

Data Brokers: Building Collections through Automated Negotiation

Fillia Makedon¹, Song Ye¹, Sheng Zhang¹, James Ford¹,
Li Shen¹, and Sarantos Kapidakis²

¹ The Dartmouth Experimental Visualization Laboratory (DEVLAB)
Department of Computer Science
{makedon,yesong,clap,jford,li}@cs.dartmouth.edu
² Department of Archive and Library Sciences
Ionian University, Greece
sarantos@ionio.gr

Abstract. Collecting digital materials is time-consuming and can gain from automation. Since each source – and even each acquisition – may involve a separate negotiation of terms, a collector may prefer to use a broker to represent his interests with owners. This paper describes the Data Broker Framework (DBF), which is designed to automate the process of digital object acquisition. For each acquisition, a negotiation agent is assigned to negotiate on the collector's behalf, choosing from strategies in a *strategy pool* to automatically handle most bargaining cases and decide what to accept and what counteroffers to propose. We introduce NOODLE (Negotiation Ontology Description Language) to formally specify terms in the negotiation domain.

1 Introduction

Digital materials collection has traditionally been a complex and time consuming multi-step process. A collector may have multiple requirements that may change over time, from initially identifying needs to signing on to services, to obtaining approvals for purchases. Collecting objects from different providers can be tedious for collectors because each provider may have his own formats, policies, asset value system, and pricing, and a separate negotiation may be necessary or desirable with each party in order to fully satisfy the collector's requirements. Automating object collection has the potential not only to make the process more efficient, but also to address an important challenge that arises as modern collections are developed – namely, the desire to unify the physical and digital. Automating negotiation is central to the automation of object collection.

Generally, negotiation can be understood as the process toward a final agreement on one or more matters of common interest to different parties. It has been widely accepted that there are two major obstacles in automating negotiation: knowledge representation and strategic reasoning [1, 2], or incorporating necessary negotiation knowledge and intelligence into a computer system that will carry out a negotiation. We introduce NOODLE (Negotiation Ontology Description Language) to address the knowledge representation issue in negotiation and a *strategy pool* to support a

flexible mechanism for choosing and applying negotiation strategies. This work is built on top of an general-purpose negotiation system: *SCENS* [3] (Secure/Semantic Content Exchange System). In *SCENS*, we have been working on building a three mode Web Services-based negotiation system that enables automated negotiation on scientific data sharing. *NOODLE*, which is based on current Semantic Web [4] techniques, is designed to address knowledge representation issue in *SCENS* by creating a standard language for representing and reasoning about negotiation concepts. *NOODLE* provides *SCENS* with a common means to represent different aspects of negotiation.

Here, we incorporate *SCENS* and the strategy pool into a unifying *Data Broker Framework* (DBF) in order to automate the process of collecting widely varying objects. DBF is a distributed framework designed to match needs with available resources. It can be applied to all types of object owners (*e.g.*, libraries, labs, museums, government centers) and object requesters (*e.g.*, conventional libraries, digital libraries, metadata-based digital libraries [5]).

The remainder of this paper is organized as follows. Section 2 reviews the related work in automated negotiation. Section 3 presents the details of the DBF. Section 4 introduces *NOODLE* and the strategy pool technique. Finally, Section 5 offers some concluding remarks and notes on future work.

2 Related Work

Of the two main problems in automated negotiation, knowledge representation is more fundamental than negotiation strategy – after all, all negotiation strategies are based on a correct understanding of the concepts and terms used in a negotiation. There have been several previous efforts to find commonalities across different negotiation protocols [6, 7], and with the development of the Semantic Web, it appears possible to solve or partially solve the problem of knowledge representation using ontologies, which are formal models that describe objects, concepts, and the relations between them [8, 9]. Tamma, *et al.* [8] have theoretically analyzed an ontology for automated negotiation, and Grosz and Poon [9] proposed a rule-based approach to representing business contracts that enables software agents to conduct contract-related activities, including negotiation. However, most existing negotiation ontology work has focused on negotiation activities in e-commerce, and as a result existing techniques cannot be efficiently used for general data sharing negotiation, where many different negotiation conditions might be considered rather than a simple optimization on *e.g.* price.

