References

[Bear94] M. Bearman. *Tutorial on Trading Function*. Part of ISO/IEC 13235: 1994 or ITU-TS Rec X.9tr - WG7 Committee Draft ODP Trading Function Standard, September 1994.

[BeRa91] M. Bearman and K. Raymond. *Federating Traders: An ODP Adventure*. International IFIP Workshop on Open Distributed Processing, October 1991.

[Burg95] C. Burger. *Cooperation policies for traders*. In K. Raymond and L. Armstrong, editors, Open Distributed Processing - Experiences with distributed environments. IFIP, Chapman & Hall, February 1995, pp. 208-218.

[dPLJMa95] L. A. de Paula Lima Jr. and E. R. M. Madeira. *A Model for a Federated Trader*. In K. Raymond and L. Armstrong, editors, International Conference on Open Distributed Systems. IFIP, Chapman and Hall, February 1995, pp. 173-184.

[Ecke95] O. Eckert. *Implementing the supermarket policy for cooperations among traders and integration into MELODY* (in german). Master's thesis, University of Stuttgart, November 1995.

[Fehr96] T. Fehre. *Concepts for user controlled cooperations among traders and integration into MELODY* (in german). Master's thesis, University of Stuttgart, February 1996.

[ISO95] ISO. *Draft ODP Trading Function*. Technical report, OSP/IEC JTC1/SC21, June 1995.

[KoBu95] E. Kovacs and C. Burger. *Project description for MELODY - Management Environment for Large Open Distributed sYstems* (in german). Technical Report 8/95, University of Stuttgart, 1995.

[KoWi94] E. Kovacs and S. Wirag. *Trading and Distributed Application Management: An Integrated Approach*. In Proceedings of the Fifth IFIP/IEEE International Workshop on Distributed Systems: Operations & Management. IFIP/IEEE, October 1994.

[Kutv95] L. Kutvonen. *Achieving Interoperability through ODP Trading Function*. In 2nd International Symposium on Autonomous Decentralized Systems (ISAD'95). IEEE Computer Society Press, April 1995.

[MuMe96] K. Müller-Jones, M. Merz, W. Lamersdorf. Agents, services and electronic markets: how do they integrate? In Alexander Schill, editor, IFIP / IEEE International Conference on Distributed Platforms. IFIP / IEEE, February 1996, pp. 287-300.

[PuBu96] A. Puder and C. Burger. *New Concepts for Qualitative Trader Cooperation*. In Alexander Schill, editor, IFIP / IEEE International Conference on Distributed Platforms. IFIP / IEEE, February 1996, pp. 300-313.

[VBB95] A. Vogel, M. Bearman, and A. Beitz. *Enabling Interworking of Traders*. In K. Raymond and L. Armstrong, editors, Open Distributed Processing - Experiences with distributed environments. IFIP, Chapman and Hall, February 1995, pp. 185–196.

[Wann95] M. Wannenmacher. *Concepts for a universal trader and implementation in MELODY (in german)*. Master's thesis, University of Stuttgart, december 1995.

6. Conclusion

We have described concepts to control cooperations among traders either by providing support for system administrators or by applying a basic set of policies to enable automatic decision making. These concepts were prototypically implemented ([Ecke95], [Fehr96]) and integrated into the MELODY trader ([KoBu95]). We found out, that mere control by human administrators is not sufficient but has to be supplemented by some kind of automatic control. Nevertheless, in this approach a priori agreements to cope with availability problems and estimation of trustworthiness are left to humans. Furthermore, concepts as described above restrict to fixed values of thresholds. We will continue to study adaptation of these values to dynamic behavior.

Up to now, traders have been passively waiting for offers and requests. Another scenario would let traders explore electronic markets and actively look for service providers and users by means of public relations (first steps towards this goal have been done in [MuMeLa96] and [Wann95]). This can be performed e.g. by adapting the technology of agents moving around in the electronic world. But in this case, traders are forced even more to cooperate with other traders and automatically control their behavior, because they will be without permanent contact to their administrators.

In electronic markets, not only traders but arbitrary service providers will have to cooperate also. It has to be examined, how concepts as described above can be extended to cooperations among autonomous service providers.

[Ecke95]. The policy management cares for replacing older versions and having realized current policies.

