TAGA: Trading Agent Competition in Agentcities [1]

Youyong Zou, Tim Finin, Li Ding, Harry Chen, and Rong Pan
Computer Science and Electrical Engineering
University of Maryland, Baltimore County
{yzou1,finin,dingli1,hchen4,panrong1}@cs.umbc.edu

Abstract

Travel Agent Game in Agentcities (TAGA) is a framework that extends and enhances the Trading Agent Competition (TAC) scenario to work in Agentcities, an open multi-agent environment based on FIPA compliant systems. TAGA uses the semantic web languages and tools (RDF and DAML+OIL) to specify and publish the underlying common ontologies; as a content language within the FIPA ACL messages; as the basis for agent knowledge bases via XSB-based reasoning tools; to describe and reason about DAML-S based services. TAGA extends the FIPA protocols to support open market auctions and enriches the Agentcities with auction services. The introducing of the semantic web languages improves the interoperability among agents. TAGA is intended as a platform for research in multi-agent systems, the semantic web and automated trading in dynamic markets as well as a self-contained application for teaching and experimentation with these technologies.

Keywords: Agentcities, FIPA, OWL, Semantic web, Trading Agent Competition.

1 Introduction

The Trading Agent Competition (TAC) [Wellman, 2002] was a test bed for intelligent software agents that interact through simultaneous auctions to obtain services for customers. The trading agents operated within the travel market scenario, buying and selling goods to best serve their given travel clients. TAC was designed to promote and encourage research in markets involving auction and autonomous trading agents and had proven to be successful after three consecutive year's competitions.

Although TAC's framework, infrastructure and game rules had evolved over the past three competitions [Stone, 2000] [Greenwald, 2001] [Wellman, 2001] [Wellman, 2002], the assumptions and approach of TAC limited its usefulness as a realistic test bed for agent based automated commerce. TAC used centralized market server as the sole mechanism for service discovery, communication, coordination, commitment, and control among the participating software agents. The trading agents communicate with the central auction server through simple socket interface, exchanging pre-defined XML-based messages. In real world, the auction servers and service providers are distributed among the massive open Internet and have distinct service descriptions and diverse service access interfaces. The abstractness and simplicity of the TAC approach helped to launch it as a research vehicle for studying bidding strategies, but are now perceived as a limiting factor for exploring the wide range of issues inherent in automated trading in open environment.

Agentcities [Willmott, 2001] [Dale, 2002] is the international initiative designed to explore the commercial and research potential of agent-based applications by constructing an open distributed network of platforms to host diverse agents and services. The ultimate goal is to enable the dynamic, intelligent and autonomous composition of services to achieve user and business tasks, therefore creating compound services to address changing needs. In such an open and distributed environment, the need of standard mechanisms and specifications is crucial for ensuring interoperability of distinct systems. The Foundation for Intelligent Physical gents (FIPA) produces such standards for heterogeneous and interacting agents and agent-based systems [O'Brien, 1998]. In the production of these standards, FIPA promotes the technologies and interoperability specifications that facilitate the end-to-end inter-working of intelligent agent systems in modern commercial and industrial settings.

Inspired by TAC, we developed Travel Agent Game in Agentcities (TAGA) on the foundation of FIPA technology and the Agentcities infrastructure. The agents and services used FIPA supported languages, protocols and service interfaces to create the travel market framework

[1] The research described in this paper is partly supported by DARPA contract F30602-00-2-0591 and Fujitsu Laboratories of America.

and provide stable communication environment where messages expressed in semantic languages can be exchanged. The travel market was the combination of auctions and varying markets including service registries, service brokerage, wholesalers, peer-to-peer transactions, bilateral negotiation, etc. This provided a much richer test bed for experimenting with agents and web services as well as a rich and interesting scenario to test and challenge agent technology. TAGA is running as a continuous open game at http://taga.umbc.edu/ and source code is available for research and teaching purposes.

The next section introduces the TAGA game and six types of agents. The detailed design of interaction protocol and ACL content language are presented in Section three. Finally we discuss our work in Section four and suggest the future works in Section five.

