, the agents proceed to establish connections for the requesters and the providers' resources at the local level. A proxy server of adjustable complexity could reside at the local resources to enable peer-to-peer transactions. The requests become jobs at the local resources and queue themselves according to the price offered and follow the local queuing discipline. This process does not require global coordination. The queuing status, including workload, will become feedback to update the local resource's agents at the blackboard. The global server contains a Metadatabase about tasks characteristics and enterprise requirements to support the blackboard. The server also includes an Agent-Base to create and manage the task agents online, which could number in the millions, according to users' instructions.

An alternative to going through the Black Board is for a participant to initiate directly task agents that visit other task agents at other local sites to find-negotiate the matches and conduct auction when necessary, on their own. This alternative is available to local sites that have sufficient computing power. In this case, the initiating participants will control the virtual auctions that their task agents started first (distributed computing); and hence simplify the computing load at the global server.

The details of these techniques and methods are discussed in the next section. Leading to it, we first describe here the overall architecture of the artificial market. The fundamental architecture connects its global server (the exchange site) to the local resources providers and users (the participant sites). The architecture also recognizes the other operating regimes existing in the community. In a nutshell, there are three basic domains of administrative control: The artificial market comprises its own operating domain, the Exchange Domain, and has full control of this domain. The enterprise participants control their own local operating domains, the Participant Domain, delegated to them from the rest of the organization's operating regime, the Enterprise (Agency) Domain. That is, the Exchange Domain only inter-operates with the Participant Domain as mutually agreed upon but does not otherwise intrude it. Firewalls could exist between these domains. Similarly, the Exchange Domain could support the Enterprise Domain according to some charters; otherwise, they are separate from each other. In this research, we will focus on the Exchange Domain and the extent to which it inter-operates with the Participant Domain. We assume the participants come from multiple organizations of the community and could locate physically in any part of the world. Thus, the architecture includes Internet as a means of networking between the participants and the artificial market. Other means owned by the Enterprise could replace (or augment) the Internet without altering the overall design of the model.

The basic structure of the participant side of the architecture consists of a proxy server controlled logically by the global server of the exchange side. The proxy server "represents" the exchange at a local participant site and inter-operates with the local systems. The exchange side features the black board, an agent-base, and a common schema, among other things. To develop the required results, we first create a generic match engine using a thin code such as Erlang, and then develop the rest of the elements of the artificial market into a prototype, including the extended agent design, the assignment algorithm, the global server, and the proxy server at local resources to conduct peer-to-peer transaction. To address implementation issues, we also

formulate a plan about how an information enterprise might map the pricing model to its organizational valuation metrics and budgets, and what standard it might adopt to inter-operate the artificial market with other functions it conducts. The prototype can then be validated through either a laboratory testing or simulation, before being site tested in an actual implementation. These components and other particular methods that make the Enterprise Resources Market possible are discussed in detail below.

3. THE METHODS FOR THE ENTERPRISE RESOURCES MARKET

The above architecture entails several major elements. Some of the key methods and techniques are adopted from our past research. We adapt these results for the Enterprise Resources Market, and further develop them and the new elements not available currently to satisfy the requirements of the system. We discuss the details below.

3.1 The Agent Model: Task Agent, Agent-Base, and Metadatabase.

Resources providers and users initiate their offers and requests through custom created task agents. They log on (remotely) to the Agent-Base at the exchange site and instruct it to create a task agent for their offers or requests. The software agents are uploaded to the local sites of the participants. They use the same mechanism to update or delete their agents. The participants can now initiate a task by launching the agent created to the Black Board of the exchange site. The agent ***publishes*** its information content (see below) at the Black Board, with possible subsequent modifications. The Black Board uses this information to conduct matching, negotiation-auction, and assignment, as described in the next section on Black Board.

The agent has three basic elements: the communicator, the information content, and the rule-base. The communicator includes header (e.g., ID or IP address, XML-SQL or inter-operation protocols, and other metadata and routines required for agent processing). We will consider the best practices in the field as well as the operating policies of the enterprise to determine the actual design of the communicator. The information content describes the conditions and semantics of the task according to certain representation methods acceptable to the enterprise. In any design, the conditions specify the price demanded or offered, the deadline, and processing requirements; while the semantics use either the common schema of the organization or, if one does not exist, the common dictionary of keywords to communicate to other agents the information nature of the task. The processing requirements include the type of resource offered or requested, job constraints and task status if the task belongs to an automated workflow (series of single tasks) or complex task. A complex task will be processed as a sequence of single tasks according to processing or workflow rules. The rule-base contains operating knowledge for conducting automatic negotiation, auction, and other similar behaviors, such as choices of pre-determined negotiation schemes. It also contains workflow rules and other logic for the processing of complex tasks. All agents follow a unified protocol regulated by the Agent-Base using the knowledge stored in the Metadatabase.

