

Considerations on a New Software Architecture for Distributed Environments Using Autonomous Semantic Agents

Atilla Elci and Behnam Rahnama

*Dept. of Computer Engineering, Eastern Mediterranean University, Famagusta, TRNC, Turkey
{atilla.elci, behnam.rahnama}@emu.edu.tr*

Abstract

Distributed processing environments such as that of a traffic management network system (TMS) can be implemented easier, faster, and secure and perform better through use of autonomous semantic agents (ASAs). For an ASA can then be realized as a semantic Web Service, a whole TMS is easily implemented through a collection of semantic Web services agents arranged according to the topology of the traffic network. It would suffice to develop a generic ASA Web service class, instantiate individual ASAs from it in numbers as required one per junction, and supply specific intersection data in semantically-enriched representation to each. Should advanced information support and control services be required, one of the ASAs may be configured slightly differently in that it acts as the operational overseer and repository for aggregated data and ASA class code. Once created, this Facilitator ASA knows the topology of the whole traffic network, identifies each intersection (and its associated ASA), can interrogate and instruct individual ASAs. Aspects of ASA design, operation, and application development using ASAs are taken into consideration. Simulations show high performance and the benefits of load distribution using ASAs.

1. Introduction

Systems with distributed control such as a traffic management system (TMS) and factory floor production line can be implemented easier, faster, and secure and yet perform better through use of autonomous semantic agents (ASAs). Let's first consider a scenario in which this is realized; and, then work out bases on how that all could become true.

In the distributed processing environment of a TMS, management of each traffic junction is taken over by an agent with semantic intelligence. An ASA executes its business logic while operating on its policy store, maintains junction parameters, and interacts only with its

neighboring nodes in the network topology. In all its endeavors, an ASA benefits from an understanding of its immediate vicinity gained through semantic technology and business area ontology. Its functioning is for the most part semantic, autonomous, and guided by globally imposed business policies.

For an ASA can be realized as an intelligent agent, a TMS is easily implemented through a collection of semantic Web service agents arranged according to the traffic network topology. In preparing a TMS application system, it would suffice to develop a generic ASA Web service class, instantiate ASAs from it in numbers as required one per junction, and supply each with specific intersection data in semantic representation.

One of the ASAs is configured slightly differently in that it acts as the operational overseer and repository for aggregated data. The central ASA knows the whole traffic network, identifies each intersection (and its associated ASA), can interrogate and instruct individual ASAs.

TMS is a hot topic in artificial intelligence, mathematics, and graph theory in computing departments currently. Such systems are typically implemented to control traffic lights and balance the traffic in city roads. However, people are interested in finding factual info such as the best path to their destination in near real-time. Current TMSs are able to meet that requirement only through a dedicated add-on traffic information system, which is quite costly. Modality and utilization of ASAs for information provisioning in a TMS is taken elsewhere [4, 5]; nor TMS taxonomy / ontology will be considered here.

An ASA would be capable of intelligent interaction with neighbors and people in the traffic, thus both gathering and responding to information requests. Let's suppose that a service request is received by an ASA. That ASA considers the request, either responding to the requester with the required information, or, not having sufficient data to compose a response, forwarding the request to its (connected) neighbors. An ASA receiving the forwarded request behaves as the forwarding requester did: responding or forwarding. Finally, the ASA at the source node which originated the request, collates

all responses received, composes an answer, and provides it to the requester-user. This is pictorially displayed in Figure 1.

