Information Discovery and Fusion: Semantics on the Battlefield

Katia Sycara and Massimo Paolucci
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
katia@cs.cmu.edu

Michael Lewis
School of Information Sciences
University of Pittsburgh
Pittsburgh, PA 15260
ml@sis.pitt.edu

Abstract *To help translate information superiority to decision superiority (i.e. to rapidly arrive at better decisions than adversaries can respond to), many scientific, technical and technological challenges must be addressed. The most critical of those are information fusion and management at different levels, communication, planning and execution monitoring. Multiagent infrastructures allow information producers and users to discover one another and establish direct links. The robust, decentralized Infosphere which results, can be stood up rapidly and ensures that information of the specified types will be delivered to the right users under the right conditions. Problems involving discovery of and interoperation with new information sources are central to realizing such dynamic battlefield networks. In this paper we introduce a semantic description language and discovery technologies which may hold the keys to building large scale heterogeneous battlefield networks.*

1 Introduction

Today's commanders are faced with an operational environment that is fast moving, uncertain, and flooded with information. The number of threats are bewildering and in multipolar conflicts with new and potentially shifting alliances. Therefore, just maintaining situational awareness may be a significant challenge. Add to this the new agility of rapid deployments, joint operations, and an information infrastructure that must be hastily erected in hostile environments and current C4ISR technologies are no longer adequate. Visionary concepts including Network Centric Warfare, ForceNet, the Joint Battlespace Infosphere (JBI) [11] and the Expeditionary Sensor Grid have been proposed to automate these functions but none are yet operational.

The basic problems for users of current C4ISR identified by the Scientific Advisory Board on the JBI [11] were:

- Information overload
- Lack of interoperability
- Immaturity in [higher level] fusion
- Limits in display technology
- Legacy tactics, techniques, and procedures

Additional design problems posed in building a JBI-like infrastructure are:

- Rapid network stand-up
- Information source discovery
- Dynamic reconfiguration

Meeting these challenges requires changes in hardware, software, and the way information is represented. We have proposed [7] an agent based infrastructure to implement the publish/subscribe mechanism used by JBI to manage information. This paper deals with representational issues and illustrates how DAML-S (Darpa Agent Markup Language for Web Services), a semantic description language, could be used by agent accessible battlefield networks to address C4ISR problems. The Web Services ontology was chosen for this purpose because its tripartite distinction of *profile* describing a provider's capability, *model* describing how the provider operates, and *grounding* describing how to access the provider is a good fit for the types of contextual information that would be needed to integrate battlefield information from heterogeneous sources.

Consider a noncombatant evacuation operation (NEO) conducted in a central Asian country, called Irat, undergoing a revolution. There are massive amounts of information available from communication intercepts, satellite imagery, weather reports, street maps, floor plans, and maps showing gas and electric lines, as well as dynamic information sources such as spotters, air dropped sensors, and surveillance from aircraft as they enter and leave the area of interest. The commander wants to find a safe route from the US embassy where the evacuees are massed to an airport where they can be flown out of the country. His problem is that the massive amounts of

---

[1] This research was supported by AFOSR contract number F49640-01-1-0542.