2. TAGA Game and Agents

We design TAGA framework to support agent-based market simulations and games. Our first TAGA game is the travel agent competition along the lines of those used in the last three year's TACs. In the competition, customers travel from City A to City B and spend several days there. A travel package includes a round-trip flight ticket, corresponding hotel accommodation and ticket to entertainment events. A travel agent (an entrant to the game) competes with other travel agents in making contracts with customers and purchasing the limited travel services provided by the Travel Service Agents. Customer selects the travel agent with best travel itinerary. The objective of the travel agent is to acquire more customers, fulfill the customer's travel package, and maximize the profit.

TAGA provides a flexible framework to run the travel market game. Figure 1 show the structure of TAGA. The collaboration and competition among six kinds of agents, which play different roles in this market, simulate the real world travel market. We found that basing our implementation on FIPA compliant agent platforms have made the framework extremely flexible. We'll briefly describe the different agents in our initial TAGA game.

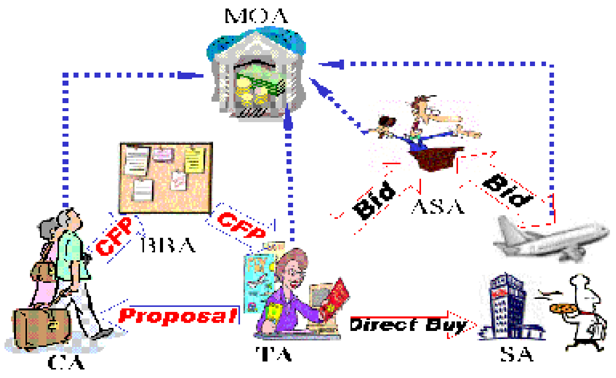

Figure 1: TAGA Architecture

The *Auction Service Agent* (ASA) operates all the auction markets in TAGA. Market types currently include English and Dutch auctions as well as other dynamic auction markets similar to Priceline and Ebay's fastbuy.

The *Service* Agent (SA) offers travel service such as airline, lodging and entertainment activities. Each class of travel related services have multiple providers with different service quality level and limited service units (e.g. hotel room, airline ticket). It allows other agents to query its description (e.g. service type, service quality, location) and its inventory (the quantity or price of a certain type of goods). Other agents may directly buy the service units through published service interface. SA also bids intentionally in the auctions to sell its goods, e.g. listing its hotel rooms in auction and wait for the proper buyer.

The *Travel Agent* (TA) is the broker business that helps customers acquire travel service units and organize travel plans. The units can be bought either directly from the service agents, or through the auction server.

The *Bulletin Board Agent* (BBA) provides a mechanism for customer agents to find and engage one or more travel agents.

The *Customer Agent* (CA) represents an individual customer who has particular travel constraints and preferences. Its goal is to engage one or more TAs, negotiate with them over travel packages and prices, and finally select one TA to purchase the travel package.

The *Market Oversight Agent* monitors the game simulation and updates the financial model after each reported transaction and finally announces the winning TA when the game is over.

The basic cycle of the TAGA game has the following five stages:

(1) A customer-generating agent creates a new customer with particular travel constraints and preferences chosen from a certain distribution.

(2) The CA registers with the BBA sending the customer's travel constraints and preferences in the form of a CallForProposal (CFP) message. The BAA forwards the CA's CFP message to each of the TAs which has registered with it. Each TA considers the CA's CFP and decides whether and how to respond to the CA.

(3) The TAs, who decide to propose a travel package, contact the necessary ASAs and SAs, and assemble an itinerary to propose to the CA. Note that a TA is free to implement a complex strategy using both aggregate markets (ASAs) as well as direct negotiation with SAs. The final

proposal to the CA includes a set of travel units, a total price and a penalty to be suffered by the TA if it is unable to complete the transaction.

(4) The CA negotiates with the TAs ultimately selecting one from which to purchase an itinerary based on its constraints, preferences and purchasing strategy (which might, for example, depend on a TAs reputation).

(5) Once a TA has a commitment from a CA, it attempts to purchase the units in the itinerary from the ASAs and SAs. There are two outcomes possible: the TA acquires the units and completes the transaction with the CA resulting in a satisfied CA and a profit or loss for the TA, or the TA is unable or unwilling to purchase all of the units, resulting in an aborted transaction and the invocation of the penalty (which can involve both a monetary and a reputation component).