Negotiation, while a very human process, often paradoxically produces the most useful results if automated, with all terms, sequence of requests, and outcomes recorded and supported by a computer system. Agent technologies are widely used in negotiation systems to replace the activities of human beings and thus automate the negotiation process. Distributed Artificial Intelligence (DAI) and Multi-Agent Systems (MAS) [10] laid important groundwork for agent technology research. Other AI techniques are also frequently used in negotiation systems to help people or agents exhibit rational behavior and obtain the best outcomes. Among these existing approaches, a traditional one is to use game theory to analyze the negotiation process to provide a theoretically sound mathematical solution and winning strategy. However,

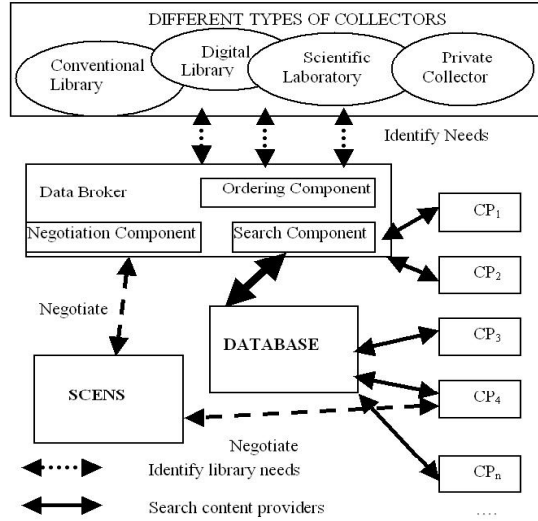


Fig. 1. The Data Broker Framework. The Ordering Component identifies needs of the collector (*dotted arrows*) and feeds these as queries into the Search Component, which retrieves a list of potential content providers (*solid arrows*). The Negotiation Component uses SCENS to negotiate with the content providers for the best offer based on the needs and the optimal strategies for each provider (*dashed arrows*).

this analysis is based on the assumption that the system can get full information about the participants; in the general case, where information and rules are hidden, machine learning technologies are widely used. For example, negotiation can be modeled as a sequential decision-making task using Bayesian updating [11], fuzzy logic [12], defeasible logic [13], or genetics-based machine learning [14]. The latter provides a methodology for constructive models of behavioral processes, in which negotiation rules might be derived and learned from existing rules by means of genetic operations (reproduction, crossover, and activation).

3 The Data Broker Framework

DBF is a distributed framework (as shown in Figure 1): libraries are assumed to use local data brokers that know about local library policies, assets, and similar information. A typical data broker includes the following three major components, which are tightly related to object acquisition: the Ordering Component (OC), the Searching Component (SC), and the Negotiation Component (NC).

3.1 The Ordering Component (OC)

In a conventional library context, the Acquisition Department will order needed content periodically after receiving requests from library users. The *Ordering Component* (OC) similarly identifies a collector's needs by executing several phases automati-

cally (with possible human intervention): (a) the entry of requests by users, and a matching with what is already there, (b) the search for potential providers through a metadata library or database, and (c) the automation of a variety of collection procedures. Essentially, DBF extends the library paradigm by making the acquisition process proactive as well as reactive.

Publishers periodically send out a list of recent publications to libraries, and libraries choose to order some items in the list. Based on its needs, usage history, and the publication lists it receives, a library must decide on acquisition priorities [15, 16]. For example, a book lost or damaged by a user may have to be reordered. If numerous users wish to borrow a specific item from the library but only one copy exists, the library may want to order additional copies.

The above scenarios can be characterized as “reactive” because they react to a need after the need has been expressed. A “proactive” process instead anticipates needs: for example, if the first and second editions of a book exist in the library, the library may wish to order a new third edition. Our system supports both reactive and proactive acquisition processes. OC has an interactive object collection interface for librarians and other collectors to enter object needs. The OC component can request human approval before proceeding into negotiation.