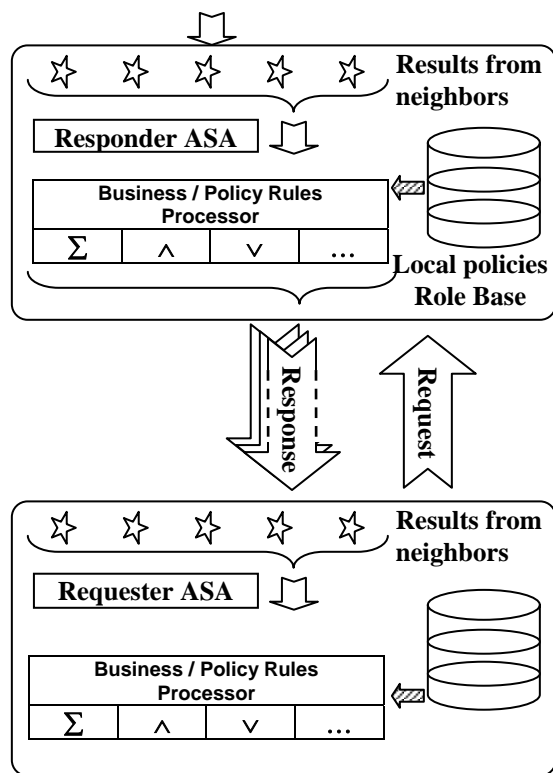


Figure 1. Request propagation and response collation in ASAs.

How effective would such a message - passing modus operandi be? An ASA forwards an unanswerable request to all of its neighbors. Additionally, responses to a request would have to be broadcast back similarly. Could this cause an avalanche of messages, consequently bugging down the network through excessive message traffic? We will look in to that and other aspects of ASA network in the following sections.

2. ASA: theory and practice

An ASA is an intelligent agent which essentially operates as a semantic Web service (SWS, [7]). It functions by employing Web services protocols and carries out its built-in business logic. ASA description, registration, and discovery are same as for a Web Service [12].

An ASA is additionally made context-aware through semantic Web technology. It uses the full range of semantic Web protocol stack [12]. Consequently, it is

capable of network and neighbor discovery and employs ontology and rule-based reasoning as described above.

It may have interface with people and vehicles in the network to display information and possibly to receive queries; NLP capability may be required depending on the application.

2.1. ASA: physical entity

An ASA is an intelligent agent disguised as a Web service. As Web services go, an ASA could be practically executing on any platform. Depending on processing requirements of the job at hand, it may be implemented at any complexity from a hosted Web service sharing same platform with many others, to a self standing processing node with own server and storage, and up to a server farm performing as one of the nodes in a semantic grid network.

2.2. ASA-based system development

Implementing a system based on ASAs may be carried out in the following steps:

- i. Preparing a generic class definition of ASA having the required business functions and also fitted with capability to carry out site management, inter-node communication, and interaction with external actors;
- ii. Instantiating one ASA for each node (such as, for each junction in TMS, manufacturing stop at production line, and processing joint in a semantic grid system);
- iii. Initializing each ASA with its node configuration, general and site-specific business policy directives, processing rules in terms of semantic representation, and their relevant ontologies;
- iv. Preparing a Facilitator ASA object with supervisory capability;
- v. Initializing the Facilitator ASA with network topology, business logic, system overall and node policies, administrator interface, and all concerned ontologies. Should it be required to instantiate ASAs during system execution, such as adding new junctions to an already operating TMS, the Facilitator also include generic class definition of ASA.

Then, the system can be started rolling.

2.3. ASA interaction

ASAs establish a virtual network of open, cooperative, and multi-agent system. Developing interaction rules and protocols among agents is one of the central research topics in multi-agent systems. For cooperative agents, protocols are required so that agents can achieve a common goal if they follow the protocol.

Similarly, in the case of competitive or selfish agents, we need to design protocols so that a socially desirable outcome can be achieved, even if agents act selfishly [14]. However, in the case of ASAs, as they are built to satisfy the specific business function, agreeing on a common goal and cooperation towards that end is a given. On the other hand, means of carrying out the interaction requires further development. It is thought that collaboration & interaction could be facilitated through a standard agent communication language (ACL, [17]) devised according to social commitment and asynchronous collaboration views [18, 19].

Whereas above treatise is suitable for inter-agent collaboration, ASAs may as well be setup in situations where collaboration & interaction with people is required (such as in TMS with traffic query and information dissemination features). In such systems, where robots, agents, and people will be coordinated to achieve shared goals, distributed collaboration and interaction (DCI, [16]) and proxy-based integration architectures [15] may prove helpful. On the other hand, server-side management of intelligent agent-based Web services [20] such as that provided by ASAs was not favored in this study.

