23

Implementing the OMG Trading Object Service - the TOI Product

V. Tschammer, H. Herzog, N. Aghoutane, G. Ercan
GMD-FOKUS
Hardenbergplatz 2, D-10623 Berlin
Tel.: +49 30 25499 200, Fax: +49 30 25499 202
toi@fokus.gmd.de

Abstract

The offering and discovery of services in a globally distributed, open environment is supported by trading object services, implemented by specific trusted entities, called traders. Traders allow run-time bindings between clients and servers, support service providers in advertising their service offers, and help clients in selecting the best offers according to service properties and server qualities. Recently, the OMG has published a paper which specifies a trading object service for the OMG/CORBA environment. This specification is to become a component standard for open distributed processing in the future information society. We describe TOI - the first OMG Trading Object service Implementation, available for the ORBIX environment on Sun Solaris and Windows NT platforms. The implementation is based on a flexible, modular architecture which allows different traders, e.g. simple traders and linked traders, to be configured easily. The TOI trader is embedded in a trading community which includes additional components for policy administration and federation management.

Keywords

CORBA Object Trading Service, OMG Trader, Trading in Open Service Markets

1 INTRODUCTION

On the future global information highway, users of very large distributed applications and global information systems will have access to numerous information sources and tele-services. The offering and discovery of services in this globally distributed, open environment will be supported by trading object services, implemented by specific trusted entities, called traders. Traders allow run-time bindings between clients and servers, support service providers in advertising their service offers, and help clients in selecting the best offers according to service properties and server qualities. Traders are

J. Rolia et al. (eds.), *Open Distributed Processing and Distributed Platforms*

usually embedded in so-called trading communities which are groups of cooperating entities encompassing the trader itself, its related clients and servers, and a set of administrative entities, such as managers of service types and trading policies. Trading communities can be federated so that clients can use offers advertised in other communities.

The OMG and the ISO/ODP committee have recently published papers which specify a CORBA Trading Object Service [1] and an ODP Trading Function [2], respectively. The TOI (Trading Object service Implementation) product implements these specifications in a flexible, configurable way so that different conformance criteria can be met and requirements of different trading environments can be satisfied.

The paper describes the TOI product and elements of the TOI trading community. Conceptual and implementation issues are discussed with respect to flexibility and portability. Future work is outlined which will add value to the trader and improve its development and run-time environment.

2 THE OMG/ ODP TRADER: INTERFACES AND CONFORMANCE CLASSES

The OMG document on the Trading Object Service and the ISO/ODP Trading Function Specification define a similar set of interfaces to the trading object service. These interfaces are separable and trader implementations may select to support various combinations of those interfaces. Therefore, conformance classes have been defined which specify the conformance requirements on a per-interface basis. In the following, we briefly describe the interfaces and the conformance classes.

The OMG and ISO documents specify five functional interfaces to a trading object service: Lookup, Register, Link, Admin, and Proxy.

Lookup: The Lookup interface allows an importer, i.e. a client, to obtain references to servers which have submitted service offers that meet the importer's requirements.

Register: The Register Interface provides the means by which an exporter, i.e. a server, can advertise a service offer to the trading community. Operations for modifying and withdrawing service offers are included.

Admin: The Admin interface is used for reading and modifying trader attributes and for housekeeping on service offers and proxies. Likewise, trading policies are administered via this interface.

Link: The Link interface allows the federation manager to administer the links which interconnect the trader with other traders situated within the own or foreign trading communities. List, add, remove, and modify operations on links are provided.

Proxy: The Proxy interface allows the export and subsequent manipulation of proxy offers.

Based on these interfaces, the following conformance classes are defined:

Query Trader: A trading object service implementation claiming conformance to the query trader conformance class shall meet the conformance requirements of the Lookup interface as server.

Simple Trader: An implementation conformable to the simple trader conformance class shall meet the requirements of the Lookup and Register Interfaces.