3.2 The Searching Component (SC)

Finding all potential object providers is usually not easy for a collector, especially when some special objects are desired, *e.g.*, images of a film star to be added to a cinematographic collection. For this purpose, our system contains a *Searching Component* (SC), which may contain a database or a digital library such as a metadata-based digital library to facilitate the searching process. This database might contain information about object providers, with listings of available objects and preset trading conditions. SC basically acts as a broker between the object requester and object provider, thus making highly heterogeneous objects interoperable and amenable to an efficient search.

Once a data broker knows what to order, it will need to find appropriate object providers and communicate with them. Here we assume that every object provider has a data broker-like interface. For some specific objects, such as journals, there will be only one or two well-known object providers. However, if a library wants to buy a new book, it may be potentially available everywhere – directly through different book resellers, online bookstores, or publishers, or even from individuals.

3.3 The Negotiation Component (NC)

Different object providers may provide different offers for the same object. Due to budget limits, conventional collectors, such as libraries, hope to find agreeable offers for all needed objects. Negotiation is currently seldom used by libraries in the acquisition process because of its high overhead and uncertain results. Automated negotiation, because of potential for dramatically low cost, can be used for most negotiations, thus making the acquisition process more scalable.

In Figure 1, the broker is conducting negotiation with CP_4 through a negotiation agent. Rather than conducting negotiations directly, the *Negotiation Component* (NC) creates a set of negotiation agents that conduct negotiations autonomously with other

agents [17, 18]. When the broker finds potential providers, NC will generate a negotiation agent for each upcoming negotiation activity. The negotiation agent will communicate with SCENS to obtain the negotiation protocol, which includes the knowledge of how to understand negotiation proposals, conditions, agreements, *etc.* Then it will be assigned a negotiation strategy and will use this strategy to conduct negotiation with other agents through SCENS. The details of representation of negotiations and strategies are discussed in Section 4.

3.4 A Sample Scenario

Assume a Data Broker is responsible for representing a client (a library to be populated) with relevant content providers. It is to acquire objects for the client under a given set of requirements covering purchase price, duration of use, restrictions (or lack thereof) on usage, *etc.* The following summarizes its object acquisition process:

1. It identifies the object needs.
2. It identifies all possible object providers, some available locally and some after consulting centralized servers, such as a MetaDL server.
3. A negotiation strategy is chosen (but may change or be revised later as negotiation proceeds and the system “learns” from past or different types of negotiations).
4. While searching for all sources, it can enter negotiation mode with one of the object providers it has found in order to determine whom to negotiate with later (stepwise negotiation).
5. It can conduct multiple such negotiations simultaneously (Figure 1).
6. The negotiation strategy may change, but will always aim to optimize the criteria of the object requestor.
7. This can be a cyclical process (since negotiation with an earlier party might resume under different conditions) and, in the process, the ranking of object providers can change.
8. The process ends or is suspended at the decision of either party, *e.g.* because he is not prepared to commit or because a certain time has elapsed. The process can resume at a later time, when conditions may have changed (*e.g.*, changes in price or budget). In this case, the data broker should alert the parties of these changes.

4 Structure of the Negotiation Component (NC)

As mentioned above, the Negotiation Component is the most important part of DBF. The key functionalities of NC are correctly understanding the negotiation ontologies and choosing appropriate negotiation strategy. NOODLE, described below, is used to ensure that all negotiation agents have a uniform knowledge of negotiations, including how to conduct them. Negotiation agents are assigned appropriate negotiation strategies from a *strategy pool* based on the current negotiation task. Appropriate strategies are generated based on the past history of negotiations.

4.1 NOODLE

NOODLE (Negotiation Ontology Description Language) is an important part of SCENS. With NOODLE, the negotiation protocols, proposals, conditions, and final

agreement will be described in a negotiation agent-understandable manner, which will allow automated negotiation to be supported by SCENS layers 2 and 3. NOODLE is based on DAML+OIL [19], a standard ontology description language. The goal of NOODLE is to help formalize negotiation activities by predefining some commonly used negotiation concepts and terms. Although these concepts and terms could be defined directly on top of the DAML and OIL ontology description languages, NOODLE is focused on providing a standard specifically for negotiation ontologies. The implementation of NOODLE will be available at <http://scens.cs.dartmouth.edu>, which is still under construction.