In this work, we restrict on cooperation relevant policies. To this end, policies define conditions to enable cooperation decisions by specifying

- upper threshold for trader's load and minimum duration of states,
- lower threshold for prices,
- maximum number of cooperations,
- minimum number of service requests/offers to choose negotiation,
- number of direct partners,
- restrictions concerning partners,
- chaining flag (direct or indirect cooperation),
- rules for removal of cooperations.

5.2.2 Decision about cooperation

The algorithm for automatic control decides about initiation and removal of cooperations as well as about necessity of negotiations. The following rules are applied to initiate cooperations as long as maximum number of cooperations has not been reached:

- If a request or offer can not be treated locally and partners with acceptable prices and suited types exist, a simple cooperation is initiated with these partners up to the allowed maximum and price limit of the client.
- If load has been above the upper threshold for the specified duration, a cooperation is initiated without any further consideration.

Negotiations are required, if

- heterogeneous type managers are involved,
- according to the chaining flag, indirect cooperation is not allowed, because otherwise indirect cooperation cannot be prevented,
- more than the minimum number of requests/offers are causing cooperation need concerning the same partner.

If a complex cooperation request is received, it is accepted only if it is not below the lower threshold for prices.

If the maximum number of cooperations is reached and local load has been above its upper threshold for the specified period, cooperations are finished according to removal rules. These rules define to start either with removing complex cooperations with many service types or vice versa.

5. Automatic control

The example of section 4.5 demonstrates, that it does not always make sense to let human administrators decide about cooperations. Sometimes it is even impracticable to wait for the administrator's decision. Therefore, automatization of cooperation control should be considered. To this end, policies must be defined to specify the cooperation behavior of traders in a more or less complete way. Before introducing our new approach, we first describe existing solutions.

5.1 Related research

[ISO95] and [VBB95] apply search policies to limit the use of resources. Thus, cooperations are not allowed, if time to provide a certain search request has expired. This control mechanism can be extended to include economic and security aspects. But decisions only concern the initiation of a cooperation and determine number and characteristics of chosen cooperation partners in an indirect way.

Methods to directly select cooperation partners for traders, are proposed in [ISO95] and [LeBe95]. Both are based on the notion of a 'link' between traders. Links are unidirectional and inform one trader about the other. Whereas [ISO95] allow arbitrary properties of such links, [LeBe95] describe a more sophisticated approach. Its basic concept consists in deriving affinities for links. To this end, personal interest and cooperation offer are compared and can be combined with recent experiences about real cooperation with this partner. These affinities indicate, how well partners are suited. The inclusion of economical and security aspects into affinities will be straightforward.

It can be concluded, that current approaches to automatic control restrict on deciding about initiating a cooperation and on choosing cooperation partners, i.e. the first and second one of the questions as posed in the introduction are partially answered. The first step towards a more comprehensive solution was introduced in [Burg95] and implemented in [Ecke95]. One of its basic ideas was derived from the behavior of supermarkets, where a cooperation is established only if the number of requests concerning a certain unknown product exceeds the limit as defined by the policy. Furthermore, thresholds for load were used to derive decisions about initiation of direct cooperation, choice of partners and the need for negotiation from the performance viewpoint. But indirect cooperation, modification and removal of cooperation, economical aspects and policies of clients have not been taken into account.

5.2 Architecture

Automatic control consists of policy management and components to decide about all characteristics of a certain cooperation.

5.2.1 Policy management

In current trader research, a number of different policies is treated. Policies are defined by administrators, by service users and by service providers. They are defined in a certain language, e.g. the minimal rule language of [VBB95] or the specific language that has been developed in

