Object-Oriented Framework for the Multicast Applications Integration

M. I. Smirnov

GMD FOKUS
Hardenbergplatz 2, D-10623 Berlin, Germany
telephone: +49 30 25499200; fax: +49 30 25499202; email: smirnow@fokus.gmd.de

Abstract

The paper presents an object - oriented framework aiming at integration of networked collaborative applications strongly demanding smart multicasting capabilities from the telecommunication environment. A new mediator for this support is introduced - the multicast agent (MCA), which is a software object moving across the internetwork and delivering all needed functionality for reliable and flexible participation in a multicast session to all members of the group. The MCA is an object, whose methods provide for the interoperability of heterogeneous applications, the highest achievable scalability, for individual patching of the multimedia multicast flow, and for simultaneous flows from different senders. Based on some precalculated multicast tree (MCT) spanning all dedicated receivers, it is possible to make efficient use of alternatives during the MCT development, the needed solution is presented in a form of a generic behaviour of the splitting point router. It is pointed out that the MCA implementation could be realized within the CORBA approach.

1. Introduction

The future network essential service is most likely to be “store and multicast” [1] for a different number of multimedia objects produced and consumed by various applications running within heterogeneous software and hardware platforms all over the internetwork.

Actually, the *multicast* refers to the network routing service providing for efficient dissemination of the same data packets to various network destinations; the data packets being replicated only within the so called splitting points of the multicast tree. The *multicast tree* (MCT) is a specially constructed virtual structure over existing network topology spanning all participants of the multicast session in the most effective way (known approaches are: minimal total communication cost, minimal delay). The *multicast session* is a multi-user temporary association, where each user runs a set of applications sharing the same multicast connectivity. The *multicast connectivity* is multi-directional in general, i.e. at any time there could exist several senders simultaneously, the rest of the participants being the receivers. Generally each participant may use his own set of applications (audio, video, whiteboard, joint editing tools, CAD, etc.) and define his own requirements for the multicomponent multicast flow. The *multicast flow components,* called here streams, could be characterized by their origination, media, and relations between different substreams (substreams are determined by different media). The user could be represented within this object model as a complex object combining all interfaces of all his applications involved in the multicast session.

As it could be seen from the recent developments of CORBA [2, 3] it’s deployment goes in the direction of advanced support of distributed object systems collaboration, enabling asynchronous messaging, supporting transactions and multi-endpoint interaction and replication of objects. The latter is seen as a support for the reliability improvement (via object replication), for conferencing and groupware. This paper considers heavily needed support from the network service provider for future applications in those areas which will dominate in high speed networks - joint research, concurrent design and development, parallel CAD, etc. On top of existing transport more and more value adding services will be provided in the nearest future.

Almost all of the above named future areas of *Network Supported Cooperative Work* (NSCW) will rely on some applications which most probably will be comprised by the integration of advanced existing tools. This integration is rapidly facilitated by the progress of the Middleware towards mutual interoperability (Fig. 1).

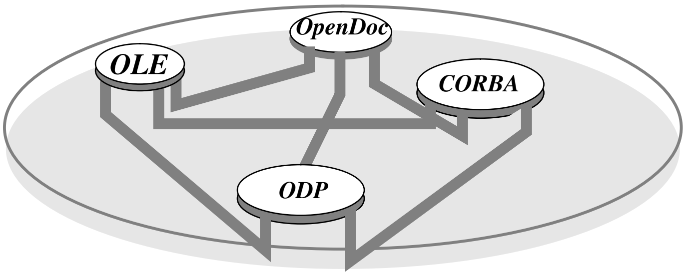

Fig. 1 Platforms integration vision

Based on this platforms integration vision the paper elaborates the idea of the multicast software agents migrating along the multicast tree delivering to each user (complex object) all functionality needed to make local control, customization, patching, etc. of the multimedia streams received by the user. This process is called the *MCT development*.

It is clear now that there exists a strong tendency for various platforms integration and deployment towards mutual coexistence within a framework of common computational telecommunication platform like TINA [4]. Based on this tendency and the networked applications analysis many authorities tend to consider multicasting capability as one of the most prominent network technologies which will shape the future market of the basic telecommunication services and value added services on top of that.