Stand-alone Trader: An implementation of the stand-alone trader must satisfy the Lookup, Register, and Admin interface conformance requirements.

Linked Trader: The linked trader implementation is a stand-alone trader which additionally supports the Link interface.

Proxy Trader: The proxy trader implementation is a stand-alone trader which additionally supports the Proxy interface.

Full-service Trader: The full-service trader shall meet the conformance requirements of the complete set of interfaces defined, i.e. the Lookup, Register, Admin, Link, and Proxy interface.

Currently, TOI implements the Linked Trader conformance class as illustrated by (Figure 1). The Full-service Trader is subject to future work.

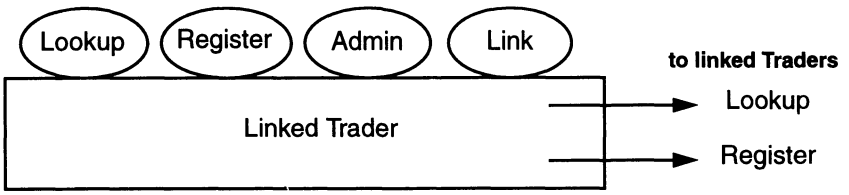

Figure 1 The Linked Trader and its interfaces.

3 TOI, THE OMG TRADING OBJECT SERVICE IMPLEMENTATION

TOI [3] facilitates the offering and discovery of service instances of defined services types which are administered by the type manager. Through the TOI trader, other objects can advertise their service offers and match their requests against offers advertised by other objects. Advertising a service offer is called export and matching requests or discovering offers is called import. Export and import facilitate the navigation through large open service environments and allow late binding of clients and servers.

The basic trading process is as follows (Figure 2): To export a service offer, a server sends the trader a description of a service offer together with the interface at which the offered service is made available (1). To import, a client sends a service request to the trader indicating the service type and the service characteristics and server properties required (2). The trader then checks the request against the service definition provided by the type manager (3) and searches its database for matching service offers. Dynamic service properties are evaluated (4) if requested. On success, the trader returns one or more service interfaces (5). The importer is then able to use the selected service via one of these interfaces (6).

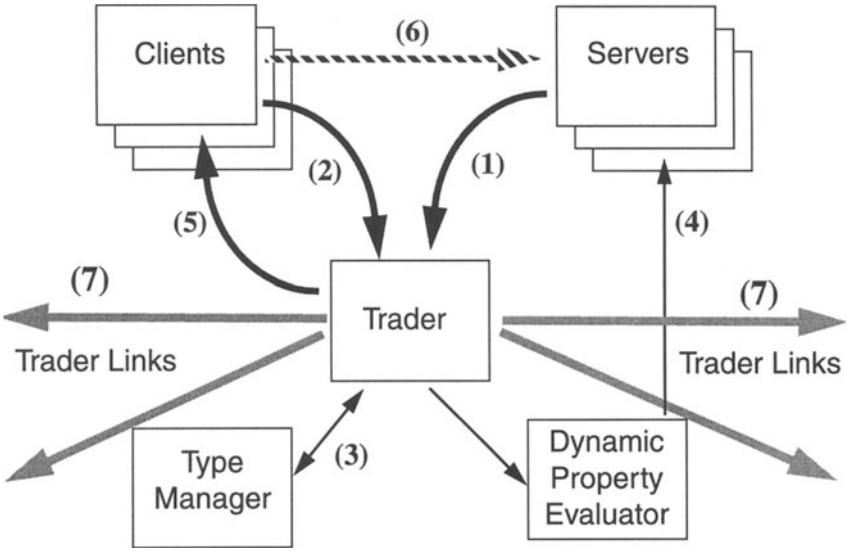

Figure 2 The TOI trader and the basic trading process.