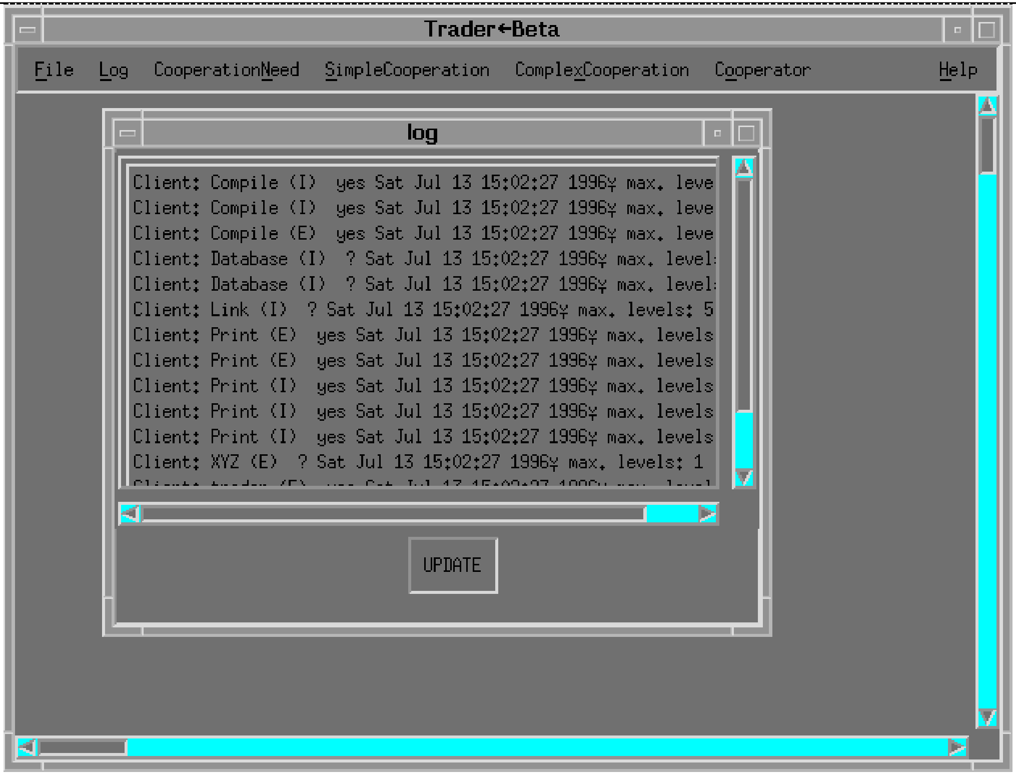
4.5 Change to functionality of isolated trader

If cooperations are controlled by human administrators, it takes some time until cooperation relevant information is examined and decisions are driven. Whereas execution times of traders are below one second (cf. table 1), reaction times of humans exceed these values by several orders of magnitude. As a consequence, incoming calls that have been sent by remote procedure call (RPC) will lead to time out. To cope with this problem, an identifier is returned. Answers arriving after completion of RPC, are temporarily stored until the client has fetched them via identifier.

Mediation time	Managing potential partners	Managing cooperations
0,55 seconds	0,3 seconds	0,09 seconds

Table 1: Execution times of MELODY trader

All cooperation relevant information has to be presented in a human readable form. To facilitate overview, statistics are provided also (e.g., the number of failures that happened during a certain period of time). Figure 3 shows the user interface of Melody traders. In this example, four different kinds of logs are presented. The button *Log* cares for presenting all information, whereas *SimpleCooperation* and *ComplexCooperation* restrict on calls from other traders concerning cooperation without or with negotiation. Choosing *CooperationNeed* results in a list of events that can cause cooperation like unsuccessful requests (marked with “?” like the ones for service type “database” in figure 3) or offers of other traders. *File* is applied to store one of these logs in a certain file. Each list can be sorted according to call type, service type, time and sender.



```
Client: Compile (I)  yes Sat Jul 13 15:02:27 1996¥ max. leve
Client: Compile (I)  yes Sat Jul 13 15:02:27 1996¥ max. leve
Client: Compile (E)  yes Sat Jul 13 15:02:27 1996¥ max. leve
Client: Database (I)  ? Sat Jul 13 15:02:27 1996¥ max. level:
Client: Database (I)  ? Sat Jul 13 15:02:27 1996¥ max. level:
Client: Link (I)  ? Sat Jul 13 15:02:27 1996¥ max. levels: 5
Client: Print (E)  yes Sat Jul 13 15:02:27 1996¥ max. levels
Client: Print (E)  yes Sat Jul 13 15:02:27 1996¥ max. levels
Client: Print (I)  yes Sat Jul 13 15:02:27 1996¥ max. levels
Client: Print (I)  yes Sat Jul 13 15:02:27 1996¥ max. levels
Client: Print (I)  yes Sat Jul 13 15:02:27 1996¥ max. levels
Client: XYZ (E)  ? Sat Jul 13 15:02:27 1996¥ max. levels: 1
```

Figure 3: User interface for system administrators

The button *Cooperation* is responsible for

- sending a characterization of the local trader to other traders (cf. section 3.2),
- manipulating cooperations (initiate, modify, finish).