The future multicasting service could be seen itself as a reply to the following challenges:

- integration of existing applications, their migration towards the Intelligent NSCW;
- design of a fairly new applications based on the integral computational telecommunication platform;
- influenced by new applications and platforms the main business processes will change so that new requirements analysis will be needed concentrated on new business directions and tendencies (this probably especially will be true for such business areas as concurrent engineering, CAD, parallel simulation in aerospace and nuclear industry, etc.); special attention is to be paid for changing user behaviour analysis aimed at producing new patterns of the user behaviour for this new environment.

Hence, a typical networking problem such as multicast routing becomes more and more influencing such driving forces as application and network vendors. That is why it seems reasonable to introduce some object-oriented framework according to existing templates of interoperability and, having in mind that this service is essential to the network, try to propose a suitable network solution for this new service within emerging framework.

The rest of the paper is organized as follows. Section 2 gives an overview of a network solution introducing the basic notion of the multicast contract, it's cases, and the network solution for scalable multicast. Section 3 gives an overview of the multicast agent and specifies it's interfaces. Implementation remarks and related work conclude the paper.

2. The Network Solution

2. 1 The Multicast Contract

The initial MCT development scenario is rather simple, however to explain it we need two additional concepts to be introduced such as the *contract* and the *start_time*. The multicast session contract is a sort of agreement between the network service provider (which services currently don't include support for a multicast of depicted style, but are supposed to) and the multicast stream senders. The senders are anonymous at the contract establishment phase (to provide future freedom in their change) except one representative (*initiator*) who will actually start the session, and who will negotiate with the service provider on establishing the contract. More formally the contract could be seen as having the following format:

```
multicast_session_contract(management_domain,
  initiator, QoS_requirements,
  [number_of_simultaneous_senders],
  [session_duration], MCT, [modality],
  [MCT_development_policy].
```

The management domain is needed within the contract because the service provider takes responsibility to assure the quality of service along the whole MCT and during the session's lifetime; the initiator obviously is to be mentioned within a contract, because at the contract establishment phase it is seen as a root of the MCT, however the MCT need not to be obligatory initiator-specific; the QoS could be specified in terms most suitable for involved applications. The MCT parameter provides the network routing information needed to reach all participants in the most effective way [5]. The rest of the parameters are optional and will be explained later.

The second concept, *start_time*, is important for a decision making according to the time period between the contract establishment and the start of the multicast session. We do not consider here totally planned sessions, scheduled (and contracted) in advance, however the same scenario could be used for them as well. The question we have in a light weight sessions is: When the sender can actually start sending the multicast flow to be sure that almost all dedicated receivers are ready to participate in the session? The proposed decision is based on a timeout hierarchy introduced in section 2.3.

Any multicast tree could be seen as a composition of several interconnections of nodes (unicasts) and fanouts; that is any communication system topology supporting and providing multicast could be composed out of several unicasts and simple multicasts. The multicast is called simple when the corresponding multicast tree is a fanout. From these reasonings comes the idea of a scalable multicast. It is evident that due to the inevitable overlapping of unicasts and fanouts the functionalities needed to support unicasts and simple multicasts are to be placed within their intersection points, these points being actually the splitting points of the multicast tree.

Some assumptions for the rest of the paper are:

- this paper considers only network routers participating in the multicast connectivity establishment, while the host nodes where user' applications reside could be seen as the low-level fanouts/unicasts additional to the MCT;
- the MCT unicasts could be hidden in a corresponding fanout presentation;
- each splitting point router is considered to advertise future multicast session to other, than dedicated, participants, belonging to its fanout;
- each MCT fanout is considered to have two types of timeouts, where the timeout applied to the best destination is associated with a "hard" locking of this destinations, which means that the multicast session can't start without acknowledgements from *all of the best* or without the expiry of the timeout; at the same time the advertised routers delay in acknowledgement can not prevent the start of the session (that's why it is called "soft" locking).

2.2 The Multicast cases

The multicast session is considered to enable connectivity of some set D of dedicated nodes (*d-nodes*) where reside group members critical for the future multicast collaboration. However, it could be the case that other participants may wish and be allowed to participate as well. This is a subject of the contract, however it seems helpful to consider 4 cases presented in the Table 1, assuming that any of the receivers linked currently to the MCT could potentially become a sender of it's own flow. Table 1 presents the summary of the *MCT development policy* and gives an overview of possible results for this development. We consider that the multicast session can start only after receiving of acknowledgements from all of the dedicated nodes (if they are mentioned in a contract) or after expiry of the hierarchy of all of the timeouts. The multicast contract is called *limited* if the maximal number of receivers and/or simultaneous senders is limited; otherwise the contract is called *unlimited*.

