Cooperation policies for traders

C. Burger

University of Stuttgart, Institute of Parallel and Distributed High-Performance Systems (IPVR),
Breitwiesenstr. 20-22, D-70565 Stuttgart, Germany
E-Mail: cora.burger@informatik.uni-stuttgart.de

In ODP a general mechanism has been constructed, how cooperation between trader entities can be performed. But little has been said about when and in which cases such a cooperation should be initiated. Furthermore, the above mentioned cooperation mechanism requires an enormous amount of computer and communication resources, counteracting the profit of cooperation, e.g., by degrading the mean response time of traders. As a consequence, policies are needed to resolve such conflicts.

In the following, a general cooperation model is introduced and applied to traders. From this general model cooperation policies for traders are derived. One of these policies can be adapted to optimize the quality of trader service by observing the dynamic behaviour of traders, their clients and servers.

Keyword Codes: C.2.4, H.4.3

Keywords: Computer Systems Organization, General; Distributed Systems; Information Systems, General; Communications Applications

1 INTRODUCTION

Because of increasing size and complexity of distributed systems, traders are needed to bring those objects together that want to build a client-server-pair. This topic has been widely studied before (see, e.g., [ANSA91], [Burg90], [KoWi94], [ISO/93]).

Even in small distributed systems, more than one trader can exist to avoid performance bottlenecks and increase fault tolerance. By connecting a number of such smaller systems to enable cooperation among organizations, a set of traders arises automatically. I. e., the existence of more than one trader in a distributed system is quite reasonable. By cooperating and combining their facilities in a suitable way, traders can ameliorate their functionality as well as their quality of service and stay autonomous nevertheless.

Former studies on cooperation between traders concentrated on the mechanisms needed to achieve cooperations. We classify three approaches, that have been used to cooperate in the case, that a certain client request cannot be treated locally:

- light weight cooperations by acting as a client of another trader ([ISO/93]),
- negotiations for importing exactly one service for a certain period of time ([NiGo94]),

- federations to enable cooperation for a certain set of services during a certain period of time including the handling of heterogeneity between traders ([BeRa91], [ISO/93], [SPM94]).

The only cooperation policy proposed so far, is the one to ask for cooperation help each time, a trader fails to answer a certain client request ([NiGo94], [DIN92]). To motivate further treatment of this topic, let us consider two examples of daily life. While searching for a very important reference in a local library, a scientist would be happy to get the desired article or book from another library, even if it is located at the other end of the world. On the contrary, if a single client asks about an unknown product in a supermarket, the manager would certainly not run to his or her telephone and try everything to purchase the desired product. From these examples it can be concluded, that the above mentioned cooperation policy leaves open a number of important questions:

1. In which cases is it reasonable to cooperate for each failing answer to a client request? Are there further cases in which a trader should ask other traders for cooperation help? Both questions concern the so-called IMPORT case.
2. In which cases should a trader offer cooperation help to other traders (the so-called EXPORT case)?
3. In which cases should a trader withdraw from a cooperation?
4. Which services should be included in a federation between two traders?
5. Which cooperation form (light weight, simple negotiation or federation) should be applied in which cases?
6. Which are the criteria, the answers to the above questions should be based on?

With these questions a new and complex area with a large variety of aspects is entered that must be studied by an incremental approach. In the following, we base the answers to the first five questions on the detection of cooperation need in a general cooperation model, on service usage by clients and on quality of service parameters of traders and servers. The consideration of cost and security is beyond the scope of this paper and remains a topic for future research.

The general cooperation model and its application to traders are introduced in section 2. These studies result in an answer to the first and second question by stating how cooperation need is detected. From this cooperation policies for traders are derived in section 3. The third, fourth and fifth of the above posed questions are answered by realizing policies and possibly adapting them according to the dynamic behaviour of clients, traders and servers. Section 4 describes the current state of the project and outlines future research topics.

