Global Trader Cooperation in Open Service Markets

S. Müller, K. Müller–Jones, W. Lamersdorf, T. Tu

Hamburg University, Computer Science Dept.
Vogt–Kölln–Str. 30, D–22527 Hamburg
[smueller|kmueller|lamersd|tu]@dbis1.informatik.uni-hamburg.de

Abstract

Client support for service access in open distributed systems plays an increasingly important role in the context of *Open Distributed Processing* (ODP). Examples for that include ODP's early standardisation efforts in the field of an ODP *trading* function and recent efforts of the *Object Management Group's* (OMG) to standardise a trading facility as one of its CORBA "Common Object Services". In addition to that, *integrating* different local trader functions in order to extend service access support beyond local network boundaries recently became an increasingly important new trading function.

Based upon respective standardisation activities for trader *cooperation* in open service environments, this paper first elaborates on various ways to integrate cooperating local trading facilities into a *global, distributed trading function.* It then reports on specific prototyping experiences made in the international research project on *Interworking Of Traders* (IWT). Within this context, the paper focuses specifically on various aspects of designing and implementing a trader *link management* component that forms the basis for set up, maintenance, and coordination of global trader cooperations.

1 Introduction

Driven by recent improvements in communications technology, the development of open distributed service markets [GGL+95, MML94a] has led to a situation where the dynamics is comparable to that of "real world" markets. It is this dynamic character that makes the task of finding a suitable service rather difficult for potential service users.

Flexible mechanisms have to be provided to support service *mediation* and *selection* in large open distributed service markets and *standardisation* emerges to be one of the key aspects in this field.

A possible way to solve this problem is to first generically characterise service offers by means of standardised *service types* and then to manage the exported service offers by means of *trading functions.* In this context, traders are specific service providers that enable clients to locate (i.e. *import*) suitable service offers by specifying the required service type and additional service properties. First, such offers have to be "advertised" (i.e. *exported*) to the trader by the service providers themselves. Then a potential service user merely has to specify the characteristics (i.e. the type) of service required and has to have access to at least one (local or remote) trader to send the service request to (see fig. 1 for basic principle).

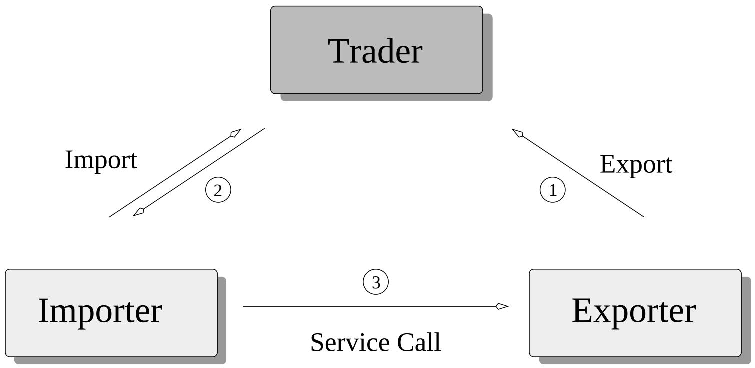

Figure 1: Trader based service mediation

Several projects have dealt with ODP–Trading recently and some respective prototypes have been developed already (e.g. [KW94, BB94, PMGG95, MML94b]). However, most of these prototypes still lack the ability to cooperate with other instances of the same trader and none of them is able to communicate with instances of different trader prototypes. In order to realize global trader cooperation in open distributed systems, additional efforts have to be made to enhance existing trading concepts to enable local trading functions to interact with one another.

One of the first projects dealing with trader cooperation is the international *Interworking Of Traders* project (IWT) [VBB95]. Initiated in 1994, its first stage has just been completed.

The remainder of this paper will introduce some of the major aspects of the IWT project and its achievements so far. Particular emphasis will be placed on the description of a *Link Management* mechanism [ODP95] proposed and implemented for global trader cooperation in a joint project of Hamburg University and the Australian *Distributed Systems Technology Centre* (DSTC) in the context of Hamburg University's *Service Trading and Coordination Environment* (TRADE) [MML95b, MML95a, LMM95].