The Metadatabase contains a version of the communal or extended organization-wide common schema, or a common dictionary of keywords that the participants use, for information enterprise resources allocation. It, along with the proxy server (see the section dedicated to this topic below), constitute the connector of the artificial market through which the system plugs into the overall enterprise environment. While the proxy server provides physical connection in an API manner, the Metadatabase offers logical integration with the enterprise. If these common schema or keywords do not currently exist, then we need to develop the keywords with enterprise experts as an implementation effort. Alternatively, when the enterprise chooses to develop its own common schema from scratch, we could employ the Metadatabase model developed at Rensselaer over the past decade [15, 16, 17, 18, 19] to accomplish this purpose. The Metadatabase model employs a particular representation method based on the TSER information model to integrate enterprise information models, contextual knowledge, software resources (for inter-operation), and user-application families. These enterprise metadata are structured into a database on its own so that the community can query, manage, and evolve enterprise metadata resources through the Metadatabase for their tasks in the same manner as they could for regular data resources with a regular database.

Therefore, the Metadatabase can be a repository of enterprise policies pertaining to the artificial market (such as rules about particular user-application families, entities, and relationships). Moreover, the scope of the representation method covers all three elements of the software agents; thus, the Metadatabase can also be a depot of re-usable objects or common raw materials (communication software, information content, and rules) from which the Agent-Base builds task agents. In any case, the provision or the construction of the common schema and/or keywords, regardless of the methods taken, would have to come from the enterprise experts in the extended organization. Without them, the proposed research would only simulate one for the final prototype.

One aspect of the Agent-Base is a method to mass-customize large amount of agents at run time. When the potential task agents at any one time could number thousands or even millions, and most of them are custom build, then we need an efficient way to create and manage these agents online and on the fly. We will use the Metadatabase to provide community resources required by mass production, and use the Agent-Base to customize the configuration of these resources for particular tasks. It will also support the owners of the agents to add ad hoc information (e.g., specific data values of some entities, relationships, or attributes, and operating rules) and route them to the Metadatabase for possible inclusion into its content. Another aspect is the ability to automatically update the metadata contents of agents when these metadata are changed at the Metadatabase. This capability, unique to the Agent-Base model, is very useful for maintaining the logical consistency across agents, or the integrity of the agent community. The third aspect is a log of the task agents currently active at the Black Board. This design allows the software agent to be a persistent surrogate (24/7) at the market for conducting asynchronous negotiation, among other things, to enhance reliability and performance. Thus, the Agent-Base is both a management shell and a gathering of active agents.

We need to fine-tune the agent design, the Metadatabase model, and the Agent-Base of the Enterprise Resources Market for particular enterprises. However, some general designs do exist. Both the Metadatabase and the Agent-Base could be implemented on a commercial, stand-alone database management system such as Oracle. We envision the necessary user interfaces of these components to be added shells of the database using its build-in facilities.

3.2 The Black Board: Match, Negotiation-Auction, and Assignment

The Black Board is the regular, default mechanism for the Enterprise Resources Market to serve the agents and conduct match, negotiation-auction, and assignment. This engine maintains a list of all tasks published by agents, including their information contents, conduct matching and negotiation, and finalize the assignment. It also consults with the Metadatabase for the latest enterprise policies to avoid chaos at the artificial market; one of these responsibilities is to break ties and ascertain that all (worthy) tasks find a match before their deadlines.

The basic logic of match goes this way. For a given task, it first satisfies the semantic constraints by looking for counterparts possessing the same information content. The match is based on metadata, either from common schema or keywords, and can either be exact or partial. The software agent specifies the rules. The Black Board could either incorporates the rules in its matching (custom match) or inform the logged agent of the result of standard match for it to exercise the rules. When semantic constraints are met, the matching proceeds to conditions including price, deadline, and other requirements. If single perfect match exists, then the Black Board will assign the task to the resources matched. If multiple perfect matches are found, then a round of auction amongst them will decide the final assignment. If only partial matches are found, then the task enters negotiation. The negotiation could be automated where the logged agents concerned use their rules to find an optimal match and break ties by auction; or, the agents could inform the task initiators and have a round of modification of the original conditions – i.e., human intervened negotiation. On the other hand, if no match, perfect or partial, is found, then the agent and/or the initiator could update the task information content depending on whether the difficulty is caused by semantics or by conditions. The Black Board will post current market conditions – e.g., statistics of bids, usage patterns of keywords, and status of hot issues – to facilitate the modification and negotiation. The initiators could also proactively update the task agents

stored at the local site and re-launch it to replace the old one. Certain conditions, especially those related to workload at local resources, would be suitable for automatic update from the participant sites. The Black Board intervenes only on an as-needed, exceptional basis to break ties and enforce enterprise policies. For example, it could check on certain types of requesting tasks that have no match and increase their offering prices on a loan basis to find them a match on or near the deadline.