2 TRADERS AS COOPERATING SUBJECTS

To study trader cooperation on a broader base, we introduce a general cooperation model (cf. [BuSe94]). A cooperation is built by a number of subjects. Depending on their facilities and goals, subjects decide how to join a cooperation. Thus, the behaviour of subjects with respect to a cooperation can be defined by the following terms that depend on time:

subject = ((facilities, cost), (goals, weights))

In this model, facilities comprise

- those actions that the subject can perform,
- the resources being needed to perform actions,
- the subject's knowledge about itself, about other subjects and its environment.

Facilities can be offered to other subjects by charging cost, a term, that is presented for the reason of completeness (cf. section 4). Goals can be divided in those concerning the execution of tasks and in qualitative ones. To order them for an execution schedule or to resolve conflicts, goals can be weighted.

In the following, this general model is applied to traders by describing their facilities and goals.

2.1 Trader facilities

The kinds of trader facilities as described above can be derived easily from the well known functionality of traders.

Trader actions concern the handling of clients requests and servers offers. In both cases a number of options like e.g., search scope and selection criteria in requests or like criteria to restrict offers to a set of importers can be involved. Because each option requires a different action, we have a wide spectrum of possible trader actions. Normally, not each trader is able to perform all those actions. For example, not each trader can take into account all existing selection criteria.

Trader resources comprise computing and communication resources, especially disk space, as well as the algorithms being used to perform actions. Some resources have limited capacities and therefore restrict the amount of facilities. This plays an important role especially with respect to the trader's response time and the number of service offers that can be stored. As a consequence, traders with different resource capacities exhibit different parameters of quality of service.

Trader knowledge can concern histories of service usage by clients, properties of registered servers and properties of other traders. A trader either knows all traders in a system or a subset, e.g., all neighbours in a spatial relationship (the possibility of trading for traders, i.e. a recursive trading, is not considered here). To avoid isolated traders in the second case, it has to be guaranteed, that the union of all subsets is equal to the set of all traders willing to cooperate. Furthermore, knowledge of traders does not only differ with respect to registered servers but also to other traders.

From the descriptions above, we conclude that traders with very different facilities can exist. If they combine them by means of a cooperation, each of them can profit. It should be noted that actions and knowledge are published explicitly whereas resources are shared only in an indirect way. The amount of offering and exporting its own facilities to other traders, depends on goals that are described in the next subsection.

2.2 Trader goals

In the following, only those trader goals are considered, that are relevant to cooperations. Furthermore, cost and security aspects are not taken into account (cf. section 4). Cooperation

relevant goals can be divided into three different types and partially subdivide in functional and qualitative goals. Some of these goals are in conflict. The following informal description of these goals is quantified in subsection 3.1.

- Client biased goals of traders:
 - Answer each client request.
 - Optimize allocation between client and server according to client biased criteria, e.g., by minimization of server response times.
- Server biased goals of traders:
 - Offer to a large set of potential clients to increase probability of client requests.
 - Optimize allocation between client and server according to server biased criteria, e.g., throughput of one server or of a group of servers.
- Trader specific goals:
 - Exhibit a high quality of trader service.
 - Cooperate with other traders if necessary.

How these goals are weighted, has to be defined by the administrator of the system. In general, traders can differ in the set of supported goals and in weights of these goals.

2.3 Cooperation between traders

Partners want to cooperate if they detect a discrepancy in a positive or negative sense between their facilities and some or all of their goals. For traders whose cooperation relevant goals are included in the set as described in subsection 2.2, the following discrepancies can appear:

- Lack of knowledge about servers or traders.
- Set of servers too small to optimize according to client biased criteria.
- Set of clients too small to have a reasonable number of allocations to servers or to optimize according to server biased criteria.

As can be seen, there are four reasons to favour a cooperation, three for importing and one for exporting facilities. Thus, answers to the first and second of the questions about cooperation need are found (cf. introduction). Furthermore, it is obvious how to answer the third question: if the discrepancy between facilities and goals has been removed, the cooperation is ended.

If another goal contradicts the desire for cooperation, this conflict has to be resolved by weighting and modifying goals. Depending on their weights, either the cooperation willingness or the contradicting goal has to be modified.