potentially relevant information is useless to him until it is aggregated and ordered in such a way that questions about possible routes can be answered. This is the *information overload* problem. After highlighting a possible route on the map he tries to match it with ground tracks from a GMTI report but cannot because the map is not geo-referenced. This is one form of the *interoperability* problem. A satellite photo showing a concentration of rebel forces and reports from deployed vibration sensors, both available on the commander's notebook, suggest that the rebels may be moving toward the road he plans to use but there is no way to automatically combine these reports to warn the commander. This is an example of *immaturity in fusion*. When the commander does look at the vibration reports he can't find anything useful because the sensor data is presented in rows with unique sensor id, power level of transmission, ATR code, and GPS coordinates. The company building the sensors has provided exactly what was requested. Now there is no convenient way to merge these readings with the satellite photos or maps to make sense of them. This illustrates some of the *limits in display technology*. Later, when the evacuation convoy encounters the rebel forces on the road the commander calls in close air support. His notebook shows the ground track the pilot is targeting but he must call an abort and a second run because a visual identification of the target being attacked is required by regulations. Requiring an ambiguous visual inference when an unambiguous electronic identification is available illustrates the problem of *legacy TTP*. Finally there are all the behind the scenes problems of connecting all these information sources together. When the commander first moved into the region his unit had a preexisting information infrastructure. Joining this with the older equipment at the US Embassy, the French electronics used by the friendly Irat forces, and the experimental network of air-dropped sensors required an extended flurry of communications to get addresses, protocols, and frequencies sorted out and registered (*stand-up*). The commander did not find out about a stringer from the Embassy who was right on the evacuation route until the man ran out of the building to join the evacuees just as they encountered the first of the rebel forces (*discovery*). Ironically, another chance to avoid the rebels had come just a few moments before when a surveillance aircraft passed overhead. The plane was listed as unavailable in the asset database so no effort had been made to contact it even though it had been fully available and taskable during its brief overflight (*dynamic reconfiguration*).

Although these problems are daunting and require comprehensive improvements much of the difficulty can be traced back to an absence of semantics for the information being managed and the lack of mechanisms for discovering and assimilating new information sources. These problems are closely related because it is only important to discover information that you need yet you must know what the information is about (semantics) before judging it. The problems of matching GPS and map coordinates for the GMTI and vibration sensors and of inferring the relation between forces in a photograph and a sequence of vibration readings all depend on semantics of spatial location. It appears entirely plausible that a description language such as DAML-S can express most of what will be needed to realize practical battlefield networks.

2 DAML-S

A battlefield network consists of many information sources including in-theater sensors, platforms, and human intelligence and remote information sources such as archival intelligence and satellite data. The purpose of the network is to "provide individual users with the specific information required for their functional responsibilities…at every echelon of command" [11]. The JBI proposes to do this using a publish/subscribe mechanism in which information providers "publish" information, which they produce on the network making it available to those information requesters who have "subscribed" to that publisher's information. JBI information is published in the form of objects containing both data and metadata describing it. The metadata is intended for use in matching information objects with subscriptions. DAML-S instantiates the "XML technologies" alternative proposed for JBI objects [11] and offers a more detailed picture of the scope of descriptions, grounding, and approaches to matching which will be needed to make a battlefield network work. One way to realize effectively the publish/subscribe paradigm in the open battlefield environment is to model information sources as providing services (service providers). Battlefield planning systems can be modeled as service requesters.

In DAML-S Information requesters need to be able to find appropriate information providers. In order to do that they need to be able to describe and register their own capabilities with public registries, as well as locate other information providers with specified capabilities. Capability information is crucial for participants in a battlefield network to locate each other on the basis of the information that they can provide rather than on the basis of their name or of the name of their unit.. In addition, an information requester should have information on how to interact with the provider, which means that it should know the interaction protocol of the provider, and binding information. Most crucially, this information should allow the requester as well as the provider to decode the information exchanged, so it should specify not only the format of the messages to exchange or the remote procedures to call, but also the semantic type of the information to exchange. This view is embraced by DAML-S [8] which defines a DAML [3] ontology for the description of Web Services that attempts to bridge the gap between an infrastructure of Web Services based essentially on WSDL (web services description language) [2] and SOAP (simple object access protocol) [12], and the

Semantic Web [1]. In other words, DAML-S bridges the gap between the specification of the format of the information to be exchanged and the specification of its meaning.