3. Agent Communication

The TACs used a straightforward client-server architecture in which a single TAC auction server managed all of the travel service suppliers as well as the customers. Game participants wrote travel agents (TA) that connected as clients to the central TAC server. Moreover, these TA agents can only interact with service providers through centralized auction markets. While this architecture greatly simplifies both the development of the TAC infrastructure and the programming of a TAC client, it is a poor model for commerce in the real world. Peer-to-peer or multi-agent systems offer a more realistic model where customers, service providers and various kinds of "middlemen", including market providers, operate as autonomous peer agents. Moreover, agents can develop complex strategies, which involve a combination of direct transactions (e.g., TA buy direct from hotel agent) as well as auction-mediated transactions of various kinds. Finally, adopting a multi-agent systems approach supports an environment in which all aspects of commerce can be integrated in a more natural manner – service discovery, information seeking, negotiation, decision making, commitment, transaction execution, etc.

The FIPA standards offer mature, published specifications for multi-agent systems communication, interactions and infrastructure with an emphasis on agent communication languages and protocols. We found the FIPA framework to be a good one for TAGA. In the remainder of this section we will briefly describe two additional interaction protocols we have developed for TAGA and the choices we made for the content languages and ontologies.

3.1 Dynamic Contract Interaction Protocol.

To facilitate agents in making contracts with other agents dynamically in a mediated system, we defined the Dynamic Contract Interaction Protocol shown in figure 2. The recruiter (BBA) helps the initiator (CA) to find the appropriate group of participants (TA). All participants (TA) are candidates who can enter into a contract with the initiator, but only one will be successful. Once the contract is struck, the MOA joins the post-contract activities to ensure the two parties fulfill the contract: either the initiator pays for a successful contract or the participant pay the penalty of being unable to fulfill the contract.

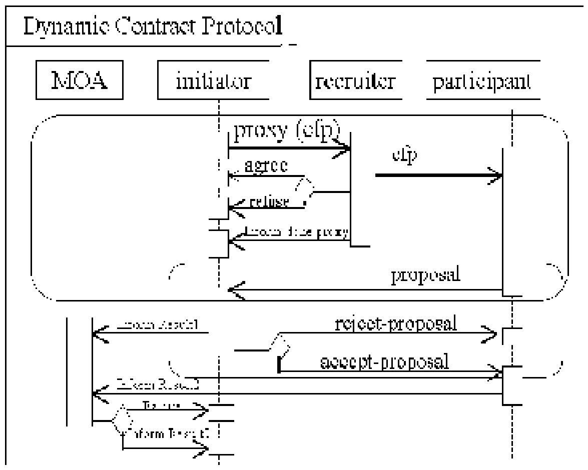

Figure2: Dynamic Contract Interaction Protocol

This protocol is composed of two standard FIPA interaction protocols. Initially, *the FIPA Recruiting Interaction Protocol Specification* is used for the initiator to find participant with the help of the recruiter: the initiator sends a *proxy* message to the recruiter with an embedded *cfp* message; if the recruiter can't find any participant, it sends an *refuse* message[2] back to the initiator, else a *agree* message. The recruiter forwards the *cfp* message to all known participants and sends an *inform-done* proxy message back to the initiator when finished. Once the *cfp* message has been received, the participant evaluates the information and decides whether or not send a propose message to the initiator.

[2] Our brief description does not include all of the richness of the FIPA agent communication language. A refuse action, for example, can include an optional proposition whose truth is a partial reason why the agent is refusing the proposal. Interested readers are encouraged to explore the relevant specifications available at http://www.fipa.org/

If the participant decides not to submit a proposal, no further action is required. Otherwise, the participant interacts with the initiator following *the FIPA Propose Interaction Protocol Specification*: the participants send a *proposal* message contains the proposed contract to the initiator; the initiator selects one most profitable proposal and sends *Accept-Proposal* to the selected participant; other participant receives *Reject-Proposal* messages. An *Accept-Proposal* message from the initiator to the participant means the two parties sign a contract. The participant needs to acquire resource unit and fulfill the contract. The MOA is responsible of monitoring the contract result and informs related parties.

3.2 Priceline Auction Interaction Protocol.

One of the auction types supported in TAGA is based on the model used by Priceline. Traditional auction types [Anthony, 2001] like English auction or Dutch auction are initialized by seller, who announces that the goods are available for sale. Buyers respond by submitting bids and the one who is willing to pay the highest price wins. The Priceline auction, which simulates the auction in *http://www.priceline.com/,* is initiated by a buyer. The buyer creates the auction with the goods he intends to have and the price wishes to pay. The first response seller wins the auction. To support this auction, we defined a FIPA interaction protocol as shown in figure 3.