Table 1:

	Limited Contract	Unlimited Contract
Send to World	Nearest to the initiator *ready to participate* nodes	The resulting configuration is unpredictable
Send to D-nodes	The result will depend on contract modality, availability of best routers and existence of alternatives	Depending on timeouts and alternatives the needed MCT will be established, however with high potential to growth

It is important to distinguish between two kinds of routers participating in this MCT development scenario: the best and the advertised routers. Each of them will receive a copy of the MCA, however with slightly different functionalities. The difference is in the *modality* of their participation within the MCT development. While the best routers *must* participate and must develop their fanouts, the advertised routers *may* participate and may develop the MCT further, based on their local decision making process. If during the MCT development some fanout will experience the timeout expiry for it's best router, then it means that the MCT could not be developed further if this best router is the MCT splitting point. However, the *fanout developmen*t algorithm presented in section 2.3 gives a solution for this problem which actually changes the splitting point based on any available alternative path which was not pruned during the MCT construction algorithm [5]. Figure 2 explains this case. Note, however, that connection from the

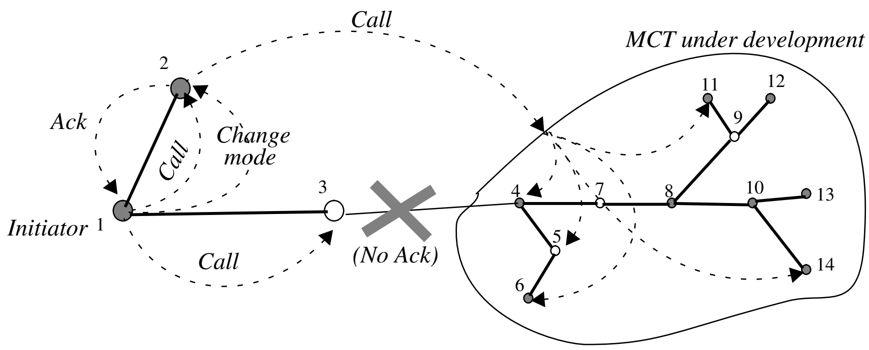

Fig.2 Changing a splitting point in the MCT

new splitting point could be established to any router mentioned in the contract as a d-node or as an s-node (network node which belongs to the unicasts of the constructed MCT).

2.3 Splitting Point Router Behaviour

The behaviour of the splitting point router processing the MCT development with the use of the multicast agent is presented in Figure 3 with the use of the timed coloured Petri Net.

The timed coloured Petri net is a place - transition net with coloured tokens representing some attribute[s] of the modelled information flow. The time delays and some data structures are associated with the positions; while transitions represent some rules for the tokens' attributes processing. TCPN are used here because of the two advantages: their expressive richness and ability for analytical study under some structural requirements [6, 7].

In Fig. 3 the following positions are used. Position `contr` represents the initial contract (transported within the so called *embryonic MCA* across the network) which initializes the current router with the following contract structural parameters regarding its location within the network (current fanout):

- set of the end receiving routers *{r}*, $|\{r\}| = R$ (each being the MCT leaf d-node) attached to the current node, placed in position `R`;
- set of the advertised routers *{a}*, $|\{a\}| = A$, placed in position `A`;
- set of the splitting routers *{s}*, $|\{s\}| = S$, placed in position `S`;
- information about advertised routers which could be used as alternative splitting nodes, placed in a position `A/S`.