The assignment phase is essentially a notification of the connections that the tasks should establish with their resources. It entails an update of the communicator of the logged task agent by the Agent-Base, if necessary, to prepare it to communicate with the proxy server at the destination resources site. The update agent will then upload to the task initiating site and initiate a peer-to-peer transaction from there. Now, the task agent is ready to ***subscribe*** to the resources; namely, connect to their system.

It is well known that the computational complexity of traditional scheduling algorithms (global control) is NP-hard; while the complexity of sorting (according to price) is linear, *O(*N*)* with N being the number of tasks. Thus, the self-scheduling nature of the Enterprise Resources Market assures a very efficient regime of computation with complexity in linear to low order of polynomial (including negotiation and feedback). Therefore, the artificial market is scalable exponentially in theory; i.e., its computational efficiency allows it to expand virtually freely. However, this is not the case with most other schedulers, whose scalability is inherently limited by its computational complexity.

In this ongoing research, we developed a preliminary prototype that provides a Black Board with interval matching capability. This research continues to extend it to provide full negotiation-auction and assignment for the Enterprise Resources Market. In actual implementations, the final design of the agents and the Agent-Base can cause modification to the basic design of the Black Board; but, otherwise, the method is reasonably generic and stable.

3.3 The Proxy Server: Peer-to-Peer Transaction and Systems Inter-operation

The proxy server is a software system that the artificial market adds to local resources sites and resides there. It accomplishes two basic jobs towards enabling the computing connection of the exchange side with the participant side: systems inter-operation and peer-to-peer transaction. The server interacts with the exchange site and collaborates with the Agent-Base to store, maintain, and process (launch) the software agents owned by its local site, as well as to respond (execute) to the call of task agents from other sites. In this capacity, it functions as the server of the local site for all task agents initiated at the site but processed elsewhere at the Black Board or other local sites. Thus, it executes the workflow logic for its complex tasks to, e.g., sequentially launch the component single tasks and maintain the overall task status. The server offers a data standard for the task agents to communicate between themselves. The exact design will depend on the enterprise requirements; but a good, standing design is to use XML-SQL. That is, the proxy server will have a standard protocol to receive and process task agents for information transfer. Part of the protocol is a standard schema for view tables that the proxy server maintains for the local databases to use. It works two ways. First, the local database publishes the select content that it slates to share with the artificial market at the proxy server, using the format provided by the latter. The information requesting task agents can then query the view tables to retrieve or transfer the selected content. Second, if its own task agents transfer in content from other resources, then the input is stored as view tables for the database to acquire under its own management. The queue discipline at the proxy server is self-scheduling based on prices. However, if the requesting task from outside is to use the computing facility for data processing, as opposed to information retrieval, then the proxy server passes it as a regular job to the local system and follow the local queue discipline. In a similar way, it monitors the work load, task status, and other relevant data of the host server to update its task agents at the Agent-Base and elsewhere in the system.

The proxy server also controls peer-to-peer negotiation, including matching and auction, as described in the next sub-section, if the local site has sufficient computing power to invoke this option. In this capacity, the proxy server launches its task agents to visit task agents publishing on proxy servers at other local sites,

as well as prepare them for negotiation with visiting agents from other participants. The proxy server from which the task agent first (time stamp) initiates a peer-to-peer negotiation in the community will function as a mini Black Board during the life of the negotiation. The basic logic of the Black Board applies here, except that the initiating task agents call on other sites rather than other agents posting onto its proxy server. As such, a proxy server might control several concurrent auctions involving different tasks at different sites, and the community might have numerous such auctions controlled by different proxy servers at numerous local sites at the same time; all are autonomous to the global server. In this mode, a proxy server will maintain a list of visiting task agents and matches its own task agents against them at the time the participant publishes them.

The employment of proxy servers allows for peer-to-peer transaction and hence concludes the assurance of the computational efficiency promised by the price-based Black Board. Peer-to-peer transaction is advantageous to the environment for three basic reasons: it allows for the participants' direct control of all tasks originating at their site as well as tasks being processed there, and hence simplifies the global control requirements and work load; it supports distributed updates and processing of agents based on local conditions; and it provides a backup to the Black Board. Furthermore, it also contributes an open and scalable way to connect any number of local databases and other resources into the Enterprise Resources Market without interrupting the operation of the market. Along with the Metadatabase, which offers an open and scalable way to incorporate any number of information models into the market on the fly, and the Black Board, which promises computational scalability, these three elements of the Enterprise Resources Market make this design **uniquely** open and scalable.