To initiate cooperations, selections have to be performed concerning complexity of the cooperation (with/without negotiation), directness, cooperation needs and partners. From these declarations, an appropriate request is built and transferred by the cooperation handler to all chosen partners. Requests/offers for simple cooperations are treated like normal client call. Possible answers to complex cooperation requests are acceptance, reject or counterproposal. Except for acceptance, further negotiation between traders can follow, consisting of a series of proposals/rejects and counterproposals. If agreement can be achieved, the cooperation is established. Otherwise it is finished without success. Modification and removal of cooperations are performed likewise.

Based on the above described logs, the accounting component calculates charges per trader client. A problem arises because of current communication practice: there is no mechanism available for traders to get in contact with service providers and service users (except a special method to call for values of dynamic service properties). As a consequence, they do not know how to send bills to their clients. This problem can be solved either by demanding a fixed amount of money to be paid before the trader service is provided. An alternative consists in letting clients specify a suited method to be called for charging them.

4.2 Partner management

To be able to cooperate, traders must know about other traders. A first approach to achieve this consists in using a directory service (e.g. X.500). But according to section 3.2, parameters characterizing potential partners are of more or less dynamic nature. Furthermore, the parameter “round trip delay” depends on two traders. Therefore, only properties like name and location should be managed by a directory service. Based on this service, a trader can find existing traders and send them a full characterization of itself. This mechanism is a subclass of service offers in general and therefore does not require any specific treatment. Round trip delay has to be determined individually for each receiver.

Each incoming characterization of another trader is presented to the system administrator or to automatic control. If a trader seems to be a promising candidate for further cooperations, its characterization is stored by the component for partner management. This can hold even for the case, that not all dynamic properties have ’good’ values. Because of their dynamics, this situation can have changed completely, when a cooperation really has to be initialized. To refresh values of dynamic properties, efficient update mechanisms can be used (cf. [KoWi94]). Characterizations can be enriched by private information about these traders, e.g. to reflect recent experiences with them. A characterization of a certain trader can be removed either by the characterized or the storing trader.

4.3 Cooperation management

A cooperation is described by partners involved and its state. After initialization, valid states are ’negotiation phase’, ’established’ and ’finished’. Such descriptions are stored by the cooperation management. For established cooperations, the following information is added (cf. federation contract of [BeRa91]):

- lists of service types that are exchanged,
- range of allowed operations,
- conversion information for heterogeneous type managers,
- chaining flag to indicate allowance or not of indirect cooperations.

If a cooperation has been finished, it is either removed or some comments are added to describe its success.

4.4 Administrator interface

The user inferface for administrators basically consists of two parts:

- inform about cooperation relevant events,
- communication with other traders.

4. Architectural framework

In the following, we describe components to observe, log and evaluate the information as explained in the last section. These components must be added to the architecture of a cooperating trader (see e.g. [Burg95], [dPLJMa95], [PuBu96]), which contains components to manage potential partners and current cooperations. All relevant data are presented to the system administrator. According to his/her decisions, cooperations are initialized, modified or finished. The overall architecture is shown in figure 2. A more detailed description can be found in [Fehr96].