2 Trading as a Service Mediation Mechanism

2.1 Domain Restricted Service Mediation

Due to organisational, historical or technological circumstances, traders traditionally cover one single domain (technological, administrative, security etc.), called a *trading domain*, only and efficient mechanisms for service mediation have been developed for use *within* such a domain. Domain–oriented service mediation, however, has several drawbacks concerning, e.g., service *autonomy* and *usability*. Most drastically, in such scenarios client users can only participate in those parts of electronic markets that are managed by the trader of their local domain; they can only import service offers explicitly exported to the one trader they ask. Servers on the other hand are forced to export their offer to the one trader in the domain they belong to and therefore will only be propagated in one certain part of the electronic market. Their offers remain unknown for potential service clients in other parts of a global network environment. Even importing from — or exporting to — multiple traders seems to be an unsatisfying solution for these problems as this reintroduces the problem of service location at a different level since both provider and client then would have to know explicitly how to contact all the possible traders in consideration.

Another problem lies in the *scalability* of local trading functions: for larger trading domains with large numbers of service providers and clients performance of a centralised trading architecture can become crucial. This is specifically important within domains where the number of offers and requests changes frequently; also mere optimisation in the area of hardware will not lead to satisfactory results either.

2.2 Cooperation as a Key Mechanism to Enhance Local Traders

To overcome the problems of domain–oriented trading as described in the previous section, mechanisms have to be introduced to weaken the domain boundaries for both traders and their clients while directing special attention to keeping maximum autonomy of all participants.

Basis for overcoming the domain orientation is a corresponding system support. Existing platforms — perfectly fitting into their context — have to be combined in a manner that enables transparent interaction of clients and servers developed on them. One possible solution for this supporting framework lies in the introduction of *interceptors* [GMJL96] that instantiate so called *proxies* to offer services on behalf of service providers in foreign domains. It is through proxies, that clients are then able to access service providers in spite of their potential location in different (technological) domains.

In addition to dealing with technical heterogeneity, a domain boundary crossing trading mechanism is necessary to realize large open service markets. Basic requirement for this is an opening of existing trading domains and a mutual exchange of information on services and service offers in these domains.

3 Types of Trader Cooperations

In general, there are several ways to overcome network boundaries and heterogeneity problems for cooperating trading functions. Therefore, this section introduces alternate types of domain boundary crossing for global trader cooperation: Beginning with the simplest form of sharing common service offer repositories it leads to more complex forms with maximum autonomy for participating traders by loosening diverse imposed restrictions.

3.1 Indirect Interaction

Indirect interaction represents the most simple class of trader cooperation. Here, participating traders use common service offer repositories and can access all offers all other traders have deposited there. Indirect interaction requires the least effort for cooperating traders. On the other hand, however, it has several drawbacks:

- All service types have to be well known to all participants.
- Indirect interaction is only practical in distributed systems without administrative, topological or technological boundaries.
- It is only applicable in cases where no security restrictions forbid storing offers in common repositories.

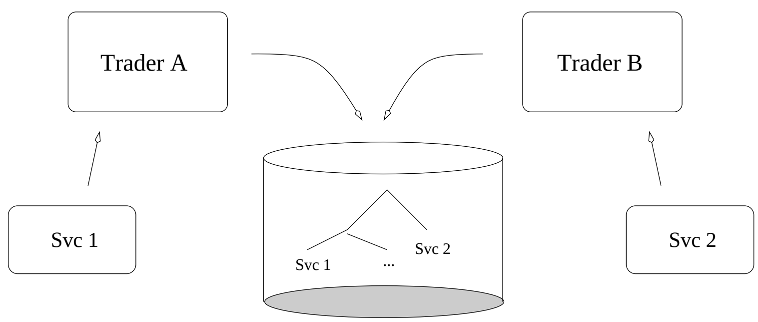

Figure 2: Indirect Interaction

Using standardised, global name services (e.g. X.500), the second and third drawback can be handled to a certain extent and some work exists already that exploits the principle of *indirect interaction* to realize trader cooperation (see, e.g., [WB95]).