ASAs in this study interact only with their “neighbors”, Facilitator, and people as explained in the introduction.

According to a recent study, the fact that ASAs are designed as semantic agents is quite helpful in improving system interoperability [10].

2.4. ASA safety & security

The fact that ASAs and the Facilitator are generated through secure means may not suffice to protect them against malicious attempts from environment. Security in an ASA-based multi-agent system (MAS [22]), should cater for unaffected operation of both the agents and the platforms supporting them. For this purpose we propose to use an agent shielding type of an encapsulation approach where an agent executes in an isolated thread. This approach also caters for secure messaging among agents. Furthermore, an ASA carries security policies to avoid unauthorized access.

Semantic nature of an ASA in its internal operation and in carrying out relations with outside world appears to be its best protection. For an attacker to entice an ASA to respond, the former had to have obtained relevant ontologies to be able to mimic a “neighboring” ASA.

3. Facilitator ASA

A network of ASAs is capable of self management and community service by consensus in the sense of

agent-based peer-to-peer service networks and common-goal cooperative agents systems [13, 14]. Therefore a central authority to control and command is not required for the most part. For example, a recent study was conducted utilizing adaptive traffic lights at junctions with no central coordination or connections to others [8]. The method works well when traffic is light, but in heavy traffic it fails as the lights change constantly due to threshold being exceeded on all directions. Lack of global strategy and adaptive coordination among intersections at real time are likely prime reasons for the failure. A similar project was developed and tested in the lab by a project team in the Technology Development Center (TEKMER), Eastern Mediterranean University (EMU); subsequently reported to have been applied in the field in Gazimagusa, TRNC [9, 23].

Once initialized, an ASA would survive on its own. An ASA then has two distinct parts: (a) Robot ASA for self-sustenance and business logic implementation; (b) Context discovery and awareness.

As ASAs would behave autonomous yet obedient, there is no reason to monitor them except for perhaps to track system performance. Even for that, ASAs could be fitted with pre-installed rule base commanding them to report significant performance factors. A central capability is conceivable in order to administer global goals and oversee compliance and performance. Such is possible through creation and infusion in to the network a special purpose ASA which may be called as the Facilitator ASA for reasons given below. The Facilitator is capable of performing as a regular ASA for all related service requests. Additionally, it has central registry, overseeing, and administrative interface responsibility. These are taken up below.

Facilitator carries the ASA generic code and knows how to instantiate one. It initializes the newly created ASA with global and node-specific goals, policies and rules base; delegates it with node function and records it at the registry. The registry is a central repository of SWSs for name resolution, description, service reclamation, and discovery. What is aimed by that is for all aspects nearly dynamic discovery of agent-based semantic Web services [6]. The registry may be hosted with the Facilitator.

Facilitator oversees the system operation. It knows the whole of the network, nodes and associated ASAs, policy and rule body. It listens to the network activity, can interrogate ASAs and monitors performance. It could take corrective move by changing policies and node parameters, creating new nodes and phasing out existing ones. In performing overseeing function, Facilitator interacts with the administrative user as the prime interface, executes requests, and returns feedback.

If a system has to have a facilitator, then probably measures should also be in place to recover from the loss of it. This would involve discovery of the network, and reconstruction of the registry, rules base, and policy set. It may be possible for the system to recover by first deciding on which ASA could take over through a distributed leader election exercise.

4. MAS: Applications of ASAs

The novel idea of treating ASAs as web services and the adaptability of an ASA-based system was described above using a facilitator, as an example.

Certain distributed control environments readily lend themselves to implementation through use of a system with multiple intelligent agents such as ASAs [19, 21]. TMS scenario is mentioned above; similarly, factory floor production line, and grid processing systems [11] can be taken up.

In all such cases, client-server solution is possible, yet that creates a bottleneck at the central site by centralizing control. The resultant application mode tends to become “command & control driven”. On the contrary, SWS (a.k.a., ASA) approach distributes load to network by delegating capability to processing nodes to decide on own considered motion. Using a distributed infrastructure without strict central control accords us the possibility of enlarging the system at will and the flexibility of network topology modification. For agents are operating in concurrent and cooperating manner, exploiting parallelism in solving complex distributed problems is another virtue of this new software architecture.