Due to the large number of service types and service offers which will be available in the open, world-wide environment, it is impossible that one trader can serve the different requirements of all the service users. Therefore, it is inevitable that the environment will be partitioned in a number of trading communities and the trading service will be implemented by a number of traders assigned to these communities. Usually, each partition will meet the trading needs of a certain community of importers and exporters, i.e. it will be characterised by dedicated sets of service types and service properties which are well-defined among the servers and clients of that community. If an entity wishes to use a service offered within another community or if a server wants to advertise an offer outside its own community interactions between traders are required. These interactions are usually referred to as interworking or federation of traders which are performed across so-called trader links (7).

The basic trading process, therefore, involves the following:

- The export of service offers.
- The import of information about service offers according to some criteria.
- The interworking of traders across trader links.

Trader federations require that traders which are linked together make their offer spaces available to each other. Likewise, members of the federated communities must be supported in using types administered by foreign type managers. But matching different service types and offers is not the only problem within federations. Service properties and security aspects must be harmonised, too. Likewise, managers and administrators must co-ordinate their activities so that negotiated agreements on the validity and availability of service types and offers are not violated by a single manager's action [4].

In order to meet the various requirements found in the open environment the trading policy concept has been introduced. Policies can be assigned to the export function in order to control the number or identity of clients which are allowed to use the service offer. Import policies can be used to control the search for and the selection of service offers, too. Trading policies can influence the strategy and depth of searches for service offers, and link policies may be applied in order to control the selection and use of trader links. The trader design, therefore, must provide for flexibility of the trader's run-time behaviour and must allow policies to dynamically control the trader's actions.

4 DESIGN AND IMPLEMENTATION FOR DIVERSITY AND SCALABILITY

4.1 Diversity in the trading environment

Various different applications may use the trading service and many different scenarios are possible in the open environment: In large information systems, for example, we can imagine that traders must administer a large number of service offers and serve a similar number of clients in parallel. In mobile and personalised communications, however, the trader may serve only a few clients but must find matching offers in real-time. Virtual malls, home-shopping, and video-on-demand may have dedicated requirements on the diversity and variety of service types and properties and the possibility of interacting with other trading communities in order to find special offers or satisfy unusual demands. Other applications may have specific security requirements. The variations in these scenarios require the trader to be scalable upwards for very big scenarios and downwards for small, real-time scenarios.

The open environment will certainly cover many different technical and administrative domains which requires the trader to operate on different machine and middleware platforms as well as to interoperate across technical,

organisational, and application boundaries. A typical example is the interworking of traders from the client/ server and the desktop environment. The trader design, therefore, must also allow for portability and interoperability aspects.