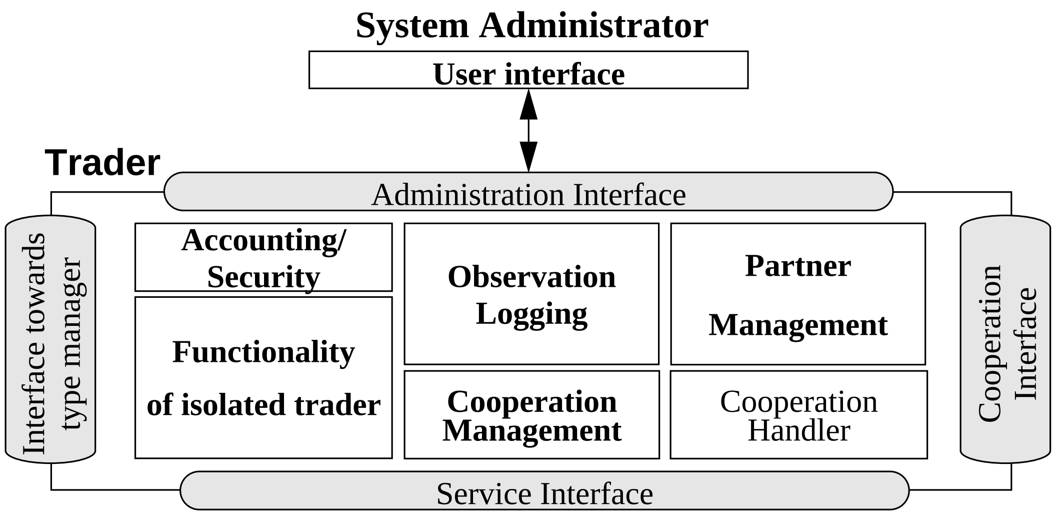


Figure 2: Architecture of cooperative trader

4.1 Observation, logging and accounting

To provide information as required by controlling entities like administrators via user interface or automatic control, a log is performed for each call being received by the trader. Logs consist of

- type of the call, e.g. request by service user, offer by provider or element of some negotiation protocol among traders,
- start and completion time of the call and required storage size of offers for considerations about performance,
- one or more service types that are requested resp. offered by service users, providers or other traders,
- success or failure,
- identification of sender of the call to be able to distinguish calls with same characteristics but different senders.

Furthermore, the controlling entity is notified about cooperation relevant events, e.g. about failure of a request with a certain service type. Besides this logging, counters for failures, cooperations etc. are used for a quick overview.

characterized, because heterogeneous type systems require extra effort for type conversions (cf. [BeRa91], [VBB95]). Finally, parameters for times and fees are needed to enable economical reasoning (cf. section 5). According to [LeBe95], all these parameters should be changed according to experiences.

3. Cooperation relevant parameters

From the above it can be concluded, that a lot of information is required to decide properly about cooperations. In the following, this information is summarized, while treating parameters of the trader that is reasoning about cooperation separately from descriptions of potential partners. For both cases, it is crucial to define as less properties as possible but with most expressive power. The usage of this information is explained in section 4 and 5.

3.1 Parameters of the local trader

To be able to detect cooperation need, the following information about a trader's state is necessary

- number of requests/offers that cannot be handled locally because of
  - unknown service type
  - policy restriction
- current performance + duration of this state:
  - number of offers stored in database
  - current queue length of offers/requests
  - mean response time for storing/matching.

If cooperation need arises, further parameter are used to properly decide about this cooperation

- number of cooperations
- per cooperation:
  - cost
  - number of clients that profit,
  - amount and frequency of information exchange.

3.2 Description of potential partners

To estimate success of cooperations, we propose the following parameters to describe properties of potential partners and their relations to the deciding trader:

- Location.
- Short description for spectrum of managed service types, e.g. software renting, mail order catalogue or travel services (cf. standardized supertypes of yellow pages).
- Short characterization of managed service providers concerning mean load, availability, interval of prices and trustworthiness.
- Characterization of type manager, free storage capacity, mean time for matching a service request and availability of the trader.
- Fees to be paid for each storage/time unit resp. each time unit of a matching process.
- Mean round trip delay between characterized and informed trader.
- Miscellaneous like administrative membership (suggesting a certain kind of policies) or number of known traders.

Depending on actual cooperation need, either type information or storage and computing capacity of potential partners is playing the more important role. Involved type managers have to be

$$\min_{\text{pathes}} \sum_{\text{traders on this path}} [2 \text{ x communication delay} + \text{execution time}]$$

if a tree-like forwarding has to be performed. This term is added to local execution time, because in general cooperation decisions are driven after local execution has failed somehow (in contrast to the case of too much load). Estimation of this additional delay is only possible for direct cooperations.

2.2 Financial cost

Up to now, concepts for accounting of trading service have not been published yet. But it is reasonable to assume, that fees depend somehow on resource consumption. Only this assumption is relevant for the paper. Therefore, consideration of cost can be treated analogous to performance. Additional cost arise because of

- increased consumption of local resources
- fees demanded by cooperation partners.