3.2 Simple Trader Federation

With *simple trader federation*, obligatory agreements must exist between cooperating traders concerning, at least, call interfaces, call semantics and data structures to be used. As with indirect interaction, service types are standardised and have to be well known to all participants in advance. Service requests and their results can be freely exchanged between cooperating partners in this reader cooperation alternative.

Additionally, traders can interact in this scenario using a well known *cooperation protocol* to obtain information on one another which is stored in so called *links*. A link consists of information necessary to bind to the referenced trader and to forward service requests and also keeps information qualifying the link itself, for example the referenced trader's average answering time.

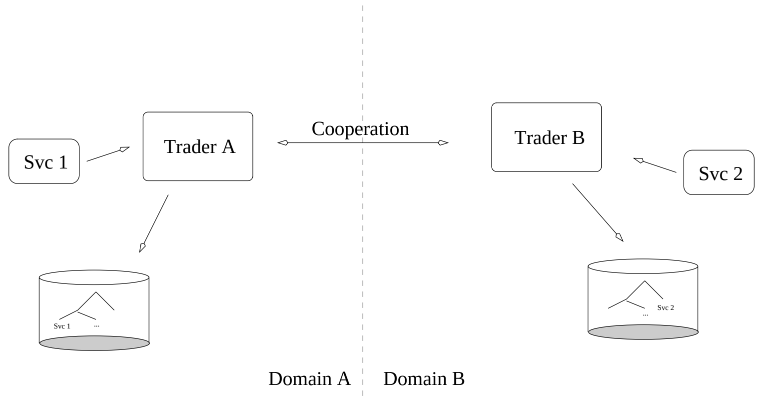

Figure 3: Simple Trader Federation

Using *simple trader federation* the autonomy of all participating traders can be better preserved in comparison to the indirect interaction approach. Clients can use the trader that suits them best while still being able to access foreign trading domains. On the other hand, obligatory agreements concerning service types are still necessary between cooperating partners. To overcome this drawback, *type managers* have to be introduced that serve to compare and transform type information.

3.3 Type Manager Based Federation

Introducing type managers and a corresponding protocol for them to interact, leads to a *type manager based federation*. With this kind of trader federation, explicit type manager functions are required and used to compare service types and to transform service type representations, if necessary.

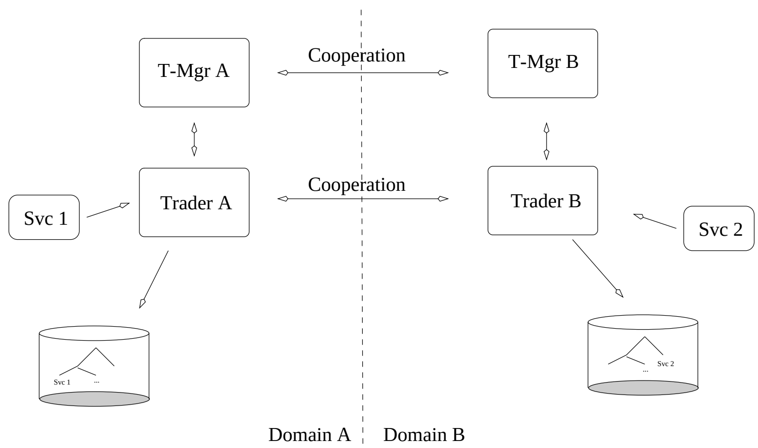

Figure 4: Type Manager Based Cooperation

This model of trader cooperation leads to an increased level of autonomy for single service and trading functions, since the necessary comparison of service types is done by type managers and service types no longer have to be agreed upon.