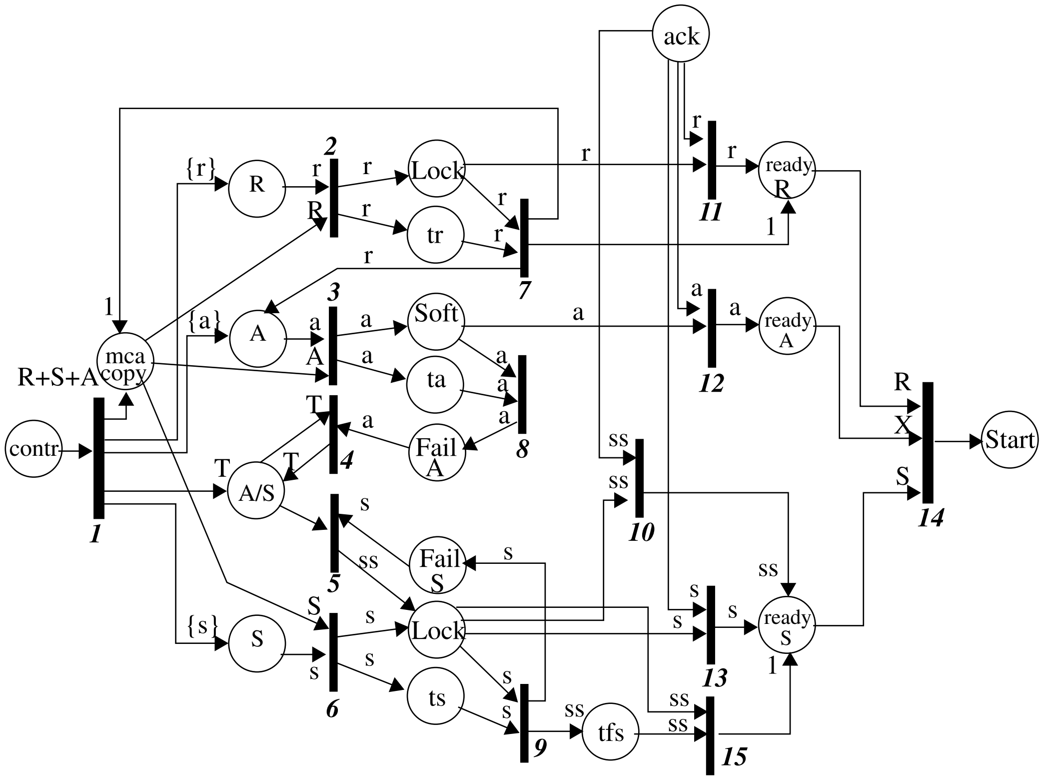

Fig. 3 The splitting point router behaviour model

The latter relation between splitting nodes and their possible alternatives is given with the table `T`, the same symbol is used to mark corresponding arcs in the Petri net. Each splitting node of the fanout (if any) may have from zero to several alternatives in this table. The case of several overlapping alternatives is not considered in this paper, however all existing alternatives are assumed to be prioritized during the MCT construction, which function belongs to the service provider. Simultaneously with the placement of needed sets (`{r}`, `{a}`, and `{s}`) of adjacent node's identifiers in corresponding positions, the needed amount (`R+S+A`) of MCA copies is prepared and placed in the position `mca copy`.

The rest of the diagram presents the fanout development concerning one splitting point. The corresponding router behaviour has three distinguishable parts: sending MCA copies and receiving some negative or positive acknowledgements from

- advertised routers (sending MCA to these routers is an optional feature, however it is considered in this paper that the best alternatives to existing splitting points could be found within advertised routers[1]);

1. This is not necessarily the case; some MCT construction algorithms may provide better alternatives within the set of end-receivers or within both sets. The splitting point router behaviour in any of these cases is easily derived from a generic model presented here.

- end receivers; and
- splitting points.

Let us consider these 3 cases.Transition ***2*** models sending a MCA copy to every end receiver (`r`); after sending this receiver is considered locked upon receiving positive acknowledgement (position `Lock`). If a positive acknowledgement is received (transition ***11***) then receiver `r` is considered ready to accept the multicast flow (position `ready R`), corresponding tokens are removed from `Lock`; if timeout `tr` expires before receiving the acknowledgement from `r` then (transition ***7***) the end receiver `r` is considered further as advertised - tokens `1` and `r` are moved respectively to `mca copy` and to `A`. That means the router `r` may join the multicast session later, however a simple token (`1`) is placed in `ready R` for proper counting of receivers.