How much traders will have to pay, is subject to negotiation before cooperation. For instance, discounts could be applied to partner traders or agreements about restriction on mere information exchange without any fees could be made, thereby decreasing the second part of additional cost. But in general, additional cost arise and can be estimated for direct cooperations. Depending on the trader's policy, these cost can be divided by the number of those client requests and offers that profit from cooperation.

2.3 Security

Analogous to human individuals in real world, traders start with a certain trust in their environment (because of being controlled by humans). According to their experience, this trust will be adapted and changed permanently. Four levels concerning a certain cooperation partner and the environment can be distinguished:

- If the partner as well as the whole environment can be fully trusted, no further mechanism is necessary.
- Authentification of the partner is needed for insecure environment.
- If the partner is reliable only in parts, information transfer is restricted and received information is not used. For instance, if service providers specify different confidence levels, the most confident offers are not transferred. In the opposite direction, information about prices can be replaced according to recent experiences.
- If a partner has proven to be unreliable, no further cooperation will take place with it.

A precise definition of trustworthiness of partners and the evaluation of experiences are affair of system administrators.

It depends on the trader's policies if it takes the risk of indirect cooperation or tries to prevent its partners from forwarding its cooperation need.

To be able to decide about a certain cooperation between traders, use and cost of this cooperation have to be estimated and compared. Depending on his/her policies, each human provider of a trading service will have different preferences. But in general, the following criteria will play a role

- performance and availability,
- financial cost,
- security.

Applying these criteria to trader cooperations, delivers results as explained in the following.

2.1 Performance and availability

There are two causes, how trader performance is decreased:

- trader internal or local reasons like too much load or being unavailable,
- cooperations.

In the first case, cooperations and the mechanism of load sharing can help to enhance performance. To this end, cooperation decisions must be driven before any local execution has taken place. Consider for example, a trader with a very large database of service offers and a very long queue of user requests on one and suited quick traders on the other side. But it is difficult to detect such cases because only response times for cooperations with direct partners can be estimated and success of cooperations cannot always be predicted. Therefore, cooperation to decrease response times should be initiated without any further consideration but only, if the trader or the local system have been overloaded for some time and this state will continue (cf. section 5).

The case of being down or having unavailable communication links is even worse than having too much load, because it cannot be hidden from clients. Independently of predictability of this state, the trader can a priori negotiate with other traders to substitute it and inform those clients it knows of. In the current model of trading, only addresses of service providers having made an offer are known to the trader. Advanced concepts like [Wann95] would allow to notify permanent service users also about substitutes in case of unavailable traders. New clients addressing themselves towards the trader while it is unavailable, cannot be informed in neither case.

On the other hand, cooperations are influencing performance by causing additional effort. In detail, cooperating traders have to (cf. figure 2)

4. manage information about potential partners
5. manage information about each actual cooperation (name of partner, status etc.),
6. observe the trader's events to detect cooperation relevant ones among them,
7. optionally: exchange information during negotiation phase,
8. exchange information concerning a certain client request or offer.

As a consequence, response times of all clients get worse, because they are increased by all of these above listed contributions. This holds especially for the client that caused a cooperation, for which the last term looks as follows in its worst respectively its best case

$$\max_{\text{pathes}} \Sigma_{\text{traders on this path}} [2 \text{ x communication delay} + \text{execution time}]$$

2. Characteristics of cooperating traders

Traders are handling service offers and requests. Offers are stored in the trader's database, requests are matched against offers. To achieve this, offer as well as request contain at least some specification of the service type. The actual behavior of a trader is determined by its own policies and those of its clients. For instance, trader policies can dictate how offers must look like or restrict the set of admitted requestors per offer. The latter can be specified by offering clients also. Requesting clients can define search scopes (cf. figure 1) and selection criteria for matches. Traders are autonomous insofar as they carry out their policies and possess knowledge about their clients. It is up to them to share this knowledge with other traders or not.

The need to cooperate with other traders evolves, if one trader cannot fulfill its functionality alone while obeying its policies. The following examples shall give an impression of trader's cooperation need:

- For a certain request, there does not exist any matching offer. Maybe this holds simply because of restrictions dictated by a service provider or because of selection criteria defined by the requestor (e.g. limit concerning cost or trustworthiness), but the effect is the same.
- A certain offer cannot be stored because of limited storage capacity or policy constraints. For example, a trader has specialized in a certain subsection of service types and has received an offer with a type outside this region.
- Service providers are complaining because only a few service users are mediated to them.
- The trader is not able to answer incoming requests in a reasonable time because of being overloaded, or clients cannot be served at all, because the trader is down. In this context, duration of the unpleasant state determines the necessity to cooperate.
- Arbitrary economic reasons like e.g. expensive resources and cheap offers of other traders.