3.4 Free federation

Finally, giving up the requirement of a homogeneous middleware platform for all participants in cooperative open service markets leads to the most extensive form of trader cooperation, called *free federation*. Free federation relies on a mechanism of *interception* that enables services to interact with one another over heterogeneous platforms. This can be achieved by instantiating *proxies* to offer the service interface of services on one platform on another platform and is supervised by an *interceptor* [GMJL96]. In this way, both client and server have the illusion of using the same platform while all calls are transparently routed through proxies to any other platform anywhere whenever necessary.

As, in an open service market scenario, also traders are just specific servers following the client/server paradigm themselves, the use of interception mechanisms enables them to interact across domain boundaries and thereby leaves them a free choice of their underlying platform as well as of their potential cooperation partners.

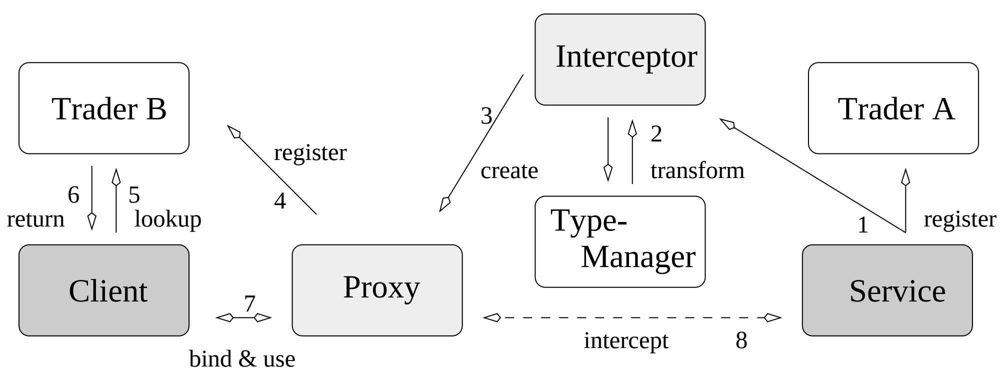

Figure 5: Free Federation using Interception

The main problem with the mechanism of interception and its concept of maximum transparency when used in the field of trader cooperation lies in the introduction of foreign binding information into a domain. In this case, service offers returned to clients may contain bindings that are only valid in the initial domain and not usable for the recipient. Where proxies normally just transform calls, route data in an uninterpreted way and stay “invisible” for traders and their clients, they now have to interact directly with the trader in order to enable an interpretation of the data routed and thus a transformation of bindings.

4 Establishing Trader Cooperations

Some proposals already exist for supporting global trader cooperations. Early works favoured the use of *federation contracts* [BR92, LM95]. These consist of an *import contract* and an *export contract* where the former is managed by the importer and consists of the service types available in the remote trader and mappings between local and remote or canonic service types. According to these proposals, the export contract is held by the exporter and contains path names of exported subsets of its offer space and service types offered. Due to their complexity, however, federation contracts were not considered suitable for ODP standardisation and are not used within the *Interworking of Traders* project, either.

A different approach, as e.g. described in [VBB95], has been adopted for ODP standardisation which requires less administrative overhead. With this kind of trader federation called trader *interworking*, federating traders pass necessary information on one another via an administration interface and store it in so called *links*. Using this information, traders can then send requests to cooperating partners as other clients would. Links contain the partner's binding information, as well as information on the type domain the referenced trader belongs to. This information is then used during requests to decide whether type transformation has to take place. Additionally, links may contain qualifying attributes as, e.g., hit-/miss ratio or average response time to enable a trader to select partners with maximum efficiency based on the statistical information it has stored. Such qualifying attributes are actualised after every call for utmost accuracy.

According to this approach, link management can basically be performed in three distinct phases:

1. Strategic Phase:

   During the strategic phase, links are created between all cooperating traders. The creation takes place via standardised interfaces of the traders, the so called *link management interfaces*. The links instantiated are unidirectional references that are stored in private *link spaces* of the originating traders. Bidirectional references can be "simulated" by creating additional references with opposite direction.