An Ontology for Services

In DAML+OIL, abstract categories of entities, events, etc. are defined in terms of classes and properties. DAML-S defines a set of classes and properties, specific to the description of services, within DAML+OIL. The class Service is at the top of the DAML-S ontology. Service properties at this level are very general. The upper ontology for services is silent as to what the particular subclasses of Service should be, or even the conceptual basis for structuring this taxonomy, but it is expected that the taxonomy will be structured according to functional and domain differences and market needs. For example, one might imagine a broad subclass, B2C-transaction, which would encompass services for purchasing items from retail Web sites, tracking purchase status, establishing and maintaining accounts with the sites, and so on. The ontology of services provides two essential types of knowledge about a service, characterized by the questions:

– What does the service require of agents, and provide for them? This is provided by the profile, a class that describes the capabilities and parameters of the service. We say that the class Service presents a ServiceProfile.

– How does it work? The answer to this question is given in the model, a class that describes the workflow and possible execution paths of the service. Thus, the class Service is described By a ProcessModel

The ServiceProfile provides information about a service that can be used by an agent to determine if the service meets its rough needs, and if it satisfies constraints such as security, locality, a_ordability, quality-requirements, etc. In contrast, the ServiceModel enables a requester to: (1) perform a more in-depth analysis of whether the service meets its needs; (2) compose service descriptions from multiple services to perform a specific task; (3) coordinate the activities of different requesters; and (4) monitor the execution of the service. Generally speaking, the ServiceProfile provides the information needed for an agent to discover a service, whereas the ServiceModel provides enough information for a requester to make use of a service. In the following sections we discuss the service profile and the service model in greater detail, and introduce the service grounding, which describes how agents can communicate with and thus invoke the service.

Service Profile

A service profile provides a high-level description of a service and its provider [5,6] ; it is used to request or advertise services with discovery services and capability registries. Service profiles consist of three types of information: a description of the service and the service provider; the functional behavior of the service; and several functional attributes tailored for automated service selection.

The profile includes a high-level description about the service and it’s provenance, which typically would be presented to users when browsing a service registry (see Table 1). The class Actor is also defined to describe entities (e.g. humans or organizations) that provide or request Web Services. Two specific classes are derived from the Actor class; the Service-Requester class and Service-Provider class, to represent the requester and provider of the service respectively. Properties of Actor include `physical-Address`, `WebURL`, `name`, `phone`, `email`, and `fax`. Functional attributes specify additional information about the service, such as what guarantees of response time or accuracy it provides, the cost of the service, or the classification of the service in some registry such as the NAICS [9].

Implicitly, service profiles specify the intended purpose of the service, because they specify only those functional behaviors that are publicly provided. A book-selling service may involve two different functionalities: it allows clients to browse its site to find books of interest, and it allows them to buy the books they find. The book-seller has the choice of advertising just the book-buying service or may also advertise browsing functionality. In the latter case the service publicizes the fact that agents may browse without buying a book. In contrast, by advertising only the book-selling functionality, the service discourages browsing by requesting agents that do not intend to buy.

While service providers define advertisements for their services using the service profile, service requesters also use the profile to specify their needs and expectations For instance, a provider might advertise a service that provides quotes for a given ticker symbol, whereas a requester may look for a service that reports current market prices and stock quotes. Services advertise their profiles with Internet wide discovery services, such as Middle Agents [4] and other registries (e.g. UDDI [10]), which then match service requests against the advertised profiles, and identify which services provide the best match. Service requests are constructed as partial service profile descriptions, which can then be matched to the profiles of advertised services stored in the registries using DAML+OIL subsumption relations. Advertisements and requests can differ sharply, in level of detail and in the level of abstraction of the terms used. Matches are generally recognized whenever the service advertised is subsumed by (is a particular case of) the service description requested.