On the other hand, satisfaction of cooperation need means giving up part of a trader's autonomy and causes security problems as well as some kind of cost. This holds especially, if more than one trader are involved and the cooperation request is forwarded from one trader to others in tree-like fashion (cf. figure 1). These other traders are not a priori known to the initiating trader

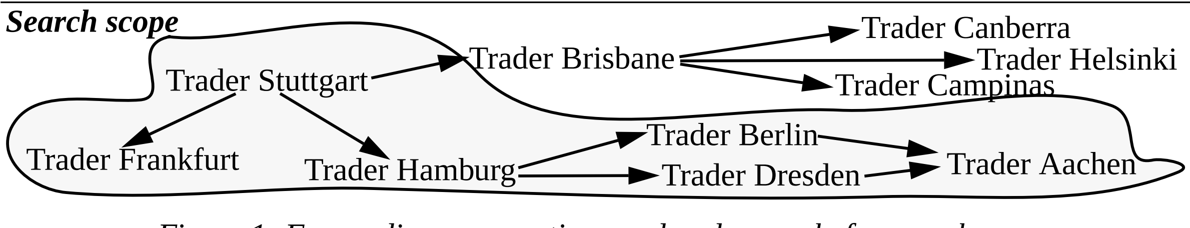


Figure 1: Forwarding cooperation need and example for search scope

but possibly restricted by the client's search scope, e.g. to traders of one nation. Because of forwarding, two different kinds of cooperations exist:

- direct cooperations
- indirect cooperations.

Negotiations and permanent control by participating traders can be performed for direct cooperations only. Therefore, indirect cooperations can provide unpredictable and unpleasant results.

Current trading research ([ISO95], [dPLJMa95]) mostly assumes human administrators to answer these questions but leaves open problems like detection of cooperation need, definition of relevant information about potential partners and the whole environment as well as the way of delivering this information to administrators. Furthermore, dynamics of electronic markets must be taken into account with respect to two different aspects. First, some of the information types will be dynamic too (e.g. load and fees). Therefore, appropriate update and statistics are necessary to base cooperation decisions on valid data. Secondly, some decisions must be driven very rapidly, a requirement that cannot always be met by human administrators. As a consequence, human decisions should be at least supplemented by some kind of automatic control.

For system administrators as well as for automatic control, components are needed to provide relevant information. In the following, we start with working out suited information types (section 2 and 3). The architecture to observe and evaluate this information is described in section 4. In section 5, limits of cooperation control by humans are demonstrated by measurements with traders of the MELODY project (Management Environment for Large Open Distributed sYstems) and mechanisms for automatic control are derived that take requirements of electronic markets into account.

1. Introduction

The importance of electronic markets and usage of electronic services are growing. For instance, people will not buy expensive software that is needed very seldom and changes rapidly (e.g. tax software). Instead, they will look for suited service providers who permanently run the latest version of software products and make them available to remote users at much lower cost with much more computing power than local PCs. An immense number of such services will arise with very different levels of quality, cost and trustworthiness (cf. development of WWW-servers). Individual clients are overcharged by this vast quantity. Therefore, appropriate support is required to assist them in finding suited providers. The trading approach currently being standardized by ISO (see e.g. [Bear94], [ISO95]) aims at providing this support. Traders serve to store service offers from service providers and to answer search requests from arbitrary clients. While performing their task, they are consuming resources and will have to pay for them. As a consequence, for survival in electronic markets, traders have to charge for their mediation services.

Because of a number of reasons like worldwide distribution, administrative division or specialization, one has to cope with a large number of trader entities (cf. [Kutv95]). Service users and providers will only address themselves to a few traders and rely on them as people are doing e.g. with their doctors to get suited medicine. Thus, each single trader is serving a limited number of clients only, while obeying its own policies. These policies control the trader's behavior by defining rules about e.g. admission and exclusion of clients or mode of paying. As long as the autonomy of single traders is not affected, they can cooperate by exchanging information about their clients. This yields advantages concerning

- quantitative aspects of trader functionality, because the set of offered services and the set of potential service users can be enlarged
- performance, e.g. by agreements about load sharing
- increased availability, e.g. by announcing a substitute for the case of being unavailable (cf. substitutes for doctors in emergency cases)
- the economical viewpoint, e.g. outsourcing of trader functionality.