2. Tactical Phase:

   During the tactical phase, traders select the trader references from their link space they find most promising for the request to be made. This selection can be influenced via *policies* by both the trader and the requesting client. Policies can, for example, be used to decide whether the request shall be routed across domain boundaries, or whether the request may be handed to cooperating traders synchronously or asynchronously.

3. Operational Phase:

   During the final, operational phase the trader sends the request to selected cooperating partners via their import interface. Eventual results from remote requests are then merged with local results and handed back to the originating client, while the origin of the results stays transparent to him. At last, the qualifying attributes of the links used are actualised and their quality is reestablished.

5 Prototype Implementation

The following section describes a prototype link management component that has been developed as part of the "Interworking of Traders" project, jointly by the DSTC and Hamburg University.

The main goal of this project stage was to enable and demonstrate a trader interworking scenario between traders developed by each project partner independently (see [MML95b] and [BB94]). As part of this effort, the ODP standard draft was also tested for completeness and usability and first experiences could be gained on using *policies* as a formal mechanism to describe dynamic trader behaviour.

5.1 The Project Environment

This section introduces the prototype system environment that served as a testbed for the development of the link management prototypes.

The middleware platform used within the IWT project was OSF's *Distributed Computing Environment* (DCE) [Fou92]. In both Brisbane, Australia, and Hamburg, Germany, separate trading domains were established by respective DCE cells and connected via DCE *Global Directory Service* (GDS), a DCE Directory Service component. In each of the two trading scenarios, each separate domain contains at least one local trader component. In case of Hamburg University's TRADEr prototype, this trading component basically consists of the following parts: an *access control* module, a *service selection management* module, a *link management* module (as described in this paper), and a *service mediation* module. The TRADEr additionally interacts with a *type manager* and an external *service offer repository*. It offers various call interfaces which belong to either service mediation or general service offer management facilities. The link management module, for example, is part of the latter group.

Both DCE cells offer local services that use their local trader facilities. A specific GUI–based trader administration tool, the *Link Configuration Manager*, enables creation of cooperation links between trading domains. The next section gives a more detailed overview of some aspects of the TRADEr "link Configuration manager" module.

5.2 Link Configuration Manager

The *Link Configuration Manager* (LCM) (see fig. 7) can be regarded as a central controlling unit of the trader's link management facility. Via a graphical administration tool it offers access to all functions of the link management