An implementation of the model can adopt the best practices in the field to build the proxy server for the enterprise, in light of the new agent model and the Black Board. A reference point for the technology is the commonly available products such as Apache server, which is extendable with JAVA-J2EE, PHP, and other general purpose programming languages. Although one needs to design it, the proxy server has many mature technologies to choose from for its implementation.

3.4 Peer-to-Peer Negotiation: Virtual and Distributed Mini Black Boards

Peer-to-peer computing has a general complexity of $O(N^2)$ which hinders scaling-up. We develop a new basic logical structure, ***the match circles***, to denote the group of notes (local sites) whose tasks match. Pair-wise computing is unnecessary within a match circle once it is recognized. Thus, a circle will have a node serving as its mini-Black Board and thereby reduce the computing complexity. Since a local site can have a number of simultaneous tasks alive in the community, it can belong to a number of simultaneous match circles. Thus, both the circles and the mini-Black Board are virtual and task-based. This way, the overall complexity of the peer-to-peer computing is primarily the number of such virtual circles, which is arguably much less than the theoretical upper bound. We elaborate on this idea below.

The Agent Base maintains a global protocol for determining time stamps for all task agents initiating a negotiation, which starts with a search for matches at other local sites and finishes when an assignment is finalized after, if necessary, auctioning among multiple matches. The negotiation at the matching phase is concerned with task constraints, while it is about the objective function when auctioning. The initiating task agent, the one with the earliest time stamp among all agents that matched, has the control of the negotiation. Its proxy server first launches a round of search where the task agent looks for matches at (all) other local sites in the community in a purely peer-to-peer manner. If a single perfect match is found within a pre-determined time period, then the proxy server acts according to the nature of the task: for requesting task, it obtains necessary inter-operation parameters or routines from the Agent-Base for the task agent and sends it to queue at the matching site through the proxy server at that site; and for offering task, it informs the proxy server of the match task to launch the requesting agent. Optionally, the proxy server could also contain a reduced copy of the Metadatabase to allow it augmenting its task agents with the inter-operation data. If multiple matches exist, then the proxy server conducts an auctioning session where it sends out asking prices, iteratively, to the matches in the manner of traditional auctions – i.e., the proxy server singly controls the auction session. The result will either be a single meeting of the best price – in which case the

proxy server assigns the task as mentioned above, or a declaration of failure of the auction which results in a deletion of the current task. The participant in the latter case can opt to re-initiate the task or re-create a new task agent.

If no perfect match is found when the time expires, then the initiating task agent every where starts a round of lock-step, pair-wise negotiation with its host agents. Each pair of negotiation is independent of all other pairs under the initiating task agent's autonomous control, which uses the same negotiating regime to proceed. The regime could be rule-based, staged modules, or any appropriate design, as long as it uses definitive steps to define and control its gives and takes, with each step associated with a certain time window. Thus, all pair-wise, simultaneous negotiations at all local sites are at the same steps at all times. Each step modifies certain constraints in certain manner within each window, and the proxy server terminates the negotiation at the first moment when a perfect match or matches are found. At the conclusion of each step of the negotiation for matches (on constraints), the task agent returns the matches (when achieved) with an indicator of the step during which they are obtained, along with the identification of the local proxy servers of the matched task agents. The Initiating proxy server could use the indicator to determine the matches in the assumption that modifications are reversely favorable in the sequence of steps and hence the earlier the matches the more preferable. Thus, if some negotiations are lagging behind because of their local queuing situations or any other reasons, then they could preempt incumbent results (including auctions) when they report matches back to the initiating proxy server. However, the proxy server could also opt to ignore late results whenever the auction is underway. The task agents use the time-based progression of negotiation to synchronize virtually their autonomous processing. The initiating proxy server, i.e., the mini Black Board in this case for the round of negotiation, does not actively control the processing of its task agent at each local site, but only determine the matches from all reported results. When the time expires without any matches found, the task agent ceases to exist and the participant could either revive it or forgo it. The peer-to-peer design allows a proxy server to control the negotiations (matching and auctioning) of its task agents, individually as well as collectively (to manage its own local resources), and thereby promote distributed computing. There could be many proxy servers controlling many concurrent negotiation sessions in the community at any time, each of which is a virtual mini Black Board for the virtual group of matched task agents.