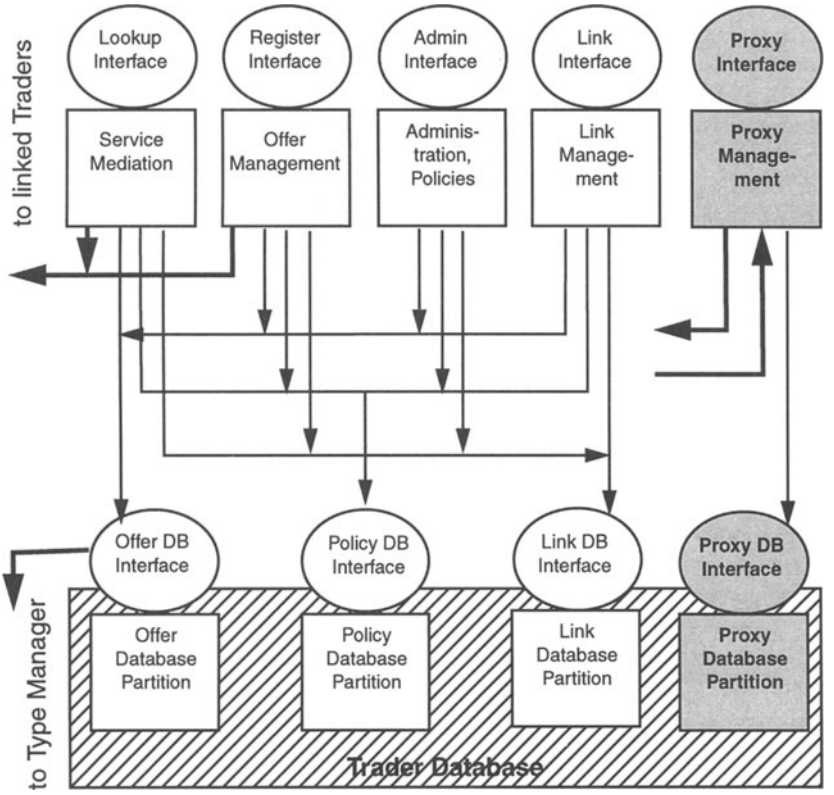

Figure 3 The TOI Architecture for a Linked Trader and extensions for the Full-service Trader.

4.2 Modular TOI architecture and configurable TOI implementation

The TOI Trader design achieves flexibility with respect to the various conformance classes by means of the modular architecture illustrated for the Linked Trader by (Figure 3). Each interface is supported by a dedicated module which implements the methods available and which cooperates with the other modules in order to satisfy the policy directives and the federation requirements. Each of these modules operates on the information stored within the trader database. The database, too, is partitioned into a service offer, link, and policy database. In this way specific trader implementations are easily configured. A simple trader, for

example, which supports the Lookup and Register interfaces only, can be configured by using the offer management, the service mediator module, and the service offer database partition. For the full-service trader a proxy manager must be added to the linked trader configuration supported by a database partition holding the proxy data.

4.3 Scalability with request to varying client and server communities

The TOI product is offered in a single- and a multi-threaded version. The single-threaded version processes requests sequentially. It is intended for small trading communities with a stable set of service types and offers and a simple trader configuration. For large, federated communities of clients, a large, frequently varying set of service types and offers, and a changing environment with frequent interventions of policy administrators and federation managers, a multi-threaded version is provided which allows parallel processing of requests. For the multi-threaded trader, a thread-pool implementation has been found most profitable. In this way, multiple lookup or register requests originating from different clients as well as multiple admin and link management request originating from different administrators can be processed in parallel. This turned out to be a good compromise between the performance and non-blocking requirements of clients' requests on the one hand and the complexity and consistency issues related with interleaved request processing on the other hand.

Scalability is also supported by a flexible implementation of the trader database. Basically, very different database implementations can be supported by the TOI architecture. In small trading communities, characterised by a small number of clients, a fixed set of service types, and a limited number of service offers, the service offer database can be implemented by local memory and the service types stored in a local type repository. In a very large trading community, inter-domain trading and global availability of service offers can be realised by a trader database implementation using X.500 directory services, and the service types be managed by a dedicated distributed type manager service. The TOI database interface, therefore, is configurable and can be implemented as an:

- Interface to an object-oriented DBMS, typically found in client/server environments.
- ODBC-interface for the desk-top environment.
- Interface to local memory database implementation for small, simple traders.
- Interface to an X.500 Directory Service Agent for very large trading communities.

The current TOI implementation uses an object-relational database product for the handling of service offers as well as for the storage of policy and link information. The service types are administered by a separate type manager service.

4.4 Interoperability and portability

Interoperability and portability are supported by using ORBIX [5], a widely accepted CORBA platform product. This product is available on different machine platforms, such as Sun Solaris and Windows NT, which can be regarded as representatives of the client/ server and desk-top environment. Interactions between the different components of the trading community are performed via the standard ORB. Interactions crossing technical domain boundaries, as for example interworking between traders operating on different CORBA platforms, are supported by the CORBA 2.0 inter-ORB protocol. Additional support for portability and interoperability is achieved by making use of available CORBA services. The CORBA event service, for example, is being used in TOI for implementing the interactions between a trader and its servers during the evaluation of dynamic service properties.