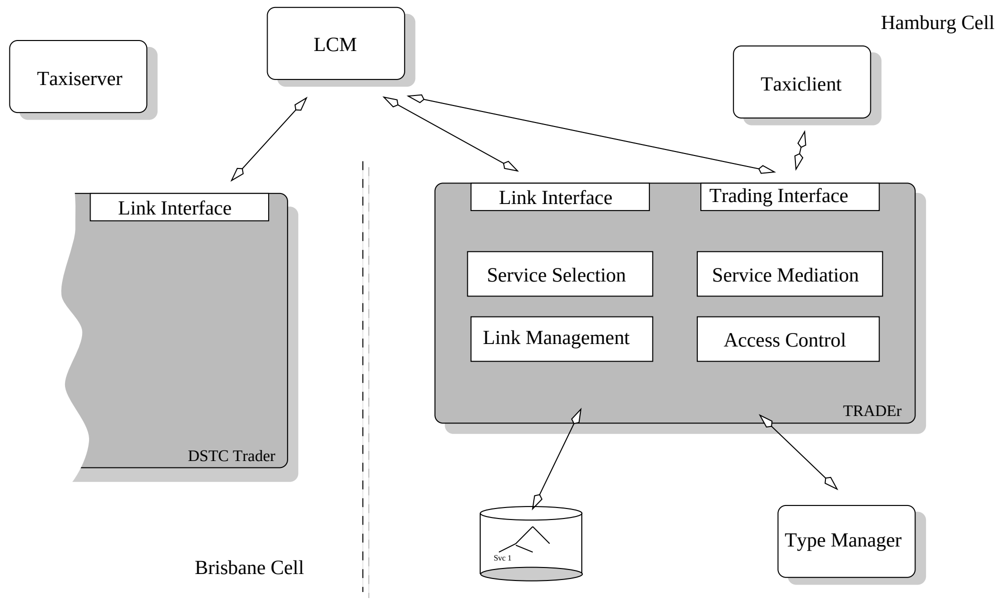

Figure 6: The Project Environment

interface. Using this tool, trading link management administrator can browse all trader service offers known to a specific trader and integrate them into global cooperation links. Additionally, the LCM can be used to supervise links held by a specific trader. By observing link attributes one can, for example, identify link problems already at an early stage (for example if response times are exceptionally high). The procedure to establish connections between traders by means of the LCM is as follows: First, the trading service offered by the foreign trader, which is to be registered as a link with the local one, has to be inquired by making a service offer request to the former one. This step is enabled in the *service offer mode* of the LCM. If successful, the next step is binding to the the Link Management Interface of the local trader and registering the binding information contained in the foreign trading service offer as part of a link. This is enabled in the *link mode* of the LCM. The Link Management Interface also allows a later control of the stored links and their attributes by means of the Link Configuration Manager.

5.3 Trader extensions

Besides the implementation of the additional Link Management Interface, the realization of interoperable trader cooperations in heterogeneous network environments also requires some modification of the Trading Service Interface of the participating traders, especially with respect to the import operation. Additional features the import operation needs to implement are the following:

- Instead of processing requests in an anonymous way, a unique identifier must be kept during the whole life cycle of a request, also when this request is forwarded to a partner trader. These identifiers are stored by

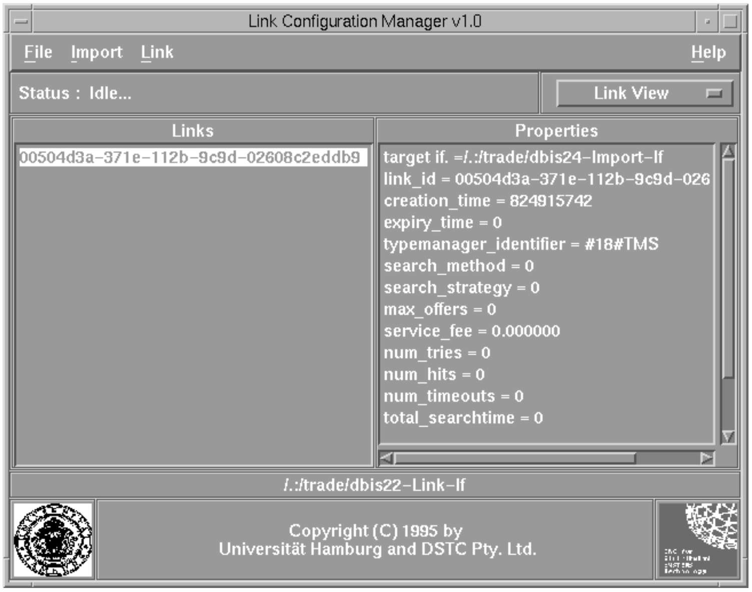

Figure 7: The Link Configuration Manager

the traders and used to recognise requests that have already been handled. Thus, cyclical requests can be avoided.

- Requests may be forwarded to other partner traders. In this case, the forwarding trader behaves to his cooperation partner in the same way as a simple client.

- Policies, which are passed together with the request, can be used to influence the processing of the request, e.g. concerning the involvement of partner traders or the specification of optimisation criteria.

- The results of subrequests are merged and returned to the immediate client of trader. In this way, the origin of a service offer is transparent for the initiator of the request.