The effort to initiate and maintain a cooperation can degrade the quality of trader service substantially. Especially response times are influenced negatively, even for those requests, that can be treated locally. Thus, the goal to exhibit a high quality of service by optimizing the response time of a trader, is an example for a goal that can contradict cooperation desires.

3 COOPERATION POLICIES

A cooperation policy determines weights of goals and the cooperation mechanism to be used. For the trader specific goals as defined in subsection 2.2, the policy function takes the following form:

$$policy = w_{coop} \cdot CooperationWillingness + w_{QoS} \cdot QualityofService$$

where the weights $w_{coop}, w_{QoS} \in [0,1]$ and $w_{coop} + w_{QoS} = 1$. Depending on these weights, two cases can appear:

- $w_{coop} = 1, w_{QoS} = 0$: The resulting behaviour exhibits as much cooperation as necessary to achieve all client and server biased goals according to their weights, whereas the quality of the trader service is not taken into account. This case coincides with the example of a scientist searching for a very important reference in a local library (cf. introduction).
- $w_{coop} < 1, w_{QoS} > 0$: As long as the quality of trader service is not affected, this case coincides with the last case. If the service quality decreases because of too much resource consumption by cooperations, the cooperation willingness is adapted according to weights. For $w_{coop} < \frac{1}{2}$, this case can be compared to the example of an unknown product in a supermarket being asked about by a single client (cf. introduction), where quality of service is weighted higher than cooperation willingness.

In both cases, a suitable cooperation mechanism has to be chosen depending on the actual situation. In the second case, adaptations have to be performed to modify the cooperation willingness. For the realization and especially the adaptation of cooperation policies, suitable parameters and mechanisms are needed. They are described in the following.

3.1 Quantification of goals

To be able to realize the cooperation policies depicted above, the goals of subsection 2.2 must be quantified. To this end, each of these goals has to be examined to find suitable parameters. Furthermore, for the second case of cooperation policies the adaptability of parameters is required.

The client biased goal to answer each request of clients is a fixed one, i.e. it cannot be adapted. Therefore it is replaced by the following weaker but variable goal:

- If a certain number N_{fail}^{max} of client requests concerning the same unknown service type S is reached during a certain time interval, a cooperation must be initiated to find a trader who is responsible for this service type.

In this goal, the parameter N_{fail}^{max} can be varied. For the case $N_{fail}^{max} = 1$, the weaker form coincides with the original one.

Both optimization goals, the client biased as well as the server biased optimization, could result in load balancing across organisation boundaries. Because of organizational problems and perhaps of problems with heterogeneity, we propose to use load sharing by applying a simple mechanism with thresholds (cf. the classification of allocation mechanisms in [Burg90]). To this end, the following parameters are needed:

- Upper threshold $UT_{server}(S)$ for load of servers of a certain service type S .
- Lower threshold $LT_{trader}(S)$ for client requests concerning a certain service type S and lower threshold $LT_{server}(S)$ for load of servers of this type.
- The period of time T , expressions with the above defined parameters must have been valid before being taken into account.

The degree of cooperation willingness is defined by taking all parameters of this subsection together. The usage of these parameters is described in subsection 3.2. Furthermore, they are subject to modifications that are needed to adapt the cooperation willingness (cf. subsection 3.4).

3.2 Realization of cooperation policies

The algorithm to realize the cooperation policies introduced above proceeds straight forward. The following tasks are accomplished:

- Count those client requests concerning a certain unknown service type S that appear with temporal distance less than T . Compare this counter with N_{fail}^{max} . If both values are equal, initiate a cooperation to try to import S .
- Compare the load of servers of a certain service type S with $UT_{server}(S)$. If the load exceeds this value during the whole period T , initiate a cooperation to import S .
- Compare the usage counter for a certain service type S with $LT_{trader}(S)$. If the counter is below this value during the whole period T , compare the load of servers of this service type with $LT_{server}(S)$. If the load is below this value during the whole period T , initiate a cooperation to export S .

At start time, N_{fail}^{max} is set to one and thresholds are set to predefined values. By modifying these values, the cooperation willingness of a trader can be influenced (cf. subsection 3.4).