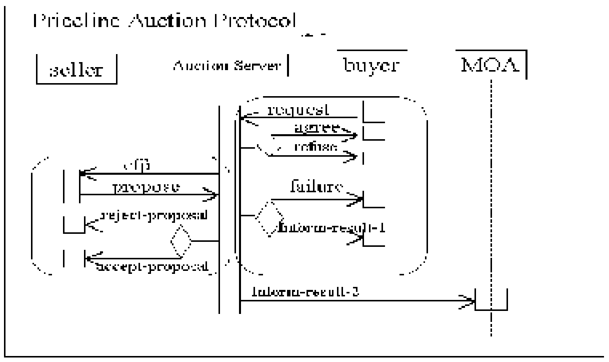

Figure 3: Priceline Auction Interaction Protocol

The Priceline Auction interaction protocol is composed of two FIPA protocols. *The FIPA Request Interaction Protocol Specification* is used for creating auction and informing auction result. The auction server (ASA) creates the auction instance when it receives a *request* message from the buyer (TA) and sends back an *agree* message. A *refuse* message is sent back it is unable or unwilling to create the auction. The auction server uses the *FIPA Propose Interaction Protocol Specification* to process the bidding of the auction. First, it sends a *cfp* message to all known seller agents (SA). A seller agent decides whether it will accept the offered price based on its target retail price and stock number. If the seller agent decides to sell the goods, it sends a *proposal* message to auction server. An *accept-proposal* message, which comes with a signed contract, is sent out when the auction server receives the first valid proposal. Other incoming proposal message will subsequently be rejected with a *reject-proposal* message. The auction server informs the buyer agent of the auction result and reports the contract information to MOA. If no proposal message received, the auction expires after timeout period and an appropriate failure message is sent to the buyer.

3.3 Content language.

The content language is a language used to express the content of messages exchanged between agents. The FIPA communication infrastructure allows agents to communicate using any mutually understandable content language as long as it satisfied a few minimal criteria as a FIPA compliant content language [FIPA, 2003]. Published FIPA specifications provide a library of registered FIPA compliant content language, including FIPA-SL, XML and RDF. A good content language should be able to express rich forms of content and can be efficiently processed and fit well with existing technology. XML language, which adopted by TAC system, can be used to express messages in a conversation and has lots of parsing tools available. However, as a representation language, XML provided essentially a mechanism to declare and use simple data structures and thus leaved much to be desired as a language of expressing complex knowledge. The enhancements to basic XML, such as XML Scheme, addressed some of the shortcomings, but still did not result in an adequate language for representing and reasoning about the kind of knowledge essential to realizing the semantic web vision.

Our TAGA system used RDF as content language for agent communication. The benefits of adopting a stronger semantically rich content language like RDF is that it facilitates a higher-level of interoperability between agents, by agreeing on how meaning is conveyed, it makes it simpler for applications to share meaningful content. The actions exchanged in TAGA include:

- **Statements:** the price of the hotel 1 in day 3 is $100;
- **Requests:** create an airline auction instance;
- **Contracts:** if the Travel Agent TA1 successful organized the travel package, customer Joe will pay $400 to TA1, else, TA1 pay $200 compensation to Joe.
- **Policies:** to win the contract of the customer Joe, the travel agent must have reputation better than average (reputation is computed by comparing customers with fulfilled travel package vs. all served customers).

We have defined the ontology for use as a FIPA-compliant content language. In addition to the basic required classes (e.g., Agent, ACLMessage, Service, etc.) and necessary expressive requirement (such as Proposition, Action, and Reification), our ontology provides supports for expressing rules, queries and responses to queries.

We are currently revising the TAGA framework to use OWL [Dean, 2002] as the content language. Compared with RDF, OWL has a well-defined model-theoretic semantics as well as an axiomatic specification that determines the intended interpretations of the language. OWL is unambiguously computer-interpretable, thus making it amenable to agent interoperability and automated reasoning techniques. The benefit of adopting a stronger semantically rich content language like OWL is that it facilitates a higher-level of interoperability between agents. By agreeing on how meaning is conveyed, it is simpler for applications to share meaningful content. Further more, as a semantic web language, OWL is designed to fit into and integrate with web-based information and service systems and has the potential to be a widely accepted and used representation language, enhancing the potential for interoperability among many systems.