At the other hand, cooperations are causing additional overhead, cost and security problems. There are two complementary answers to this attack on autonomy: properly deciding about cooperations and negotiations before starting a cooperation. Our main focus lies on the first of these aspects. If a trader is considering, whether or not it should join a certain cooperation or leave it, the following decisions have to be driven:

1. When to initiate or finish a cooperation
2. Choice of cooperation partners: selection of individuals and number of partners
3. Choice of cooperation form:
   - Kind of initiating a cooperation: without or with negotiation between trader entities
   - Type, amount and frequency of information exchange

Content

1. Introduction 1

2. Characteristics of cooperating traders 3

   2.1 Performance and availability 4

   2.2 Financial cost 5

   2.3 Security 5

3. Cooperation relevant parameters 6

   3.1 Parameters of the local trader 6

   3.2 Description of potential partners 6

4. Architectural framework 8

   4.1 Observation, logging and accounting 8

   4.2 Partner management 9

   4.3 Cooperation management 9

   4.4 Administrator interface 9

   4.5 Change to functionality of isolated trader 11

5. Automatic control 12

   5.1 Related research 12

   5.2 Architecture 12

      5.2.1 Policy management 12
      5.2.2 Decision about cooperation 13

6. Conclusion 14

Controlling cooperations among traders

Kurzfassung

Mit der Zunahme elektronisch angebotener Dienste wächst der Bedarf an einer Vermittlung zwischen Kunden und Diensten, wie sie durch das Trading geleistet wird. Bedingt durch die geographische Verteilung und unterschiedliche organisatorische Zugehörigkeit sowie die Spezialisierung der Dienste ist von einer Vielzahl einzelner Trader-Instanzen auszugehen, die jeweils autonom sind und eine eingeschränkte Menge an Diensten vermitteln. Um dennoch möglichst viele Vermittlungswünsche möglichst gut befriedigen zu können, müssen die einzelnen Trader Informationen austauschen und miteinander kooperieren. Die Ausprägung solcher Kooperationen wirkt sich jedoch stark auf Kosten und Qualität des Trader-Dienstes insgesamt und auf die Autonomie von Tradern aus.

Aus diesem Grund sind sinnvolle Kooperationsentscheidungen nur durch eine permanente Überwachung und Auswertung aller kooperationsrelevanten Faktoren möglich. Im folgenden wird beschrieben, wie Trader ausgerüstet sein müssen, um eine solche Überwachung zu ermöglichen. Damit wird eine Unterstützung bei Kontrollentscheidungen sowohl für Systemadministratoren als auch für Komponenten einer automatischen Steuerung bereitgestellt. Eine solche automatische Steuerung wird zumindest ergänzend benötigt, da Menschen nicht immer in angemessener Zeit reagieren können. Zur Bewertung wurden Messungen an MELODY-Tradern herangezogen.

Abstract

With increasing numbers of services, traders mediating service users and providers will play a more and more important role. Because of worldwide distribution, a large number of traders will exist. Each of them obeys its own policies and restricts itself to a certain subset of all services. To overcome this restriction, traders have to exchange information and cooperate. But besides counteracting autonomy of traders, cooperations are influencing cost and quality of trader services by additional overhead.

Therefore, cooperation decisions cannot be driven without permanently observing and evaluating all relevant factors. In the following, we first describe monitoring of traders having twofold usage: system administrators as well as automatic control components can base their decisions on its results. At least some kind of automatic control is needed, because humans cannot always react in appropriate time. The underlying concepts of automatically deciding about cooperations are depicted also. Evaluations were made with MELODY traders.

Universität Stuttgart
Fakultät Informatik

CR-Klassifikation: C.2.4, D.3.4, H.5.2

Controlling cooperations among traders

Cora Burger

Fakultätsbericht 04/1996
Technical Report
Juli 1996

Fakultät Informatik
Institut für Parallele und
Verteilte Höchstleistungsrechner
Universität Stuttgart
Breitwiesenstraße 20 - 22
D-70565 Stuttgart