3.3 Choice of cooperation form

In the last subsection, the conditions for a cooperation initiation have been described. Figure 1 shows three different forms that can be used to perform such initiations (cf. introduction): one trader acting as a client of another trader (light weight cooperation), traders negotiating with respect to one service (simple negotiation) and traders negotiating the amount of mutual use of facilities and solving heterogeneity problems (federation). These forms differ as well in the amount of resource consumption as in their abilities.

Figure 1 : Cooperation forms

To be able to make a reasonable choice that depends on the current situation, we classify according to the following criteria:

- Homogeneity respectively heterogeneity of service types of those traders that participate in a cooperation.
- Number of services that are going to be exported or imported.
- Frequency of service usage.

The algorithm to choose a cooperation form takes the one with lowest resource consumption, that is powerful enough to cope with all characteristics of the current situation. In the following we shortly describe the conditions for each cooperation form.

For light weight cooperation, service types of participating traders must be homogeneous. Furthermore, no more than one service type may be concerned and it may not be requested very frequently. Otherwise the cooperation form with simple negotiation should be used. It applies to the homogeneous case too and concerns one service as well. But the service is requested more frequently. Because federations are the most heavy mechanism to cooperate, they should only be used if necessary. Thus either the heterogeneous case applies or more than a certain number of servers is involved or both.

There are two parameters which care for differentiating between usage of different cooperation forms: the limit concerning frequency of requests divides light weight cooperation and simple negotiation, the maximum number of servers that are handled by simpler mechanisms

separates them from the federation mechanism. Both parameters can be subject to an adaptation to influence the quality of trader service.

3.4 Adaptation mechanism

According to weights of goals in the second class of coordination policies, we base the modification of cooperation willingness on the observation of the quality of trader service: if the quality of trader service increases, its cooperation willingness increases too, whereas it decreases for the case of decreasing quality of trader service. The amount of increase or decrease can be defined as a constant or it can depend on the gradient of the quality of trader service. In figure 2 this relationship is shown by taking the mean response time as an example for the quality of trader service. The modification of the cooperation willingness is realized by modifying its parameters (cf. subsection 3.1).

Figure 2 : Adapting cooperation willingness

For the separation between usage of different cooperation forms we proceed in an analogous way: if the quality of trader service decreases, usage boundaries are shifted to be able to choose a cooperation form consuming less resources. To this end, the frequency limit or the number of services required for federation are adapted.

To demonstrate the mechanism more clearly, we study a number of examples. At start time, there is a maximum cooperation willingness, i.e. each client request that cannot be answered locally leads to a cooperation. If response time degrades in the following time interval T , cooperation willingness is decreased by increasing N_{fail}^{max} , the number of failing client requests that are necessary before cooperation is initiated.

The second example copes with lightly loaded servers. If a cooperation has been initiated to offer such a server and the response time of the trader degrades in the following time interval T , cooperation willingness is decreased by decreasing $LT_{trader}(S)$ and $LT_{server}(S)$. If one or both of these thresholds reach zero, this means a very low cooperation willingness.

If the quality of trader service behaves poorly after usage of the light weight cooperation form, the frequency limit should be decreased to allow the use of simple negotiation earlier. In a similar way the number of services needed for federation is increased, if the quality of trader service degrades after a federation. A bad reaction to the usage of simple negotiation can be answered by increasing the frequency limit or by decreasing the number of services needed for federation. These possibilities should relieve one another to avoid loops.

After having described the mechanism of adapting, it must be considered when such an adaptation has to be performed. To achieve this, traders have to observe and analyse their

behaviour. Cases in which the quality of trader service decreases independently of any cooperation, have to be filtered out. For example, the response time of a trader without cooperation has to be estimated to find out the reason for an increase.