Our current definition of NOODLE has three parts: `negotiation.daml`, `proposal.daml`, and `agreement.daml`. In each of these three files, different aspects of negotiation are defined. `Negotiation.daml` defines the skeleton of a negotiation activity, including the number of negotiation parties and the actions that can potentially be used by the requester and owner, such as *Initiate*, *Reject*, *Accept*, ... *etc.* Some actions are used together, with the association defined in `proposal.daml` and/or `agreement.daml`. For example, an *Accept* action will always be followed by an agreement; a *Propose* action likewise is followed by a proposal/offer. Figure 2 shows a part of `negotiation.daml` with the comments removed. `Proposal.daml` defines the format of the messages that are exchanged in any negotiation activity. Basically there are two types of messages, “proposal/offer” and “critique”. A proposal/offer is composed of several conditions, and a critique contains comments on one or more conditions. Currently, NOODLE defines several commonly used negotiation conditions in data sharing, such as usage period, payment, user groups, *etc.* Additional conditions can be added easily. After negotiation parties reach final agreement, they need to have something like contracts to sign. `Agreement.daml` defines the format of the final agreement with semantic meanings. Each negotiation party is allowed to review the final agreement before it is signed by a digital signature; after this, it cannot be refuted by any one of negotiation parties unless all parties agree to revoke the agreement.

```
<daml:Class rdf:ID="Negotiation">
  <rdfs:label>Negotiation</rdfs:label>
</daml:Class>

<daml:ObjectProperty rdf:ID="initiate">
  <daml:inverseOf rdf:resource="Negotiation.daml#initiateBy" />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="initiateBy">
  <daml:inverseOf rdf:resource="Negotiation.daml#initiate" />
</daml:ObjectProperty>
```

Fig. 2. A `Negotiation.daml` fragment, showing the class **Negotiation** and two important properties, **initiate** and **initiateBy**. A negotiation can be *initiated* by exactly one negotiation party, which is the party that *initiates* it, and so the two properties are semantically related. Both are needed in order to ensure that reasoning about the negotiation can be conducted automatically. In addition to the above fragment, the full code includes a “cardinality restriction”, which ensures that there is a one-to-one relationship as described above.

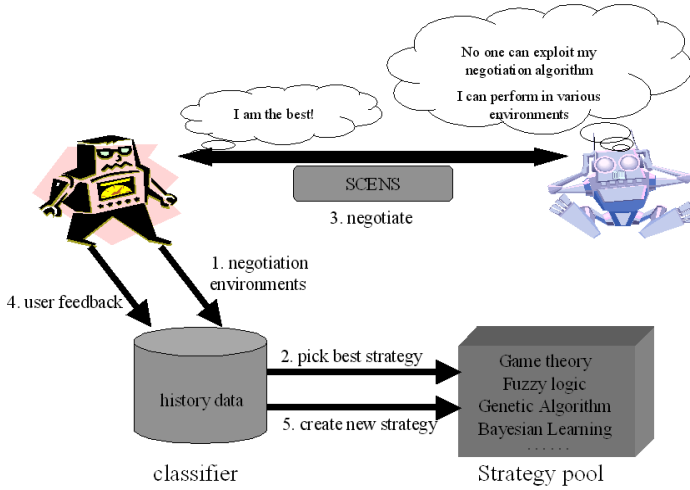


Fig. 3. Strategy Pool: (1) The agent input the negotiation environment parameters into the classifier. (2) The classifier selects a best strategy from the strategy pool. (3) The agent uses this strategy to negotiate with other agents through SCENS. (4) The agent returns the user feedback to the classifier. (5) The classifier generates new rules and creates the new strategy.

4.2 Strategy Pool

There are three important standards for a good negotiation agent. First, it should prevent the other agents easily find its negotiation rules and negotiation strategies. Intuitively, if a collector agent's reservation price for a certain object (generally the highest price the buyer can afford) is determined by a supplier agent after some interaction, the supplier agent can use this information to gain an unfair advantage in later negotiations with this collector. Second, a good agent needs to be flexible, which means it must work well in a variety of negotiation environments. Different environments include different user preferences (*e.g.* user may desire aggressive, neutral, or conservative bidding or bargaining), different user requirements (*e.g.* priority for price *vs.* delivery time), and different profiles of the agents to be negotiated with (agents' reputations). Finally, a negotiation agent needs to be more economical (or no worse) than a human being, taking into account any cost or savings from replacing human negotiators with agents and any required human interventions.