3.5 The Implementation Model: Organizational Metrics and Data Standard.

To implement the artificial market in the enterprise environment, we need to investigate some of the organizational issues, especially how to map the pricing model to organizational control metrics and how to inter-operate the Enterprise Resources Market with other functions and systems. We assume that the extended organization uses a budget model to control the overall allocation of resources to operating units and individuals. The model will create artificial funds (either of money or of fungible credit) for participants of Enterprise Resources Market, and periodically deposit or adjust them according to their overall performance at the market over the period, among other things, on an off-line basis. In addition, the market maintains these funds and will automatically adjust them for the participants after each transaction to reflect their revenues (sales) or payments (purchases) in a manner similar to a bank. Thus, individuals and operating units do not exchange money directly, but their purchasing power as recognized by the market. The bids, therefore, reflect the perceived value of the resources requested or offered by the participants. To make the scheme work, the fund owners must have control of their funds and the ability to use surplus for real world purposes such as hiring people or purchasing facilities. Thus, the market itself is the first mechanism to measure performance, reward the participants, and thereby reallocate resources. The managers of the participants will also assess the performance of resources in terms of value added to their missions; that is, the quality and quantity of information provided and utilized. They could measure the value in terms of how many people have used and benefited from the offerings (sales) and how actively people seek out for useful information. The system will generate market statistics on problems (e.g., loans), requests, offerings, transactions, connections, keyword usage (hot topics), as well as accounting and scheduling logs. Results are performance feedback to the managers for the budgeting process and reward systems. This

periodic review assisted by market statistics is the second line of control and reward for resources allocation.

We need to study the optimal way to design the pricing model for an implementation so as to best fit the organizational metrics of performance evaluation and reward. Furthermore, for organizational evaluation, the implementation also should consider developing the possibilities of collaboration with other enterprise management functions. For instance, the patterns of subscription (connection) reveal the need for new or ad hoc channels of communication or workflow processes. Thus, these data could feed into such models as organizational networks and processes.

The data standard issue concerns inter-operating local databases within the domain of the enterprise and possible collaboration with other enterprise functions. The first aspect is a matter of information integration and data interchange protocols. As discussed above in the sections on the agent model and the proxy server, we propose to use the current organizational specification of keywords or common schema to represent data semantics and store them into the Metadatabase, to achieve logical information integration; and use the proxy server to handle data interchange. This approach is well established in industry; in addition, the above sections also offered realistic alternatives for the development of Market data standard. We are confident that in the case of actual implementation, one can adopt the best practices in the field of e-business and database integration to recommend a data standard for the enterprise and make the artificial market work as designed. At present, we propose common technologies including XML-SQL, relational databases, and Internet-based computing protocols. The data format required for collaboration with other enterprise systems will come from these sources. We will continue to investigate these issues and hopefully recommend some general designs as the field matures.

3.6 Open Common Schema: a Metadatabase for Extensible Information Integration

The section on the Agent Model provides a Metadatabase to represent data semantics of all resources in the enterprise. This task usually corresponds to developing a common schema at the high end and a reference dictionary of keywords at the low end. Current practices in the field at both ends have certain drawbacks: available common schema methods tend to be hard to develop and rigid to maintain, while keywords might not cover the full semantics contained in information models (especially relationships and contextual knowledge - processes). The Metadatabase model, on the other hand, is an open, scalable and integrated repository of enterprise information models (in the form of metadata), constructed on a minimal ontology of generic information modeling concepts per se. The ontology is comparable in concept to the Information Resources Dictionary System (IRDS) effort of NIST (see [16]) and similar approaches in the current Enterprise Integration community. However, it differs fundamentally from designs that generalize application logic for an entire domain. The methodology-based ontology offers efficiency and simplicity (minimalism), but is limited to the applicability of the method on which the design is based. For the Metadatabase, the basis is the Two-Stage Entity-Relationship model. Previous works have shown the model to be scalable (i.e., metadata independence) [14] and capable of incorporating rules [18, 19] and supporting global query processing across multiple databases [17].

The icons of the Metadatabase structure, or the graphical representation of the ontology, shown in Figure 4, represent either a table of metadata or a particular type of integrity control rules. The ontology in Figure 1 extends slightly the previous Metadatabase structure by also including user words and cases (as in case-based reasoning), to enhance the extension of the model. The metadata include subjects and views, entity-relationship models, contextual knowledge in the form of rules, application and user definitions, database definitions and database objects. User-words are defined as ordered pairs (class, object). Classes include Applications, Subjects, EntRels (entity-relationship), Items, Values, and Operators; all of which are metadata tables as shown in Figure 4. Objects are instances (contents) of these classes. An object is uniquely identified by an ordered quadruple (Item name, EntRel name, Subject name, Application name) as well as an identifier. A case consists of a problem definition and a solution, but not the usual outcome, because the Metadatabase contains the complete domain knowledge needed. New problems (e.g., exceptions to general policies) would use the

problem definition to find the (best) matching cases and apply the associated solutions to them. A set of metadata for a task describes the problem definition, and its interpretation defines the solution. Cases strengthen the Black Board's ability to perform real time matching and assignment of tasks when uncertainty arises. As such, the Metadatabase collects local information models as the elements (metadata entries) constituting the common schema.