Taking advantage of semantic Web technology in representing network topology, node configuration, business policy, and performing objective functions, accords an ASA with power to reconcile with its environment. Furthermore, ASAs can cooperate in resolving their share of information request problems through taking advantage of other agents’ semantic data.

An application of the ASA-based architecture to a multi-nodal traffic management system is interesting and shows the benefits of load distribution using ASAs. This aspect we take up below.

5. Message Passing Performance

While forming an answer in response to a user query, ASAs cooperate in broadcasting the query to the network and responses back to the requester as discussed above. It was feared that the message broadcasting modus operandi might cause excessive load on the network thus invalidating any advantages gained. In order to determine

exact behavior and if possible work out an upper bound on the number of messages, simulation experiments were carried out. Simulations of network models showed close to linear incremental load on the communication network. For example, for a typical network representative of real traffic junction system, message load increases linearly with the number of intersections.

Simulation of network models of sizes one up to ten ASAs for traffic network was conducted using Message Passing Interface (MPI, [1]) on VMware v.4.5.1 Parallel Virtual Machine. MPI is a message-passing middleware library facilitating communication for exchanging data and synchronization of tasks among processors in a distributed-memory parallel processing environment [2, 3].

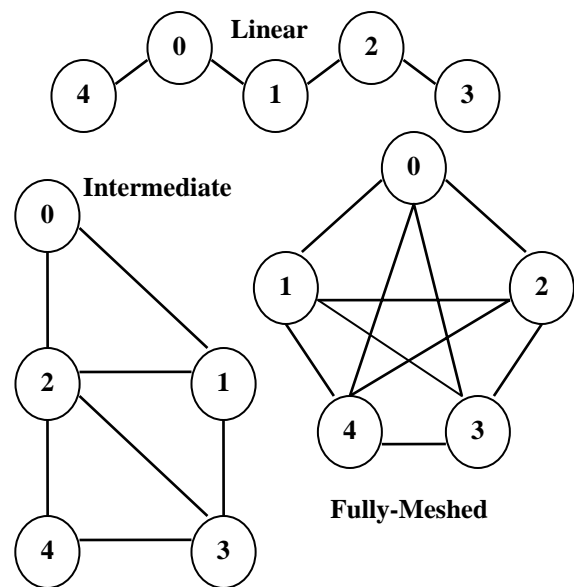


Figure 2. The Three traffic network models considered: Linear, Intermediate, and Fully-Connected Mesh.

Three different traffic network models were considered based on topology: linear, intermediate, and mesh. Linear network is akin to a railroad where stations line up along tracks and essentially there are no intersections. The linear model is the simplest of the three; although unrealistic, it is included here as a boundary / best case. The mesh model is the most complicated traffic network consisting of fully-connected junctions; that is, all intersections connect to all others, definitely quite unrealistic as traffic networks come. The mesh model is included here as the worst-case topology. On the other hand, it’s clear that the intermediate model is quite realistic. All three models are being displayed in Figure 2.

Table 1. Maximum number of messages due to message propagation in all models for up to 10 ASAs.

Number of Nodes	Number of Messages		
	Linear Model	Intermediate Model	Fully Connected Mesh Model
1	0	0	0
2	1	1	1
3	2	3	3
4	3	5	6
5	4	7	10
6	5	9	15
7	6	11	21
8	7	13	28
9	8	15	36
10	9	17	45

Table 1 depicts the maximum total number of messages passed due to propagation of a single message from one originator node. Obviously these are worst-case scenarios. The numbers are given for all three models. The same is charted in Figure 2. The growth trend is clearly linear for linear and intermediate models. In the case of the mesh model, it grows similar to Fibonacci series, such as, maximum number of messages for size n is the sum of n-1 and that of size n-1 where base cases are 1 and 0. In reality, results would be more favorable, because a node having received a message would not send it to the links it received it from; also if the node can respond to the message there is no reason for forwarding it further. These shortcuts will likely reduce message propagation traffic drastically.