To allow an agent to achieve these three standards, we propose using a *Strategy Pool*. Figure 3 shows that for each negotiation process, the DBF system deploys a new negotiation agent on its behalf. That agent enters the current negotiation environment features into a classifier, which then selects a negotiation strategy or a combination of several strategies from the strategy pool according to past experiences and feedback. The agent then uses this negotiation strategy to negotiate through SCENS. After the negotiation process ends, the agent and its user can provide a negotiation history and feedback on the result to the classifier. Over time, based on the feedback from past negotiation processes, the system can thus make use of machine learning to find the best strategy for each different negotiation environment. Moreover, the classifier may create new negotiation strategies by discovering new negotiation rules or

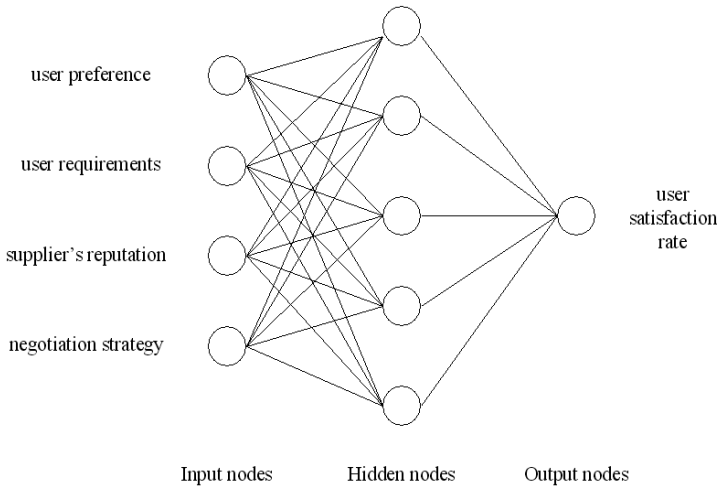


Fig. 4. Using a neural network to choose an appropriate strategy for a given negotiation. The inputs to the neural network are user preferences and requirements (negotiation conditions), the reputation of the current supplier, and current strategies (*left nodes*). The output node (*right*) encodes the expected average user satisfaction rate, which the network attempts to optimize by changing the value of the negotiation strategy input.

combining groups of existing strategies. Each such new strategy can then be added to the strategy pool for later use.

By using the strategy pool framework, we argue that the negotiation agent in the DBF system is made more flexible. This is because the negotiation strategy picked for the agent for a particular negotiation process is generally one that performed well on similar negotiation cases in the past (if such cases are known). Another advantage is that the strategy in each negotiation process is potentially different and is always subject to revision, which should make it more difficult for other agents to deduce or induce the strategy the agent uses; this may reduce any potential vulnerability arising from the discovery of “secret” information (such as reserve prices).

The learning in a DBF system can take one of two forms. In the first, the classifier can use a supervised learning process such as neural network (see Figure 4) to help it benefit from its experiences. To do that, after each negotiation process, the user performs some evaluation – say, assigns a ranking score to show his satisfaction rate on the negotiation result. Thus, a more favorable strategy will be chosen the next time based on the set of environments. In the second learning formulation, the classifier can use the data mining techniques to find the interesting association rules like “in those negotiation process getting the top 10% user satisfaction, 80% of buyer agents bid with a 5% concession from their previous bid when the supplier agents have the same percent concession”. Such a rule may be helpful, but not be in the current strategies. Therefore, we can incorporate this rule into those strategies where appropriate to form new strategies, which will then be loaded into the strategy pool.

5 Concluding Remarks

The Data Broker Framework is currently under development. Certain components have been implemented (SCENS, nAGENTS and the data collection interface) for the area of brain imaging data [20-22]. There are several difficulties involved with the implementation so far: ensuring all providers can understand each other (*i.e.* encode their using the same format and ontology); ensuring each provider updates offers, especially when going through a central server (if a provider uploads a pricelist to server, then changes a price, a client may still bargain based on an old price); and preventing inefficient use of the system (*e.g.*, malicious users who just want to see how low providers will go, but not actually buy anything).