The service representation of DAML-S is much richer than the representation provided by emerging standards such as UDDI or WSDL. UDDI’s description of a service does not include any capability description, limiting itself to the name, a pointer to the provider of the service and a port where to access the service. In addition, UDDI allows

Description Properties and Functional Attributes

`serviceName`	The name of the service
`intendedPurpose`	A high-level description of what constitutes (typical) successful execution of a service.
`textDescription`	A brief, human readable description of the service, summarizing what the service offers or what capabilities are being requested.
`role`	An abstract link to Actors involved in the service execution
`requestedBy`	A sub-property of role referring to the service requester.
`providedBy`	A sub-property of role referring to the service provider.
`geographicRadius`	Geographic scope of the service, either at the global scale (e.g. e-commerce) or at a regional scale (e.g. pizza delivery).
`degreeOfQuality`	Quality qualifications, such as providing the cheapest or fastest possible service.
`serviceParameter`	An expandable list of properties that characterize the execution of a service, such as averageResponseTime or invocationCost.
`communicationThru`	High-level summary of how a service may communicate, e.g. what communication language is used (e.g., KQML, SOAP).
`serviceType`	Broad classification of the service that might be described by an ontology of service types, such as B2B, B2C, etc.
`serviceCategory`	Categories defined within some service category ontology. Such categories may include *Products, Information Services*, etc.
`qualityGuarantees`	Guarantees that the service promises to deliver, e.g. guaranteeing to provide a response within 3 minutes, etc.
`qualityRating`	Industry-based ratings, such as the “Dun and Bradstreet Rating” for business.

services to refer to “TModels” that are used to link a service to technical specifications or to classification schemes. Therefore, it is possible to ask UDDI for all the services that have a WSDL scheme, but not for all the services that provide a requested functionality.

Process Model

Web Services are Web-accessible programs or devices. Their operation is described in terms of a process model, which details both the control structure and data flow structure of the service, i.e., the possible steps (typically initiated by messages sent by the client) required to execute a service. The process model comprises subclasses and properties of the ProcessModel class. The two chief components of the process model are the Process Ontology which describes a service in terms of its inputs, outputs, preconditions, effects, and, where appropriate, its component subprocesses; and the Process Control Ontology which describes each process in terms of its state, including initial activation, execution, and completion.

Service Grounding

The grounding of a service specifies the details of how to access the service –details having mainly to do with protocol and message formats, serialization, transport, and addressing. A grounding can be thought of as a mapping from an abstract to a concrete specification of those service description elements that are required for interacting with the service; for our purposes, the inputs and outputs of atomic processes[2]. Note that in DAML-S, both the ServiceProfile and the ProcessModel are conceived as abstract representations; only the ServiceGrounding deals with the concrete level of specification.

In DAML-S, the abstract content of a message is specified, implicitly, by the input or output properties of an atomic process. Thus, atomic processes, in addition to specifying

---

[2] Atomic processes are one step executable processes

the primitive processes from which larger processes are composed, can also be thought of as the communication primitives of an (abstract) process specification.

Concrete messages, however, are specified explicitly in a grounding. The central function of a DAML-S grounding is to show how the (abstract) inputs and outputs of an atomic process are to be realized concretely as messages, which carry those inputs and outputs in some specific transmittable format. In crafting our DAML-S grounding mechanism, we use Web Services Description Language (WSDL), a particular specification language proposal that is representative of efforts in this area and that has strong industry backing.

WSDL "is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate" [2].

The DAML-S concept of grounding is generally consistent with WSDL's concept of binding. Indeed, by using the extensibility elements already provided by WSDL groundings can be defined for DAML-S atomic processes.

3 DAML-S for Battlefield Networks

In this section we will illustrate ways in which semantic information provided by a language such as DAML-S could be used to help resolve problems in battlefield networks.

Standing up the Network

Rapidly setting up a networked information system and putting it to use immediately are significant problems when forces are deployed to new areas. Unlike designed or constructed networks which associate known entities with addresses and routings there will be no time to design, assign, or even know all the entities which may need to be part of the network. Finding resources on such a new network will prove even harder than the Web because there will have been no opportunity for spiders to index them for a search engine. The alternative, as proposed for the JBI and supported by DAML-S, is to have the resources register themselves as they join the network. This can be done in either of two ways: by setting up central or distributed registries or by using a peer-to-peer protocol such as GNUTELLA to create "virtual registries".