Transitions (***3***, ***12***) and (***6***,***13***) as well as positions (`Soft`, `ta`, `ready A`) and (`Lock`, `ts`, `ready S`) represent the same successful scenario for advertised routers and splitting points respectively. However if the advertised router is silent during the timeout period `ta` (transition ***8***) then the table `T` for possible alternatives for splitting points has to be updated (transition ***4***). Consequently, if splitting point `s` fails (position `Fail S`) then, probably updated, table `T` is used to find the best alternative to which the copy MCA is to be sent, however, now with the different modality (transition ***5***). Simultaneously with the locking of this substituted splitting point `ss` a new timer `tfs` is set up to wait for this additional MCT development. Note that the MCA copy could be installed already in `ss` (however, as for the advertised router). Now this router has to elaborate the MCT further according to the MCT description sent to it as a part of the contract with modality of a splitting point. If `ss` fails to fulfil the job before the `tfs` expires (transition ***15***) then a simple token is sent to `ready S` position. The transition ***14*** represents a decision making on whether to start the multicast session or not. This decision is made on the base of any number (`X`) of ready to participate advertised routers, of `R` ready end receivers and of `S` splitting points (where some of the receivers and splitting points could be marked by simple tokens as failed).

Figure 3 shows only a sketch of the model needed for further performance analysis. A complete model has to go into the MCA implementation details.

3. The Object-Oriented Framework

3.1 An Overview of the Multicast Agent

The multicast agent is a complex software object which actually has to carry all necessary functionality to enable local negotiation, multicast flow transmission, session call establishment, etc., to all the participants of the session. Following the above mentioned MCT development scenario and [8, 9, 10, 11] let's define shortly all needed functionalities.

The *Src (sender)* method of the complex MCA object is an object with the following methods: attach/add/detach/remove media; define medium-to-medium relations (for multi-media multicasting); include/exclude applications; floor control; give current multicast state (optional).

The *SP (service provider)* method of the complex MA object is an object with the following methods (parameters): enable/disable a multicast session (contract); quality of service parameters (d-nodes list; Media list).

The *MRA (receiver)* method of the complex MCA object is an object with the following methods: filtering (Media list; QoS requirements list); include/exclude applications; attach/add/detach/remove media; patching (Medium, badly received packets); advertising (optional); floor control; participation control; give current multicast state (optional).

The *MTA (self-transmit)* method of the complex MCA object is an object with the following methods: clone (MCA); send copy (MCA); invoke (MCA_method); prune (MCA_instance);

split, i.e send copy (multicast stream).

We assume that when the SP method is activated, it starts negotiation with the service provider for a multicast session with required dedicated receivers set and required QoS parameters for a certain media involved. This contract is assumed to be assured by the service provider all over the MCT. If different receiver hosts will require changes in the contract then this is to be done via their negotiation over the MRA at the nearest splitting point.

Actually the MTA entity realizes a protocol for distributed multicast management; this particular method has to realize essential multicast functionality to achieve scalability of the sparse multicast groups. Some features of the proposed framework are emphasised below.

Support for the Multidomain Networks. When the multicast Call traverses several domains (where domain could be represented by a single service provider and/or network management) then at the border node of a new domain the MCA has to invoke it's SP method to renegotiate all needed parameters and services (media, etc.). At the same time the whole new domain is visible to the source domain as a single receiver, that's why the border node within the sender domain has to invoke the MRA method and to communicate with the Src method of the MCA copy inside a new domain.

Sender Substitution. One of the main advantages of the proposed scheme is that any network node which belongs to the MCT possesses the embryonic MCA, that is why any node can invoke needed interface, thus enabling a new active sender or joining the current MCT by a new receiver, etc. We assume that when the SP method starts negotiation with the service provider it sends a multicast session request message with the list of dedicated receivers as it's parameter; that means for the service provider that any member of the d-nodes list can become a sender. As an option we can also use another parameter, say, maximal number of simultaneously active senders.

Quality of Service. The default quality of service with regard to each media mentioned in the contract with the service provider is to be guaranteed by the service providers in each domain. However, due to different hardware equipment, applications and types of host connection to the internetwork actual receivers may have a willingness to change the default QoS parameters. This can be done by the filter interface of the MRA object for every pair (MRA, Receiving Host).

Reliability. This feature could be achieved at the network level with the use of the same technology as proposed in [21] for a single application. However, reliable integration of applications could be achieved only with application independent reliability, i.e. within the underlying transport service including multicast.

3.2. Object Framework Description

The complete set of primitive objects introduced to support the object-oriented framework for a scalable multicast is presented in Figure 4.

This set is called a framework because it is not elaborated yet in the sense of object messages, also some methods are optional and others could arise during the multicast implementation. These building blocks can support the whole lifecycle of the multicast session, including Call Establishment and Call Modification.