The common schema so constructed will be able to accommodate changes, including deletion, addition, and modification of information models for local resources. For instance, when a new local site is added to the Market, the necessary "registration" effort will be to create an information model using the TSER methodology for the new system (either by the local participants or by the Market experts), and add the information model as new metadata entries to the appropriate meta-tables of the Metadatabase (using e.g., SQL Insert statements). The Metadatabase does not need to shut down at any time during the update, since the operation is no more than a regular database transaction. After this logical connection, the Market will install a proxy server fine-tuned for the new local system in the local environment. This installment does not interfere with the regular operation of the (rest of the) Market, and hence the whole addition process will not affect any existing local systems nor the on-going tasks at the Market. Once the process is completed, the new site takes part immediately and automatically in the Market. Other changes are similarly self-contained and autonomous. Therefore, the design offers an open common schema to enable extensible information integration for the community: again, a facility for the openness and scalability of the Market.

Among the six elements discussed above of the Enterprise Resources Market, two are adapted from standard results in the field: the Black Board and the Proxy server, and one, the Implementation Model, is organization-specific. The other three, namely, the new Agent Model, Peer-to-Peer Negotiation, and Open Common Schema, are new contributions of the paper. The Metadatabase has been tested extensively in LAN, WAN, and even Internet-based environments at some industrial companies. However, it has not been deployed for an open community such as the extended information enterprises targeted here. Thus, its development into the common schema will represent a new contribution to the filed. The computing feasibility of the whole approach is challenged mostly by the Black Board on the global server (the Market), since it entails high volume of concurrent processing for matching and negotiation and thereby presents a possible bottleneck. Ongoing research, especially a rapid prototyping of the market model, tests this bottleneck with in laboratory. The actual work, however, is beyond the scope of the paper.

4. AN ASSESSMENT

The proposed model is complete in its design, but we still need to evaluate it through rigorous testing and comparison with alternative results. We developed a preliminary prototype to help reduce the concept to practice. Although more work is needed in the ongoing research, we believe that the current results show evidence of the basic promises of the proposed model.

The main conceptual accomplishment of this work is the realization that an industrial exchange (marketplace) is a self-scheduling mechanism for allocating certain resources – i.e., buyers and sellers. Therefore, we submit that a similar mechanism with adequate extensions can self-schedule resources for information enterprises and maximize the overall value of resources for the extended organization while satisfying tasks requirements. We then developed a complete design to solve the enterprise resource allocation problem and reduced the concept to practice, as presented above in the paper. In the work, we also developed a new agent model, a peer-to-peer negotiation design, and an open common schema method - which in their own right contribute to the software agent literature and the field of industrial exchange, to enable the new design. Preliminary results of the implementation of the design in a laboratory environment have substantiated the technical feasibility of the approach. The prototype developed so far has shown that it can scale up to a million of simultaneous task agents even running on a relatively simple platform using only off the shelf and public domain technology.