Other implementation aspects as, e.g., performance and memory usage have also been considered in the IWT project, partly by using the DCE thread library. By means of this library *asynchronous remote procedure calls*, which are not directly supported by DCE, could be realized in the extended TRADEr prototype scenario.

5.4 A sample application scenario

In order to gain first experience with the implemented prototype trader cooperation mechanisms described above, an example taxi booking service was realized. Using this simple application, it was also possible to demonstrate how the processing of service offer requests can be dynamically controlled at run-time. For example, requests can be specified to be processed locally or

Figure 8: Policy selection using the taxi booking client

also across trading domain boundaries. Figure 8 shows a sample dialog box by means of which users can automatically select the most appropriate taxi booking service available anywhere by using (location transparently!) either local or remote trader functions interconnected by respective LCM managed external links.

6 Summary and Outlook

This article gave an overview of concepts, alternatives, and a prototype implementation of advanced trading functions which aim at realizing global, i.e. interdomain cooperation of local traders. It first addressed problems of centralised service mediation approaches and then introduced as an attractive alternative that of global trader cooperation. The most important core component of such a global trading cooperation mechanism is the *link management* facility. The design and implementation of a prototype for such a component was finally presented in the main section of the paper as developed as part of the "Interworking of Traders" project, a joint project between the Australian "DSTC" and the German Hamburg University recently.

In general, link management functions enables local traders to extend their service offer space beyond local domain boundaries using protocols to contact remote traders whenever service requests can not be granted locally. All information necessary for that is stored in so called *links* within each trader and covers binding information as well as statistical data to qualify a link and enable an optimised selection of cooperating partners. Example test scenarios using the mentioned prototypes demonstrated the usability of both the proposed trader cooperation concepts and their respective realizations as reported in this paper. In these test scenarios, trader functions developed independently in Brisbane and Hamburg could be connected in an application scenario where an example taxi booking services could be mediated across domain boundaries. This way also first experiences could be made with a trader cooperation mechanism as initially specified (but not tested yet!) in the respective ODP standard draft extension.

Among several yet unsolved problems of a trading link management mechanism is, for example, the (dynamic!) control of requests sent to other traders. Yet, ways have to be found to synchronise subrequests and, for example, stop complete request chains where subrequests exceed resource limits or where the requested number of service offers has been reached. In this context, partial results in *mobile agents* research [MLL96] seem promising, but autonomous request coordination protocols will have to be evaluated as well in the future. Additionally, trader links are still only unidirectional references. It is also yet to be examined which consequences (concerning, e.g., the autonomy of cooperating traders) would arise from changing to explicitly negotiated bidirectional trader links instead.

Acknowledgements

The authors wish to thank their Australian partners in the IWT projects David Barbagallo, Melfyn Lloyd, Kerry Raymond and Andreas Vogel etc. at the *Distributed Systems Technology Centre* (DSTC) in Brisbane for the fruitful and stimulating cooperation which lead to many of the results reported here.

The authors would also like to acknowledge technical support from subsequent joint work in related areas with the GMD *Research Institute for Open Communication Systems* (FOKUS) in Berlin.

References

[BB94] A. Beitz and M. Bearman. An ODP Trading Service for DCE. In *Proceedings of the First International Workshop on Services in Distributed and Networked Environments (SDNE)*, pages 42–49, Prague, Czech Republic, June 1994. IEEE Computer Society Press.

[BR92] M. Bearman and K. Raymond. Federating traders: An ODP adventure. In J. de Meer, V. Heymer, and R. Roth, editors, *Open Distributed Processing, Proceedings of the IFIP TC6/WG6.4 International Workshop on Open Distributed Processing, Berlin, 1991*, pages 125–141. Elsevier Science Publishers B.V. (North–Holland), 1992.

[Fou92] Open Software Foundation. *Introduction to OSF DCE*. Prentice–Hall, Englewood Cliffs, New Jersey, 1992.