3.1.1 Discovery using Registries

Much of the work on resource discovery is based on centralized registries such as UDDI or the DAML-S Matchmaker. An architecture based on centralized registries assumes that every information resource coming on line advertises its existence and its capabilities/ functionalities with the registry; and that every information requester contacts the registry to discover the most appropriate provider and gather information about it. Centralized registries are effective since they guarantee discovery of providers that have registered. On the other hand, they suffer from the traditional problems of centralized systems, namely they are performance bottlenecks and single points of failure. In addition, they may be more vulnerable to denial of service attacks. Moreover, the possible storage of vast numbers of advertisements on centralized registries hinders the timely update, as changes in the availability and capabilities of providers change. These problems can be partially alleviated through replication of servers, to mitigate against single point of failure and performance bottlenecks.

3.1.2 Peer-to-Peer Discovery

Peer-to-Peer (P2P) computing provides an alternative that does not rely on centralized registries; rather it allows information requesters to discover providers dynamically. Under this view, requesters and providers are nodes in a network of peers. At discovery time an information requester queries its neighbors in the network. If any one of them matches the request, then it replies, otherwise it queries its own neighboring peers and the query propagates through the network. Such architecture does not need a centralized registry since any node will respond to the queries it receives. P2P architectures do not have a single point of failure; rather the high connectivity guarantees that the message reaches the provider. Furthermore, each node contains its own indexing of the existing requesters and providers so there is no danger of a bottleneck effect. Finally, nodes contact each other directly, so there are no delays with the propagation of new information.

The reliability provided by the high connectivity of P2P systems comes with performance costs and lack of guarantees of predicting the path of propagation. Any node in the P2P network has to provide the resources needed to guarantee query propagations and response routing, which in turn means that most of the time the node acts as a relayer of information that may be of no interest to the node itself. This results in inefficiencies and large overhead especially as the nodes become more numerous and connectivity increases. Furthermore, there is no guarantee that a request will spread across the entire network, therefore there is no guarantee to find the providers of critical information even if they are on the network.

Because of their respective advantages and disadvantages P2P systems and centralized registries strike different trade-offs that make them appropriate in different situations. P2P systems are more appropriate in dynamic environments such as ubiquitous computing, while centralized registries are more appropriate in static environments where information is persistent.

3.1.3 DAML-S and Discovery

As an implementation independent language, DAML-S can support either registry based or peer-to-peer discovery. For battlefield networks it is likely that both may be needed. Registry based discovery would allow the rapid registration of massive numbers of information providers and allow efficient matching. When communications channels became degraded or registries lost, damaged parts of the network could fall back to a peer-to-peer discovery protocol and continue operating albeit in a degraded mode.

3.1.4 Dynamic Reconfiguration

A key feature of registration and discovery of providers is that it allows far more flexibility than standard indexing schemes. Temporarily available assets can register and unregister themselves as they enter and leave availability. If a superior information provider registers or a preferred information provider is lost, the matchmaker can still satisfy the request using ontology to bridge differences in terminology. While even as now unregistered providers cannot be found, to the extent that providers comply in registering and unregistering, the network can be reconfigured in close to real time.

Information Overload

The key purpose of associating semantics with battlefield data is to assist in supplying decision makers with information they want and protecting them from the masses of data they do not want. By making data addressable by properties such as spatial location or spot report type, DAML-S would allow a commander to narrowly specify what is of interest to him in his current context.

Unlike systems, for instance, the ability to enter an open request to receive spot reports, the task of reading and eliminating irrelevant reports is eliminated freeing the commander for more productive activities.