The Call Establishment phase needs to invoke the Src and SP objects to establish primary contract with the service provider and MTA object to clone, transmit and selfinstall the MA complex object within the nodes of the MCT. Note that we suppose to clone not an abstract MCA complex object but its instance customized according to a pre-negotiated set of session parameters (dedicated receivers list; media list; default QoS parameters; maximum number of active senders, etc.) with regard to its future network positioning. The Call Establishment phase continues with invoking of the MRA objects and their local negotiations with local receivers, session advertising, etc.

The Call Modification phase can start even during the Call Establishment. For example some of the dedicated receivers could become unreachable; crossing the domain boundary may cause changes in the default session parameters, one of possible active senders may ask for a floor even before the start of the initial flow, etc. We assume that this could not cause any problem because any MCA instance after the selfinstallation is ready to accept any of the messages which could take place within the entire multicast session. The shortcoming of this approach is that we have no opportunity to change all global parameters of the multicast session (d-nodes list, Media List and default QoS for the reserved Media). The only possibility to change these parameters is to renegotiate with the service provider via the SP object. This object could be invoked only by any of the dedicated receivers (each of which have a potential to become a sender). The only possibility to change the d-nodes list is to apply to the initial source.

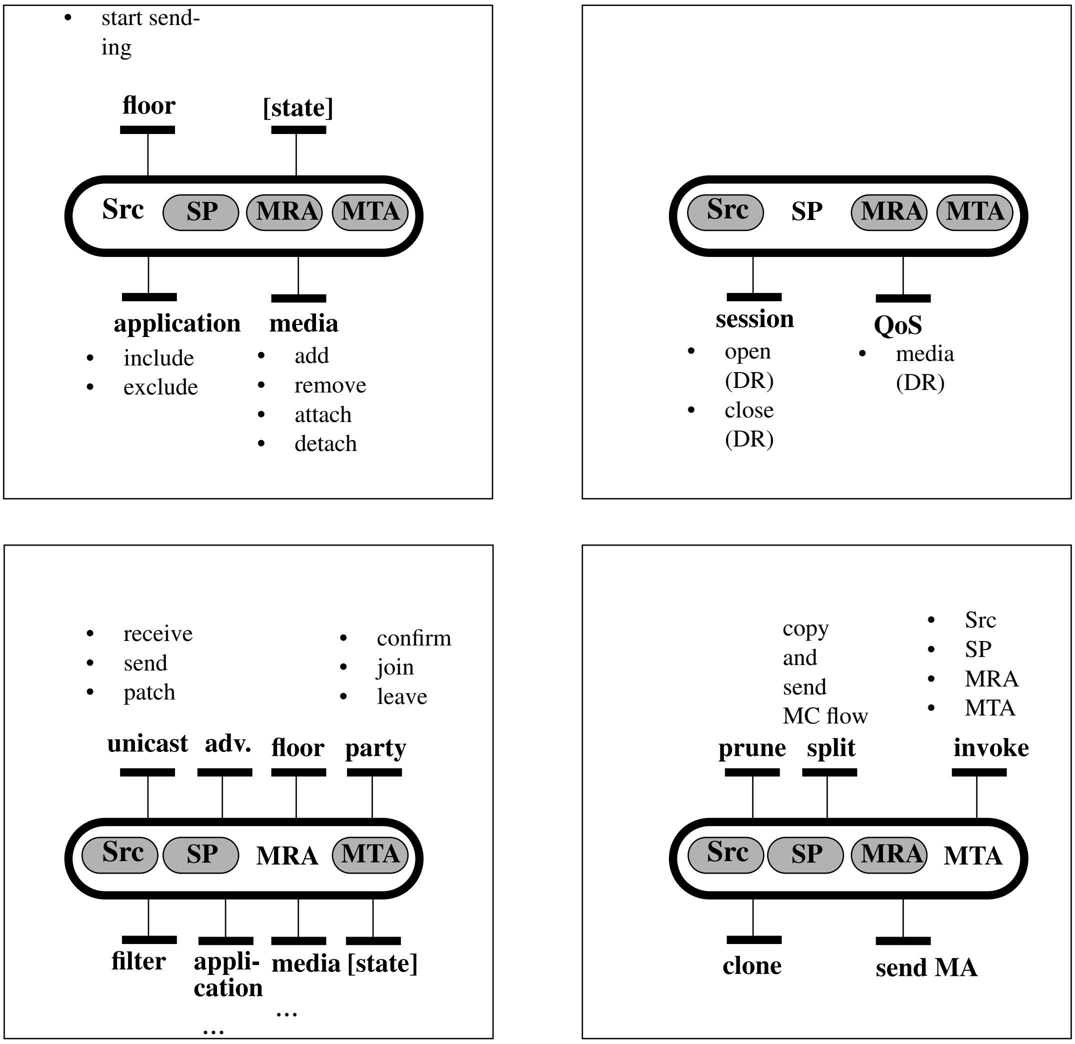

Fig. 4 The Object Framework

Note, that if the number of dedicated receivers is too large for a service provider to permit them to send in parallel, then the number of active sources is to be introduced. However this number is to be somewhat less than the actual possible number of concurrent sources because we assume no information exchange between the instances of the MCA. Consequently, the only way for them to know the number of actual active sources is to listen to the multicast. However, due to a delay introduced by the network, several sources could start sending simultaneously based on their local permission. To prevent this situation we assume simply that the MCA instance nearest to the last source has to stop sending when recognizes that actual number of active sources is greater than the default maximum.

4. Implementation Remarks and Related Work