The handling of dynamic service properties must be considered with special care, as server implementations may provide this information via different interfaces. X.600 managed objects, accessible via CMIS/ CMIP, and CORBA object interfaces are examples of possible implementations. A dynamic property evaluation module which flexibly supports different information access protocols is another TOI element implemented for interoperability.

5 TRADING POLICIES

Trading policies are rules which guide the trader's behaviour. Typically, trading policies relate events, status, and attributes to required trading goals and actions. Usually, such policies are defined by a policy makers and executed by the policy administrator. Policy makers and administrators usually are agents that act on behalf of human administrators, service users, service providers, and trader owners. Trading policies include trader management policies, export policies, and import policies.

Import policies are defined by service users and are related to the import operation. Import policies include:

- Import-request-acceptance-policy, defining which import requests are accepted by the trader, e.g. imports for certain service types only.
- Local-search-policies, limiting the scope of searches within the trader's database, so that, for example, a trader stops searching after having found a defined number of matching offers.
- Global-search-policies, defining constraints on the search across trader links.
- Resource-consumption-policies, limiting the use of resources, e.g. processing time, that a single trading action can consume.
- Domain-boundary-crossing-policies, providing information needed when crossing service type domains, security domains, etc.

Export policies are defined by service providers and are related to the export operation. Export policies include:

- Service-offer-acceptance-policy, defining constraints on the acceptance of service offers, such as offers for a limited set of service types, offers of limited costs, etc.
- Service-placement-policies, influencing the internal administration and storage of service offers and, particularly restricting the possibilities of transferring offers to interworking traders.

Trader management policies are defined by trader owners and are related to the trader's operations. Trading policies include:

- Type-policies, defining relationships between service types, e.g. super-, subtype relationships.
- Storage-policies, guiding the administration of the trader database.
- Security-policies, restricting the use of the trader by importers and exporters, and defining the authority needed for trader administration.
- Remuneration-policies, defining the costs of trading services.
- Arbitration-policies, defining guidelines for the arbitration of conflicting import, export, and trader management policies.

The policies are executed on the trader by the policy administrator by means of the Admin interface. Inside the trader is a component, in TOI called the policy manager, which reacts on the administrators requests and which controls the internal processing of policy data.

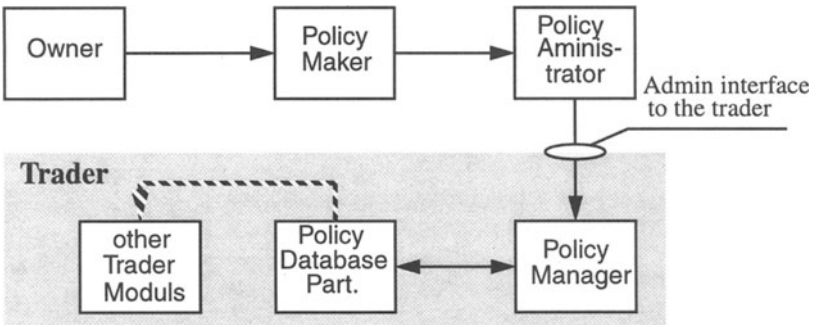

Figure 4 Components participating in the processing of trading policies.

The Admin interface provides create-, modify-, delete-, and list-policy-operations to the policy administrator. The policy manager processes these operations and changes the policy data accordingly. In case of an event related to a certain trading policy, the service mediation, offer management, or link management component which must react on the event evaluates the rules given by the policy and controls its actions so that the policy's goal is satisfied.

6 TRADER INTERWORKING

Interworking traders allow their clients' requests and their servers advertisements to go across their trading communities' boundaries. Interworking is accomplished via the trader links. Trader links contain information about the interworking traders. The standards currently foresee asymmetrical links which allow only a uni-directional information flow from the source trader to the sink trader. Symmetrical links which facilitate trader interworking and link administration will be added to the current TOI product as additional value.