[GGL+95] K. Geihs, H. Gründer, W. Lamersdorf, M. Merz, K. Müller, and A. Puder. Systemunterstützung für offene verteilte Dienstemärkte. In K. Franke, U. Hübner, and W. Kalfa, editors, *Kommunikation in Verteilten Systemen: Neue Länder – Neue Netze – Neue Dienste*, Informatik–Aktuell, pages 445–459. Springer–Verlag, February 1995.

[GMJL96] F. Griffel, K. Müller-Jones, and W. Lamersdorf. Ein Interoperabilitätskonzept zur komponentenbasierten Software–Entwicklung auf heterogenen Middleware–Plattformen. To be published in Proceedings of GI'96, Klagenfurt, Austria, February 1996.

[KW94] E. Kovacs and S. Wirag. Trading and distributed application management: An integrated approach. In *Proceedings of the 5th IFIP/IEEE International Workshop on Distributed Systems: Operation and Management*, October 1994.

[LM95] L. Lima and E. Madeira. A model for federated trader. In *Proceedings of the Third International Conference on Open Distributed Processing (ICODP '95)*, Brisbane, Australia, February 1995.

[LMM95] W. Lamersdorf, M. Merz, and K. Müller–Jones. Middleware support for open distributed applications. In *Proceedings of the First International Workshop on High Speed Networks and Open Distributed Platforms*, St. Petersburg, Russia, June 1995.

[MLL96] M. Merz, B. Liberman, and W. Lamersdorf. Interorganizational workflow management with mobile agents in cosm. In *Proceedings of the First International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agent Technology*, 1996.

[MML94a] M. Merz, K. Müller, and W. Lamersdorf. Service trading and mediation in distributed computing environments. In *Proceedings of the 14th International Conference on Distributed Computing Systems (ICDCS '94)*, pages 450–457. IEEE Computer Society Press, 1994.

[MML94b] K. Müller, M. Merz, and W. Lamersdorf. Der TRADE–Trader: Ein Basisdienst offener verteilter Systeme. In C. Popien and B. Meyer, editors, *Neue Konzepte für die Offene Verteilte Verarbeitung*, pages 35–44. Verlag der Augustinus Buchhandlung, Aachen, September 1994.

[MML95a] K. Müller–Jones, M. Merz, and W. Lamersdorf. Kooperationsanwendungen: Integrierte Vorgangskontrolle und Dienstvermittlung in offenen verteilten Systemen. In F. Huber-Wäschle, H. Schauer, and P. Widmayer, editors, *GISI 95 – Herausforderungen eines globalen Informationsverbundes für die Informatik, Zürich, 1995*, pages 518–525. Springer–Verlag, 1995.

[MML95b] K. Müller–Jones, M. Merz, and W. Lamersdorf. The TRADEr: Integrating trading into DCE. In K. Raymond and L. Armstrong, editors, *Open Distributed Processing: Experiences with distributed environments, Proceedings of the third IFIP TC 6/WG 6.1 international conference on open distributed processing, 1995*, pages 476–487. Chapman & Hall, 1995.

[ODP95] Draft International Standard 13235 – ODP Trading Function. International Organisation for Standardization, International Electrotechnical Commission, May 1995.

[PMGG95] A. Puder, S. Markwitz, G. Gudermann, and K. Geihs. AI–based Trading in Open Distributed Processing. In K. Raymond and L. Armstrong, editors, *Open Distributed Processing: Experiences with distributed environments, Proceedings of the third IFIP TC 6/WG 6.1 international conference on open distributed processing, 1995*, pages 157–169. Chapman & Hall, 1995.

[VBB95] A. Vogel, M. Bearman, and A. Beitz. Enabling interworking of traders. In *Proceedings of the Third International Conference on Open Distributed Processing (ICODP '95)*, Brisbane, Australia, February 1995.

[WB95] A. Waugh and M. Bearman. Designing an ODP Trader Implementation using X.500. In *Proceedings of the Third International Conference on Open Distributed Processing (ICODP '95), Brisbane, Australia*, February 1995.