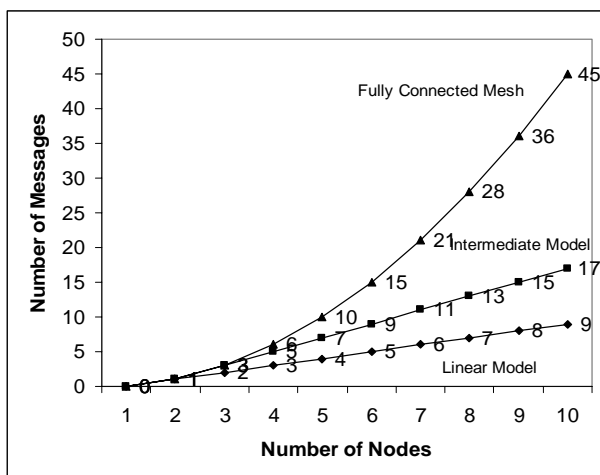


Figure 2. Maximum number of messages due to message propagation in all models for up to 10 ASAs.

Sample MPI run for fully-connected mesh model with 5 ASAs is given in Figure 3.

```
C:\Program Files\MPICH2\bin>mpiexec -n 5 TrafficMax.exe
I am agent 1 of 5
Middle agents:1
Receiving requests for available place
I am agent 2 of 5
Middle agents:2
Receiving requests for available place
I am agent 3 of 5
Middle agents:3
Receiving requests for available place
I am agent 4 of 5
Receiving requests for available place Agent:4
Responder: Receiving # of Messages from middle/requester agents
I am agent 0 of 5
Request for info available in the last Coverage Area Agent:0
Sending requests for available place to other Agents
Sending requests for available place to other Agents
Sending requests for available place to other Agents
Agent 0:4
Agent 2:2
Agent 3:1
Agent 1:3
Finalizing, Press any key
Network Complexity on 5 agents: 10
```

Figure 3. Sample MPI run for fully-connected mesh model with 5 ASAs.

6. Conclusions

In this study we looked into a new general class of software architecture using semantic web technologies in implementing distributed systems. Specifically, we drew examples from a set of distributed systems having replicated processing-cum-control nodes with or without central control such as TMS, semantic grids, and factory floor production line operation.

The ASA-based software architecture discussed here distributes processing load to network by delegating capability to individual nodes. Taking advantage of semantic Web technology, nodes can behave autonomously yet remain committed to local and global goals. Using a distributed infrastructure without strict central control allows ease in enlarging the system at will and flexibility of network topology modification. For agents are operating in concurrent and cooperating manner, exploiting parallelism in problem solving is another aspect of this new architecture.

We are currently working on implementation issues relating to central control, security, and fault tolerance.

Acknowledgement: We thank for their support in this project the Internet Technology Research Center and the parallel computing setup of the Department of Computer Engineering, Eastern Mediterranean University.