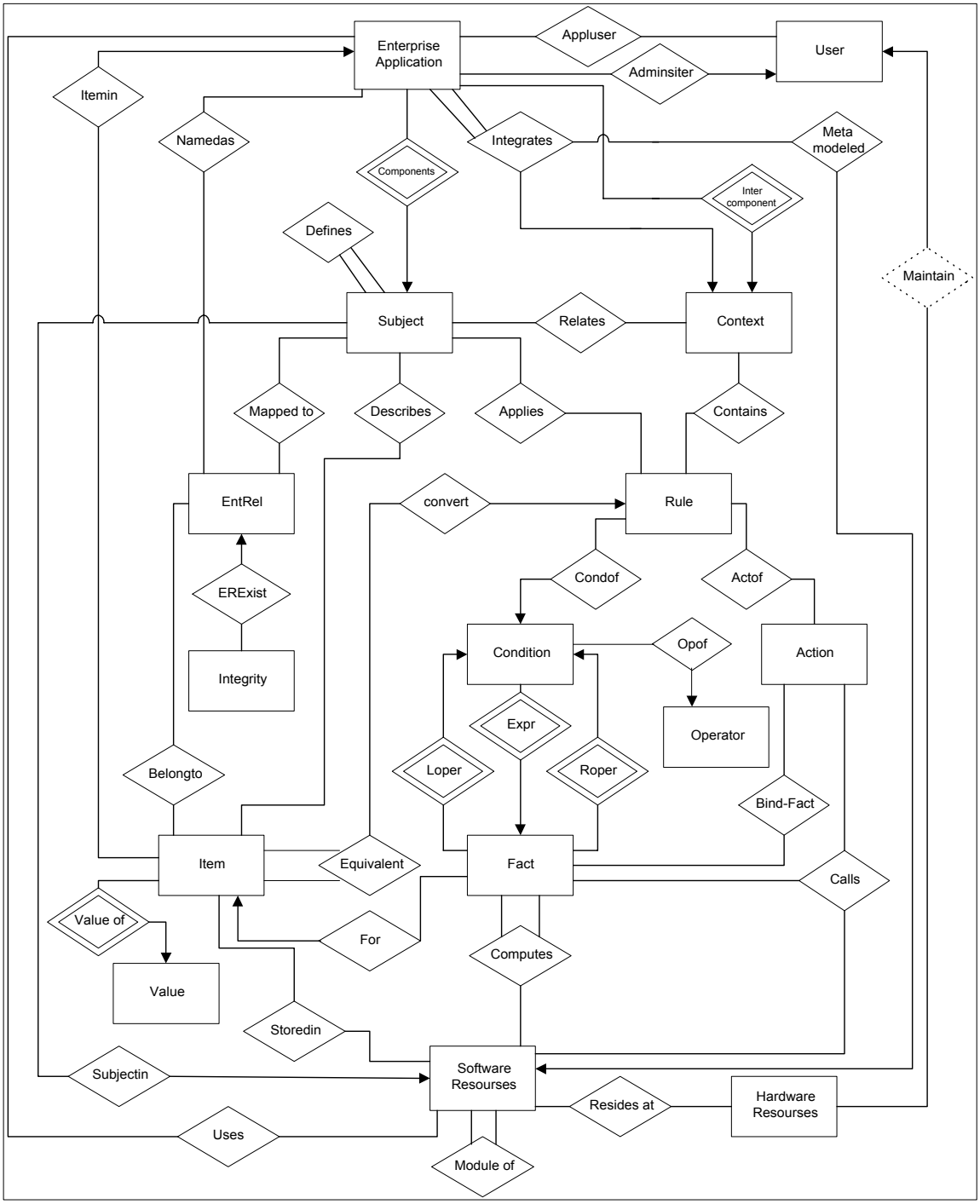

Figure 1: The Ontology: the structure of the Metadatabase.

The proposed approach provides immediately a true market – not just a market metaphor, and a scheduler that satisfies the requirements of information enterprises. Therefore, we developed a new model based on a generic exchange technology and thereby created a self-scheduling resource allocation solution to the above problem. The new design includes an agent-based match engine capable of interval match using task characteristics and multi-dimensional negotiation, a pricing model for the assignment of resources to tasks, an agent-based peer-to-peer architecture to establish the connections between the task and the resources, and a global server to manage the agents and complement the market to assure satisfaction of enterprise requirements. The required methods of the proposed model are presented in this paper. Although they do not include detailed design required for implementation, which is enterprise-specific, they are sufficient for system developers to use as blue prints to guide their implementation for information enterprises.

In the ongoing work, we will design further extensions and implement them to our exchange prototype in a laboratory setting. We will validate the design against standard scheduling approaches using laboratory testing and simulation according to the following criteria: the performance of scheduling on individual tasks (timeliness), the accuracy of tasks matching, the global valuation of resources allocation, and the potential benefits to information enterprise environments. The last criterion includes, for example, how the market might help connect field officers and inter-operate intelligence databases across organizations. It will also examine how software agents' ability to allow for anonymous publications of tasks (both resources providers and user) and subscription of resources helps the enterprises.

The proposed technology contributes intellectually to two hard problems: real time global resource allocation and information integration for extended enterprises; both of which are critical to IT-based organizations. Its extensions on the previous scheduling regimes include its true, price-driven market mechanism to provide comprehensive performance evaluation, feedback, and resources reallocation. This new mechanism is made possible by its new agent model and the extensions to previous exchange technology. The industrial exchange model has always called for the use of agent technology, but actual practices tend to stop short because of insufficient capacity for large-scale agents management. This research fills in the gap with the new Agent-Base using the proven Metadatabase technology. It further extends the previous results to allow for peer-to-peer negotiation and information sharing, where traditional exchanges tend to limit the data transactions to the processing of business documents and straightforward transfer of files. When we can show to have achieved the intended results, we will have accomplished a good progress toward resolving these two problems.