The technical availability of the migrating multicast agent could be explained within the CORBA specification as shown in Figure 5. To solve the problem of unique object references the CORBA suggests [2] the mechanism of the interoperable object reference (IOR) externalization together with the externalization of the copy object; consequently two proxies are to be installed within the naming services of two collaborating ORBs. This paper suggests additionally that no reference is to be created for a copy MCA except the IOR. Moreover suggested locking of the destination actually locks the copy MCA as well, preventing from confusion with multiple objects within one ORB. The CORBA specification was chosen among others for the following reasons:

- CORBA implementations provide efficient interoperability across heterogeneous software platforms;
- being a software development platform it has an application viewpoint;
- CORBA deployment is rather fast in the direction for a cross-compatibility with ODP (TINA) [12]; OpenDoc, OLE [13] and CAD's Portable Common Tool Environment [14];
- adopted UNO approach [2] enables flexible distributed systems architecture.

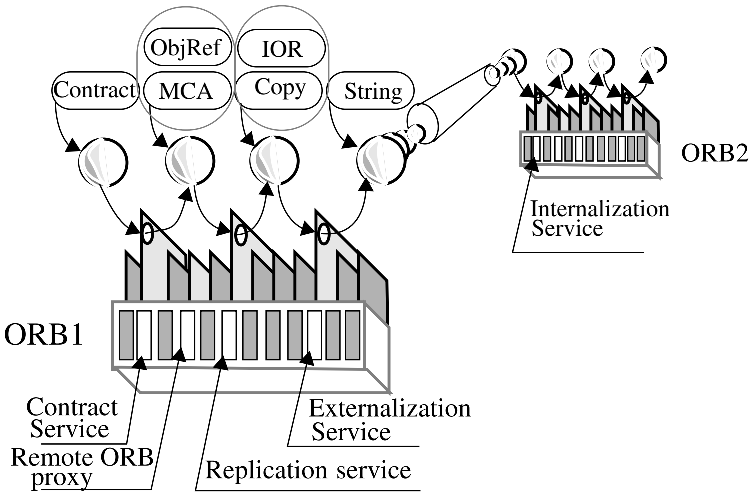

Fig. 5 MCA migration in CORBA environment

The current approach was elaborated within the STEP group of GMD FOKUS [15] as a pre-project proposal aiming at highly scalable and reliable mechanism for application integration via multicast. It is hard to position the current proposal within existing approaches, because it crosses two distinguished domains, networking and object-oriented distributed computing.

There are known successful attempts to introduce object orientation in a typical networking areas such as, most evident candidate, the call establishment [16] for B-ISDN and for multi-media multi-user applications [8]. The last work concentrates on media issues and introduces three types of objects (medium, relation, attachment) and several call establishment primitives such as Add/Remove User; Add/Attach/Detach/Remove Media, based on very general assumptions about the network support for the multi-user collaboration. However, one of the main problems for the efficient multicast remained untouched in this work - the multicast tree providing for virtual paths for the call establishment. Within the typical network domain there exist an interesting proposal for a protocol independent multicast [9], however it is unclear how to optimize the rendezvous point location within the network. This paper sets some analogous to these points to every splitting point of the MCT. The idea of network agents, travelling within the network and performing their tasks is not new, the latest complete overview of trends in the area could be found in [17], somewhat similar idea was implemented for the WWW service [18], while somewhat opposite task - migrating of user environment (teleportation) - was elaborated in [19].