7. References

- [1] B. Rahnama and A. Kostin, "MPI: A Message Passing Interface Standard", *Technical Report*, Dept. Computer Engineering, EMU, TRNC, December 2004.
- [2] Gropp, W., E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-Passing Interface, Second Edition*, MIT Press, 1999.
- [3] Snir, M., S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra, *MPI: The Complete Reference, Volume I: The MPI Core, Second Edition*, MIT Press, 1998.
- [4] B. Rahnama and A. Elci, "Traffic Control using Autonomous Semantic Robots", *Technical Report*, Dept. Computer Engineering, EMU, TRNC, January 2005.
- [5] B. Rahnama and A. Elci, "Traffic Info Gathering and Dissemination Using Interconnected Autonomous Semantic Robots as Junction Managers", accepted for *Proc. TMT 2005-9th International Research / Expert Conference "Trends in the Development of Machinery and Associated Technology"*, Univ.s of Zenica, Politecnica de Catalunya, and Bahcesehir, 26-30 September 2005, Antalya, Turkey
- [6] K. Sycara, M. Paolucci, J. Soudry, and N. Srinivasan, "Dynamic Discovery and Coordination of Agent-Based Semantic Web Services", *IEEE Internet Computing, Volume: 8, Issue: 3*, May-June 2004, pp.: 66-73.
- [7] Dogac, A. et al., Artemis Deliverable D3.1.1.2: Review of the State-of-the-Art- Semantic Web and Web Service Semantics, *EU FP6 Artemis Project (IST-2103 STP Artemis)*, METU-SRDC, Ankara, Turkey, 7 April 2004.
- [8] K. Patch, "Adaptive Lights Organize Traffic", *Technology Research News, January 26 / February 2, 2005*. http://www.trnmag.com/Stories/2005/012605/Adaptive_lights_organize_traffic_012605.html. See C. Gershenson's page: <http://homepages.vub.ac.be/~cgershen/sos/SOTL/SOTL.html>.
- [9] "Traffic Signalization System Equipped with Detectors", *Doğu Akdenizli, EMU Alumni News, Vol.1 No.1, January 2005*, p.:37.(in Turkish).
- [10] X. Su, S. Hakkarainen, and T. Brasethvik, "Semantic Enrichment for Improving System Interoperability", *Proc. of 19th ACM Symposium on Applied Computing (SAC'04)*, Nicosia, Cyprus, March, 2004, ACM press.
- [11] Foster, I. and C. Kesselman (editors), *The Grid: Blueprint for a New Computing Infrastructure, Second edition*, Elsevier, 2004.
- [12] W3, Web Services Activity: <http://w3.org/2002/ws/>; and Semantic Web Activity: <http://w3.org/2001/sw/>. Semantic Web Protocol stack: <http://www.w3.org/DesignIssues/diagrams/sw-stack-2002.png>. Last visited on Feb. 28, 2005.
- [13] Y. B. Udupi, P. Yolum, M. P. Singh, "Agent-Based Peer-to-Peer Service Networks: A Study of Effectiveness and Structure Evolution", *Proc. Third International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS 2004 - Volume 3*, July 2004, ACM.
- [14] M. Yokoo, "Protocol / Mechanism Design for Cooperation / Competition", *ibid Vol. 1*.
- [15] P. Scerri, D. Pynadath, L. Johnson, P. Rosenbloom, M. Si, N. Schurr, and M. Tambe, "Teamwork: A Prototype Infrastructure for Distributed Robot-Agent-Person Teams", *Proc. Second International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS 2003*, July 2003, ACM.
- [16] C. Martin, D. Schreckenghost, P. Bonasso, D. Kortenkamp, T. Milam, and C. Thronesbery, "An Environment for Distributed Collaboration among Humans and Software Agents", *ibid*.
- [17] M. Verdicchio and M. Colombetti, "Semantics and Pragmatics of Interaction: A Logical Model of Social Commitment for Agent Communication", *ibid*.
- [18] N. Fornara and M. Colombetti, "Semantics and Pragmatics of Interaction: Defining Interaction Protocols using a Commitment-Based Agent Communication Language", *ibid*.
- [19] D. Weyns and T. Holvoet, "Synchronous versus Asynchronous Collaboration in Situated Multi-Agent Systems", *ibid*.
- [20] L. Ardissono, D. Cardinio, G. Petrone, and M. Segnan, "Business Processes and Conversations: A Framework for the Server-Side Management of Conversations with Web Services", *Proc. 13th International World Wide Web Conference on Alternate Track Papers & Poster, May 2004*, ACM.
- [21] T. R. Payne, M. Paolucci, R. Singh, and K. Sycara, "Facilitating Message Exchange through Middle Agents", *Proc. of AAMAS 2002, Part 2*, July 2002, ACM.
- [22] V. Varadharan and D. Foster, "A Security Architecture for Mobile Agent Based Applications", *World Wide Web: Internet and Web Information System*, 6, 93-122, 2003.
- [23] M. Yakup, M. Abbaz, and S. N. Bayindir, "Alternative Solution in Signalization Using PLC and Vehicle Detection", *Proc. Cagdas Yasam ve Trafik Sempozyumu*, 2-4 March 2005, KTMMOB, Lefkoşa. (in Turkish).