The TOI trading community includes a federation manager and a link administrator for the management of trader interworking. These are agents that work on behalf of the trader owner. Trader owners are humans or application entities that cooperate across trading domain boundaries. The federation manager administers the agreements about the cooperation and the trader interworking and establishes the trader links. The link administrator executes the requests of the federation manager and creates, modifies, and removes trader links via the operations offered at the Link interface of the trader. Inside the trader is a component, in TOI called the link manager, which reacts on the administrators requests and which controls the internal processing of link data. The other trader components consult the link data when processing import or export requests.

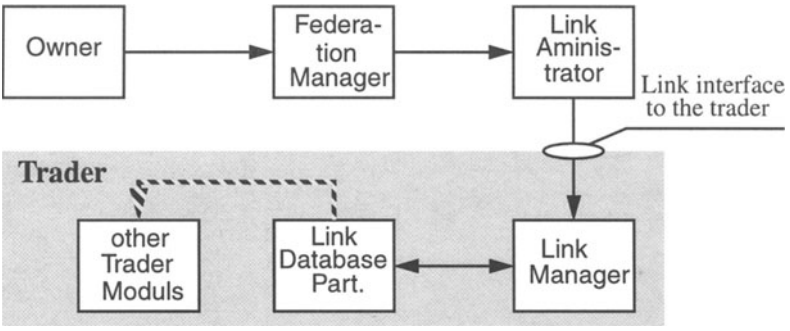

Figure 5 Components participating in the management and processing of trader links.

Agreements administered by the federation manager include agreements about the crossing of domain boundaries and the use of trader links. Trading policies are harmonised by these agreements and a common understanding and mapping of service types is achieved. Federation managers of interworking trading communities also coordinate their administrative actions on service types, offers, and policies, so that, for example, cooperating traders are informed when new service types are introduced or existing types modified or removed. The components involved in the management and administration of trader links are illustrated by Figure 5.

7 CONCLUSION AND FUTURE WORK

The TOI product is the first OMG trading object service implementation available for the ORBIX environment on Sun Solaris and Windows NT platforms. It offers the full functionality as defined by the OMG specification - except for the Proxy Interface which will be provided, soon.

The TOI architecture supports implementation flexibility and, thus, various trader configurations for different service environments can be delivered, such as those defined by the different OMG conformance classes. In this way, the specific requirements imposed by different application areas can be met, including multimedia applications, open service markets, and personalised communication and processing environments.

Future work will concentrate on the implementation of the Proxy Interface, Graphical User Interfaces for trading community components, such as link, policy, and trader administrators, and TOI product support and demonstration. The trader will also be integrated in other support environments, such as the TANGRAM TINA/ DPE implementation [6] and the Distributed Object Management Environment [7] developed by GMD FOKUS within various research and development projects, sponsored by the Deutsche Telekom, DeTeBerkom, ACTS, and others.

8 REFERENCES

[1] OMG RFP5 Submission. Trading Object Service. OMG Document orbos/ 96-06-07.

[2] ISO/IEC JTC1/SC21 N10575. Draft Rec. X.950l ISO/IEC 2nd DIS 13235-1 - ODP Trading Function: Specification. (June 1996).

[3] http://www.fokus.gmd.de/minos/toi/entry.html

[4] Tschammer, V., T. Magedanz, M. Tschichholz, and A. Wolisz. On the Co-operative Management in Open Distributed Systems. *Computer Communications*, Vol. 17, 10, (Oct. 1994).

[5] http://www-usa.iona.com//Orbix/index.html

[6] M.K. Durmosch and K.D. Engel. The Tangram DPE - a Distributed Processing Environment in a Heterogeneous CORBA 2 World. submitted to HICSS-30 Conference, Hawaii, (Jan. 1997).

[7] M. Tschichholz, V. Tschammer, and A. Dittrich. Integrated Approach to Open Distributed Management. Computer Communications 19 (1996) 76-87.