Finally, it should be mentioned that a software migration itself seems to become one of the emerging areas of the software industry [20].

References

1. Kay, J., Kummerfeld, R.,J., *Customization and Delivery of Multimedia Information*, in Proc. of Multicomm'94, Vancouver, Canada, Nov.,1994
2. ORB 2.0 RFP Submission "*Universal Networked Objects*", Sept.,1994, OMG TC Document 94.9.32
3. *Object Services Architecture*. revision 8.1 OMG Document 95-1-47 (12.01.95)
4. Nilssen, G., Dupuy, F., Chapman, M., *An Overview of the Telecommunications Information Networking Architecture*, in Proc.TINA'95 "Integrating Telecommunications and Distributed Computing - from Concept to Reality", v.1,.Melbourne, Feb.,1995, pp. 1-11.
5. Smirnov M.I. *Efficient Multicast Routing in High Speed Networks.* - In: Proceedings of the First International Workshop on High Speed Networks and Open Distributed Platforms (HSN&ODPl)/Ed. V. Tschammer, M. Smirnov, June, 1995, St.Petersburg, Russia, 17 p.
6. Smirnov M.I., Iljin V.P. *Network control processes models based on timed Petri-nets with inhibitor arches*.- In: Automation and Computers, 1985, No 5 (in Russian)
7. Smirnov M.I., Iljin V.P. *Timed Petri-nets with inhibitor arches for system modelling*. - In: Electronic Modelling, 1990, vol. 12, No 2 (in Russian)
8. Heijenk X.H., Niemegeers I.G. *Communication Systems Supporting Multimedia Multi-user Applications.* - IEEE Network, January/February 1994, pp.34 - 44
9. Deering S., Estrin D., Farinacci D., Jacobson V., Liu Ch.-G., Wei L. *An Architecture for Wide-Area Multicast Routing,* In: Proc. SIGCOMM 94 -8/94, pp.126 - 135
10. Szyperski C., Ventre G. *Efficient Multicasting for Interactive Multimedia Applications*, TR-93-017, International Computer Science Institute, Berkeley.
11. Handley M., Wakeman I. *CCCP: Conference Control Channel Protocol. A scalable base for building conference control applications*. v.1.4. An html link was provided at http://www.research.att.com/cgi-bin/bibsearch_html?CCCP
12. *Comparison of the OMG and ISO/CCITT Object Models. The Report of the Joint NM Forum/OMG Task force on Object Modelling*, OMG Document 94-12-30, 27 p.
13. Adler R.M., *Emerging Standards for Component Software*. - In Computer, March, 1995, vol.28, No. 3, p. 68-77
14. *Relationship of PCTE to OMA*, Revision 1.0, OMG TC Document 93.4.7, 23.04.93
15. http://www.fokus.gmd.de/step/entry.html
16. Grebenö J., Hanberger N., Ohlman B. *Experiences from implementing an Object Oriented Call Model.* - Proc. XV International Switching Symposium, ISS'95. Vol.2, pp. 77 - 81
17. Magedanz T., *Intelligent Agents - State of the Art and Potential Application Areas in Future Telecommunications*. - In: Proceedings of the First International Workshop on High Speed networks and Open Distributed Platforms (HSN&ODPl)/Ed. V. Tschammer, M. Smirnov, June, 1995, St.Petersburg, Russia, 17 p.
18. Luotonen A., Altis K., *World-Wide Web Proxies*, in First International WWW Conference, (Geneva, Switzerland), May 1994, 8 pp.
19. Gajewska H., Kistler J., Manasse M.S., Redell D.D., *Argo: A system for distributed collaboration*. - In ACM Multimedia, ACM, 1994.
20. Start K, Patel A. *The Distribution Management of Service Software*. - Computer Standards and Interfaces, 17(1995):291-301
21. Floyd S.,Jacobson V., McCanne S., Liu Ch.-G.,Zhang L. *A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing*. - to appear in Proc. SIGCOMM, Cambridge, Mass., Sept.1995