4. Discussion

In this section we will briefly discuss several additional design issues we have addressed in TAGA.

Service description and matching. FIPA agents are associated with one or more FIPA platforms, each of which offers a set of agent services including a Directory Facility (DF) agent that handles service registration, deregistration and matching. The register content in the DF include service information like service type, owner. However, more specific service information may also be useful when searching for agent services. For example, a customer may want a booking in a hotel with at least three star rating, is close to public transportation, offers breakfast, and accepts VISA card payments. This can be achieved with the use of DAML-S [DAML-S, 2002] profile. In TAGA, every travel service provider describes its service process model with DAML-S language and publishes as a web page. It covers basic service information like address, phone number and service interface information. For example, a hotel may describe booking service as: customer name, payment methods, travel date as input; reserve number as output; the effect of booking is one room occupied at the travel date. The travel agent, who is responsible for organizing travel package, is able to contact with customer agent and related service agents and finds the best match. First the travel agent loads the DAML-S parsing rule and planning rules into the build-in XSB [Sagonas, 1994] reasoning engine. It then loads service agents' DAML-S profiles and customer's personal profile. The best matching service providers are selected and a most profitable travel package is composed dynamically.

Implementation comments. TAC relies on a few centralized market servers to handle all interactions and coordination, including service discovery, agent communication, coordination, and game control. In contrast, TAGA framework uses a distributed peer-to-peer approach based on standard agent languages, protocols and infrastructure components (FIPA [FIPA, 2003], Agentcities), emerging standards for representing ontologies, knowledge and services (RDF, DAML+OIL, DAML-S [DAML-S, 2002]) and web infrastructure (e.g., Sun's Java Web Start). Several FIPA platform implementations are currently used within TAGA, including Jade [Bellifemine, 2001] and AAP, demonstrating agent interoperability. Our current demonstration system allows new users to dynamically join a running game at any time. A dummy agent implemented in JADE can be downloaded and run to instantiate a new TA agent. A simple GUI allows the user to set operating parameters or the java code can be modified or extended. A set of web based monitoring services allow one to see the status of a game, examine messages being sent, lookup the reputation of agents, etc.

Contribution. We see two contributions in our work. First, TAGA provides a rich framework for exploring agent-based approaches to e-commerce like applications. Our current framework allows users to create their own agents (perhaps based on our initial prototype) to represent a TA, SA and to include it in a running game where it will compete with other system provided and user defined agents. We hope that this might be a useful teaching and learning tool. Secondly, we hope that TAGA will be seen as a flexible, interesting and rich environment for simulating agent-based trading in dynamic markets. Agents can be instantiated to represent customers, aggregators, wholesalers, and service provides all of which can make decisions about price and purchase strategies based on complex strategies and market conditions.

5. Conclusions and future work

Travel Agent Game in Agentcities (TAGA) is a framework that extends and enhances the Trading Agent Competition (TAC) system to work in Agentcities, an open multiagent systems environment of FIPA compliant systems. We hope that TAGA will serve as an experimental testbed for several communities of users.

First, it provides an environment, which can be used to explore aspects of multiagent systems technology based on the mature, published FIPA standards. Research on MAS technology is best done with in a rich yet easily understood problem domain. We have found that the travel agent scenario as originally put forth by TAC provides both the richness as well as accessibility, especially when opened up to be peer-to-peer. We are using TAGA as a testbed for research on the use of semantic web languages (e.g., RDF and OWL) as content languages and as service description languages. Future work is planned in adding more sophisticated negotiation and ontology mapping to our TAGA environment.

Second, we hope that TAGA could serve as an interesting framework and testbed for experiments with automated markets and trading. By adding autonomous service provide agents (e.g., for hotels) one could experiment with a dynamic market with both "shopbots" and "pricebots" [Greenwald, 1999] or investigate the role of intermediation in the form of agents performing a wholesale function.

Third, we hope that others will find TAGA useful as a test, demonstration and teaching environment, both in technology classes focused multi-agent systems, FIPA standards or the semantic web and in business or e-commerce classes focused on automating commerce and trading, auctions or agent-based simulations.