The current version of NOODLE is defined on top of DAML+OIL. As DAML+OIL is to be replaced by OWL in the future, we are planning to eventually convert NOODLE to OWL. Although NOODLE was originally defined to support automated negotiation for scientific data sharing, the current version is more like a general negotiation ontology definition language. We are planning to extend NOODLE to provide better support for negotiation on Digital Rights Management.

References

1. Beam, C. and A. Segev., Automated Negotiations: A Survey of the State of the Art. 1997.
2. Kraus, S., Negotiation in multiagent environments. 2001, Cambridge, MA: The MIT Press.
3. Ye, S., et al. SCENS: A system for the mediated sharing of sensitive data. In Third ACM+IEEE Joint Conference on Digital Libraries (JCDL03). 2003. Houston, TX.
4. SemanticWeb, <http://www.thesemanticweb.org>.
5. Makedon, F., et al. MetaDL: A digital library of metadata for sensitive or complex research data. In ECDL02. 2002. Rome, Italy.
6. Bartolini, C. and C.P.N. Jennings. A generic software framework for automated negotiation. In First International Conference on Autonomous Agent and Multi-Agent Systems. 2002.
7. Lomuscio, A., M. Wooldridge, and N. Jennings, A Classification Scheme for Negotiation in Electronic Commerce. Agent Mediated Electronic Commerce: A European Perspective, ed. D. F. and C. Sierra. 2000, Berlin: Springer-Verlag.
8. Tamma, V., M. Wooldridge, and I. Dickinson. An ontology for automated negotiation. In The AAMAS 2002 workshop on Ontologies in Agent Systems (OAS2002). 2002. Bologna, Italy.
9. Grosz, B. and T. Poon. SweetDeal: Representing Agent Contracts with Exceptions using XML Rules, Ontologies, and Process Descriptions. In WWW 2003. 2003. Budapest, Hungary: ACM Press.
10. Gilad, Z. and J.S. Rosenschein, Mechanisms for Automated Negotiation in State Oriented Domains. Journal of Artificial Intelligence Research, 1996. 5.
11. Zeng, D. and K. Sycara, Bayesian learning in negotiation. International Journal of Human-Computer Studies, 1998. 48: 125-141.
12. Kowalczyk, R. and V. Bui. On fuzzy e-negotiation agents: Autonomous negotiation with incomplete and imprecise information. In DEXA Workshop 2000. 2000.
13. Governatori, G., A.H.M. ter Hofstede, and P. Oaks. Defeasible logic for automated negotiation. In Fifth COLLECTer Conference on Electronic Commerce. 2000. Deakin University, Burwood, Victoria, Australia.

14. Matwin, S., T. Szapiro, and K. Haigh, Genetic algorithms approach to a negotiation support system. *IEEE Transactions on Systems, Man, and Cybernetics*, 1991. **21**.
15. Capron, L., The Long-term performance of horizontal acquisitions. *Strategic Management Journal*, 1999. **20**: 987-1018.
16. Hayward, M.L.A., When do firms learn from their acquisitions experience? Evidence from 1990-1995. *Strategic Management Journal*, 2002(23): 21-39.
17. Kraus, S., Negotiation and cooperation in multi-agent environments. *Artificial Intelligence*, 1997. **94**(1-2): 79-98.
18. Arizona Health Sciences Library, Collection Development Policy for Electronic Resources.
19. DAML+OIL, <http://www.daml.org/2001/03/daml+oil>.
20. Wang, Y., et al. A system framework for the integration and analysis of multi-modal spatio-temporal data streams: A case study in MS lesion analysis. In *EMBS IEEE 29th Annual Northeast Bioengineering Conference*. 2003. Capri, Italy.
21. Makedon, F., et al. Multi-functional data collection interfaces for biomedical research collaboration. In *Human Computer Interaction (HCI)*. 2003. Crete.
22. Steinberg, T., et al. A spatio-temporal multi-modal data management and analysis environment: A case study in MS lesions. In *15th International Conference on Scientific and Statistic Database Management (SSDBM)*. 2003. Cambridge, MA.