4 PROJECT STATE AND FUTURE RESEARCH

The functionality of an isolated trader has been implemented in the MELODY project (Management Environment for Large Open Distributed sYstems) and is documented in [KoWi94]. The architecture consists of two main parts, the trader user agent (TUA) that is bound to clients and servers and the trader server agent (TSA). The agents are written in C++ and communicate via DCE-RPC, therefore object adapters (OA) are needed to map from the object oriented to the procedural model (cf. figure 3). MELODY traders take into account dynamic properties of servers. This is achieved by management mechanisms that observe and collect dynamic information. These mechanisms can be applied to the realization of cooperation policies also.

The MELODY traders are now extended to cooperate with one another. The corresponding architecture is shown in figure 3 (cf. [KoBu95]). The first step consists in implementing a

Figure 3 : Architecture of cooperating MELODY traders

cooperation handler which makes available the three cooperation forms as described in figure 1. By intercepting the case that no suited server has been found by the functionality of an isolated trader, the cooperation manager is invoked to choose one of the operations of the cooperation handler. To start with, this choice will be configured statically. Measurements of the performance of the different cooperation mechanisms will deliver criterions for the adaptive policy, i.e. for thresholds, period of validity and the limits between usage of different cooperation forms.

Two approaches are pursued with respect to cooperation policies: in one case the trader cooperation is controlled by the system administrator, in the other case the policies that have

been described in section 3 are applied. This means answering the questions of the introduction either by the system administrator or automatically.

In a next step, cost and security aspects will be included. To prepare this, cost have already been included in the general model. The need for security could either be appended to each of the trader's facilities or introduced as a global goal.

5 SUMMARY

By considering a general cooperation model, we found a large variety of different types of traders with respect to their facilities, goals, and weights of goals and identified two main cases of cooperation policies. These cooperation policies can be realized by means of quantifying the goals of traders and observing and analysing the behaviour of clients, servers and traders. Furthermore, an adaptation mechanism for the second class of cooperation policies has been defined.

Collecting those aspects, answers to all questions posed in the introduction have been found:

- Determination of conditions leading to a cooperation respectively to its end (subsection 2.3 and 3.2).
- Finding out the service types to be imported or exported by the cooperation (subsection 3.2).
- Determination of the cooperation form to be applied (subsection 3.3).

Thus, traders are equipped with policies to decide on cooperation automatically. This contributes directly to ODP trading.

ACKNOWLEDGEMENT

This work is granted to the German Hochschulsonderprogramm II. Furthermore, the author wishes to thank E. Kovacs for his explanations of the MELODY trading system and for helpful discussions.

REFERENCES

- [ANSA91] ANSA. *ANSAware 3.0 Implementation Manual*. Architecture Projects Management Limited, February 1991.
- [BeRa91] M. Bearman and K. Raymond. *Federating Traders: An ODP Adventure*. International IFIP Workshop on Open Distributed Processing, October 1991.
- [Burg90] C. Burger. *Performance optimization in heterogeneous distributed systems by dynamic allocation of user tasks (in german)*. Doctor thesis, Universität Karlsruhe, Fakultät für Informatik, February 1990.
- [BuSe94] C. Burger and F. Sembach. *A survey of mechanisms for computer supported cooperation (in german)*. Technical Report 7, Universität Stuttgart, Fakultät für Informatik, July 1994.

- [DIN92] German DIN. *A structural specification of the ODP trader with federating included*. Technical Report N1-21.1.1/55-92, ISO/IEC JTC1/SC21/WG7, 1992.
- [ISO/93] ISO/IEC JTC1/SC21/WG7 and ITU-TS SG7.Q16. *Working Document on Topic 9.1 - ODP Trader*, July 1993.
- [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.
- [KoBu95] E. Kovacs and C. Burger. *Support for usage of distributed systems (in german)*. Technical Report 8, Universität Stuttgart, Fakultät für Informatik, February 1995.
- [NiGo94] Y. Ni and A. Goscinski. *Trader cooperation to enable object sharing among users of homogeneous distributed systems*. computer communications, 17(3):219 – 229, March 1994.
- [SPM94] O. Spaniol, C. Popien, and B. Meyer. *Services and Service Trading in Client/Server Systems (in German)*, volume 1 of Thomson's aktuelle Tutorien. International Thomson Publishing, Bonn, tat edition, 1994.