The Agentcities project is exploring the delivery and use of agent-based services in an open, dynamic and international setting. We are working to increase the integration of TAGA and emerging Agentcities components and infrastructure and will include agents running on handheld devices using LEAP [Bergenti, 2001]. We are also working to enhance the ontologies which underlie TAGA and to move them from RDF and DAML+OIL to the W3C's Web Ontology Language OWL.

References

[Anthony, 2001] P. Anthony, W. Hall, V.D. Dang, and N. Jennings: Autonomous agents for participating in multiple online auctions. In Proc. of the IJCAI Workshop on EBusiness and the Intelligent Web, Seattle WA, USA, July 2001.

[Bellifemine, 2001] F. Bellifemine, A. Poggi, G. Rimassa, Developing multi agent systems with a FIPA-compliant agent framework, in Software - Practice And Experience, 2001 N31, pp. 103-128

[Bergenti, 2001] F. Bergenti and A. Poggi, LEAP: a FIPA Platform for Handheld and Mobile Devices, ATAL, 2001.

[Dale, 2002] J. Dale, S. Willmot, and B.Burg: Agentcities: Challenges and Deployment of Next-Generation Service Environments. Proc. Pacific Rim Intelligent Multi-Agent Systems, Tokyo, Japan, August 2002.

[DAML-S, 2002] The DAML Services Coalition (alphabetically Anupriya Ankolenkar, Mark Burstein, Jerry R. Hobbs, Ora Lassila, David L. Martin, Drew McDermott, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terry R. Payne and Katia Sycara): DAML-S: Web Service Description for the Semantic Web, The First International Semantic Web Conference (ISWC), Sardinia (Italy), June, 2002.

[Dean, 2002] M. Dean and Guus Schreiber (eds): OWL Web Ontology Language 1.0 Reference. W3C Working Draft. [Eriksson, 2002] Joakim Eriksson and Sverker Janson: The Trading Agent Competition - TAC 2002, ERCIM News, pp51, October 2002.

[Eriksson, 2002] Joakim Eriksson and Sverker Janson.The Trading Agent Competition - TAC 2002. ERCIM News, 51, October 2002.

[FIPA, 2003] FIPA agent standards FIPA Interaction Protocol Library Specification, http://fipa.org/specs/fipa00025/.

[Greenwald, 1999] Amy R. Greenwald and Jeffrey O. Kephart: Shopbots and Pricebots, International Joint Conferences on Artificial Intelligence, Stockholm, August 1999.

[Greenwald, 2001] Amy Greenwald and Peter Stone: Autonomous Bidding Agents in the Trading Agent Competition, IEEE Internet Computing, March/April 2001.

[Greenwald, 2003] Amy Greenwald (ed.). The 2002 trading agent competition: An overview of agent strategies. AI Magazine, to appear

[O'Brien, 1998] O'Brien, P and Nicol, R, FIPA - Towards a Standard for Software Agents. In: BT Technology Journal, Vol.16:3, pages 51-59, 1998.

[Sagonas, 1994] Kostantinos Sagonas, Terrance Swift, and David S. Warren: XSB as an efficient deductive database engine, In ACM Conference on Management of Data (SIGMOD), 1994.

[Stone, 2000] Peter Stone and Amy Greenwald: The First International Trading Agent Competition: Autonomous Bidding Agents, Electronic Commerce Research Journal pp1-36, 2000.

[Wellman, 1999] Michael P. Wellman and Peter R. Wurman. A trading agent competition for the research community. IJCAI-99 Workshop on Agent-Mediated Electronic Commerce, Stockholm, 1999

[Wellman, 2001] Michael P. Wellman, Peter R. Wurman, Kevin O'Malley, Roshan Bangera, Shou-de Lin, Daniel Reeves, and William E. Walsh: A trading agent competition. IEEE Internet Computing, 5(2), pp43-51, March/April 2001.

[Wellman, 2002] Michael P. Wellman, Amy Greenwald, Peter Stone, and Peter R. Wurman: The 2001 Trading Agent Competition, Fourteenth Innovative Applications of Artificial Intelligence Conference (IAAI-2002), pp935-941, Edmonton, August 2002.

[Willmott, 2001] Willmott, S., Dale, J., Burg, B., Charlton, P. and O'Brien, P., Agentcities: A Worldwide Open Agent Network. In: AgentLink News, Issue 8, November 2001.