Furthermore, the new results amount to a general agent-based, peer-to-peer, publish and subscribe model applicable to a class of problems in the digital society. Examples encompass naturally the management of global (virtual) enterprises; but they also extend to such novel areas as community collaboration in for-profit or non-profit settings. A particular vision would be for persons, companies, and organizations to buy and sell information resources from the universal Internet community on a task-by-task basis. As the concept of an exchange is general and far reaching, so does the notion of the new model. The new results make it feasible technically to realize some of the new visions of exchange in practice.

ACKNOWLEDGEMENT: The authors wish to thank their colleague Dr. William A. Wallace, who helped clarify the articulation that went into the writing of this paper.

REFERENCES

1. R. Glushko, J. Tenenbaum, and B. Meltzer, "An XML Framework for Agent-based E-commerce," *Communications of the ACM,* vol. 42, no. 3, 1999, pp. 106-114.\

2. L. Tao, "Shifting Paradigms with the Application Service Provider Model," *IEEE Computer,* vol. 34, no. 10, 2001, pp. 32-39.

3. R. Conway, W. Maxwell, and W. Miller, *Theory of Scheduling*, Addison-Wesley, Reading, MA, 1967.

4. D. Coppersmith and P. Raghavan, "Multidimensional On-line Bin Packing:

Algorithms and Worst-Case Analysis," *Operations Research Letters*, Vol. 8, February, 1989.

5. D. S. Hochbaum and D. B. Shmoys, "Using Dual Approixmation Algorithms for Scheduling Problems: Theoretical and Practical Results, Journal of the ACM, Vol. 34, No. 1, January, 1987.

6. A. Baker, "Metaphor or Reality: a Case Study Where Agents Bid with Actual Costs to Schedule a Factory," in Market-Based Control: a Paradigm for Distributed Resources Allocations, ed. S. Clearwater, World Scientific Publishing, River Edge NJ, 1996.

7. A. Baker, "A Survey of Factory Control Algorithms That Can Be Implemented in a Multi-Agent Hierarchy: Dispatching, Scheduling, and Pull," *Journal of Manufacturing Systems*, Vol. 17, No. 4, 1998, pp. 297-320.

8. Cesta, A., A. Oddi and S.F. Smith, "Iterative Flattening: A Scalable Method for Solving Multi-Capacity Scheduling Problems", *Proceedings National Conference on Artificial Intelligence (AAAI-00)*, Austin, TX, July, 2000.

9. S. Heragu, R. Graves, B. Kim, and A. Onge, "Intelligent Agent Based Framework for Manufacturing Systems Control," *IEEE Transactions on Systems, Man, and Cybernetics*, forthcoming in 2003.

10. M. Nandula and S. P. Dutta, "Performance Evaluation of an Auction-Based Manufacturing System Using Colored Petri Nets," *International Journal of Production Research,* Vol. 38, No. 38, 2000, pp. 2155-2171.

11. H. Parunak, "Agents in Overalls: Experiences and Issues in the Development and Deployment of Industrial Agent-Based Systems," ERIM CEC Report, P.O. Box 134001, Ann Arbor, MI 48113-4001, 2001.

12. V. Prabhu, "Performance of Real-Time Distributed Arrival Time Control in Heterogeneous Manufacturing Systems," *IIE Transactions*, Vol. 32, No. 4, 2000, pp. 323-331.

13. J. Swaminathan, S.F. Smith and N. Sadeh, "Modeling Supply Chain Dynamics: a Multi-Agent Approach", *Decision Sciences*, Vol 29, 1998.

14. C. Hsu., *Enterprise Integration and Modeling: the Metadatabase Approach*, Kluwer Academic Publishers, Boston, 1996.

15. C. Hsu and S. Pant, *Planning for Electronic Commerce and Enterprises: A Reference Model,* Kluwer Academic Publishers, Boston, 2000.

16. C. Hsu, M. Bouziane, L. Rattner and L. Yee, "Information Resources Management in Heterogeneous Distributed Environments: A Metadatabase Approach," *IEEE Transactions on Software Engineering,* vol. 17, no. 6, 1991, pp 604-625.

17. W. Cheung and C. Hsu, "The Model-Assisted Global Query System for Multiple databases in Distributed Enterprises," *ACM Transactions on Information Systems,* vol. 14, no. 4, 1996, pp 421-470.

18. G. Babin and C. Hsu, "Decomposition of Knowledge for Concurrent Processing," *IEEE Transactions on Knowledge and Data Engineering,* vol. 8, no. 5, 1996, pp 758-772.

19. M. Bouziane and C. Hsu, "A Rulebase Management System Using Conceptual Modeling," *J. Artificial Intelligence Tools*, Vol. 6, No.1, March 1997, pp. 37-61.