Interoperability

Breakdowns in interoperability are a recurring problem in large scale systems composed of heterogeneous parts. The problems can run the gamut from radios on different frequencies to computer files with incompatible data formats. DAML-S is an extension of XML, a metadata markup language, which approaches interoperability by reducing data to the lowest common denominator, plain text ASCII, a data format accepted by most computer systems. DAML-S provides a strong tool for addressing semantic interoperability problems. These may arise when systems use different terms to describe the same or similar things, use different or incompatible units, or express information using different categories.

While DAML-S by itself cannot make information requesters and providers interoperate, it can provide the contextual information needed to make these translations. In particular, the ServiceProfile, ServiceModel, and ServiceGrounding constructs provide the context needed for semantic matching and the concrete protocols needed to access the information.

As the earlier example of incompatibility between GMTI reports and the pre GPS map suggests, interoperability problems often arise when mixing current and older technologies. If the commander could identify landmarks on the conventional map, request GPS coordinates for these, and then locate the GMTI data relative to the landmarks his problem would be solved. DAML-S semantics could assist him in this task by simplifying the process of requesting coordinates and retrieving tools to assist with scaling. If the local environment were sophisticated enough, these might even be performed automatically using the DAML-S supplied information.

Immaturity of Fusion

A final role for semantically marked data lies in facilitating high-level fusion. The massive amounts of data available on the battlefield pose difficulties to computers as well as people. If everything must be correlated with everything the problem becomes too big for even the largest of current systems. Semantic descriptions of what data are will be essential to designing fusion algorithms that can search for meaningful rather than merely serendipitous relations.

If the commander in our earlier example had queried such a futuristic system with his route the system could have used the route's spatial locations in an information request and discovered the nearby vibration readings. Chaining back through these readings the location of the forces from the satellite photograph would have been reached and returned. A relatively simple inference rule acting on these data could then alert the commander and help him avoid the rebel forces.

4 Conclusions

In this paper we have tried to illustrate the roles semantically marked data may play in future C4ISR. Achieving information dominance will require a much tighter integration of data from a broad range of sources than is currently possible. While across the board standardization could vastly simplify the problem, this is not possible either financially or politically. Data descriptions such as those provided by DAML-S appear to a route for moving toward the future with our legacy systems in tow.

References

[1] Berners-Lee, T., J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

[2] Christensen, E., F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/2001/NOTE-wsdl-20010315, 2001.

[3] D. J. Committee. Daml+oil (march 2001) language.http://www.daml.org/2001/03/daml+oil-index.html, 2001.

[4] Paolucci, M., T. Kawmura, T. Payne and K. Sycara. Semantic Matching of WebServices Capabilities. In First Int. Semantic Web Conf., To appear 2002.

[5] Sycara, K., M. Klusch, S. Wido_, and J. Lu. Dynamic service matchmaking among agents in open information environments. Journal ACM SIGMOD Record, 28(1):47–53, 1999.

[6] Sycara, K. and M. Klusch. Brokering and matchmaking for coordination of agent societies: A survey. In Coordination of Internet Agents, 2001.

[7] Sycara, K. and Lewis M. (2002). From data to actionable knowledge and decision. Proceedings of the 5th International Conference on Data Fusion, Annapolis, MD, July 7-11.

[8] T. D. S. Coalition. Daml-s: Web service description for the semantic web.”. In *ISWC2002*, 2002.

[9] U. C. Bureau. North american industry classification system(naics).http://www.census.gov/epcd/www/naics.html, 1997.

[10] UDDI. The UDDI Technical White Paper. http://www.uddi.org/, 2000.

[11] United States Air Force Scientific Advisory Board, "Report on Building the Joint Battlespace Infosphere, Vols 1 and 2," SAB-TR-99-02, December 17, 1999

[12] W3C. Soap version 1.2, w3c working draft 17 december 2001. <http://www.w3.org/TR/2001/WD-soap12-part0-20011217/>, 2001.