Object Distribution Networks for world-wide document circulation

María Eva M. Lijding, Claudio E. Righetti
Computer Science Department
Universidad de Buenos Aires
{mlijding, claudio}@dc.uba.ar

Leandro Navarro Moldes
Computer Architecture Department
Universidad Politècnica de Catalunya
leandro@ac.upc.es

Abstract

This paper presents an Object Distribution System (ODS), a distributed system inspired by the ultra-large scale distribution models used in everyday life (e.g. food or newspapers distribution chains). Beyond traditional mechanisms of approaching information to readers (e.g. caching and mirroring), this system enables the publication, classification and subscription to volumes of objects (e.g. documents, events). Authors submit their contents to publication agents. Classification authorities provide classification schemes to classify objects. Readers subscribe to topics or authors, and retrieve contents from their local delivery agent (like a kiosk or library, with local copies of objects). Object distribution is an independent process where objects circulate asynchronously among distribution agents.

ODS is designed to perform specially well in an increasingly populated, widespread and complex Internet jungle, using weak consistency replication by object distribution, asynchronous replication, and local access to objects by clients. ODS is based on two independent virtual networks, one dedicated to the distribution (replication) of objects and the other to calculate optimised distribution chains to be applied by the first network.

1. Introduction

The Internet of the 70's is performing the job of transporting documents and providing useful information increasingly slowly and painfully. This is the result of a poorly understood effect of the combination of many aspects fueled by the explosive growth in user population, the WWW traffic, and the proliferation of many contents of diverse quality.

Users experience increasing latency when accessing resources[1]. Our measures on the link between the academic networks of Spain and Argentine are higher than 2000 ms, with packet loss between 60%-70% (January - February 1997). In addition, network partitions due to link or node failures are too frequent.

The effect of an increasing number of users is worsened considering their behavior. The situation when a massive number of users have a common interest at a certain moment, causes a phenomena of inundation in a resource server and their network vicinity, called flash-crowd [Nielsen 95]. Flas-crowd can be observed with resources[2] that thousands of users want to access simultaneously (e.g. 880.000 accesses to the NASA Web pages during the collision of the Levy-Shoemaker-9 collision, or access to elections results in many countries).

As a result of the above effects, many resources become unreachable to huge amounts of users, feeling frustrated by this.

---

[1] Berners-Lee arguments that a reasonable latency is around 100 ms [Berners 95], but Viles and French have found in their studies a latency of about 500 ms [Villes 95]. Packet loss is considered common at around 40% but may be higher [Golding 92b]. In addition, network partitions are not uncommon.

[2] If the resource is a document, it is usually referred as a *hot document*.

Another problem is the large volume of information available in Internet. Generally there aren't guarantees of the quality and reliability of such information, and noisy information tends to hide any useful information[3]. Information in Internet *is not classified* based on any established quality criteria (Usenet news and some Web catalogues provide some degree of classification). This brings about yet another problem: how to find relevant information for users? (Or: How to find relevant readers for a given information?) The excess of information and their associated problems lead to total misinformation. This excess of information is as problematic as the lack of it.

Classification schemes are collections of labels or topics produced by a classification authority, and used to associate meta-information to objects. They are used by authors or publishers to describe the objects they publish, by the distribution and routing network to replicate collections of documents, and by readers to select which contents they are interested in. The content, structure, scope and procedure to associate labels to objects is diverse. For instance, Usenet News provides a classification scheme for news under many topics (newsgroups), with public and moderated topics; professional organisations, the IEEE for instance, provide a catalog of topics in terms of publications with specific rules for content selection and review.

The IRTF (Internet Research Task Force) has stated that resource discovery tools (WAIS, Gopher, Web, Archie, Prospero, etc.) have scalability problems. [Danzing 94] has classified these problems in three dimensions: Data volume, number of users, and RDT diversity.

Our work is focused on providing solutions to the lack of scalability mainly on the first two dimensions. The goal is to provide local access to relevant and pre-selected information; obtaining the best service from the ordered use of global interconnections where bandwith is scarce, quality is unstable and network partitions occur too often; and providing a global and cooperative mechanism for content classification and qualification (meta-information).

We focus on a model centered on communities or organisations that are producers and consumers of information: they may produce, classify, label, offer and publish information, and also look for and consume information produced by other distant communities. These interactions occur with a local (region, organisation) service agent, while object distribution is done asynchronously, reliably and cooperatively among agents located anywhere in Internet. This model is appropriate because intra-community networking is usually adequate meanwhile external networking is usually poor and more expensive.

2. Framework: weak consistency replication

While some sites in the global Internet become inaccessible due to latency, network partitions or flash-crowd, objects in our system are accessed in the more robust local region and updated asynchronously (policies: in batch operation, in hours of low traffic, always except in failures, etc.). ODS maintains metadata about objects about their classification under labelling schemes. In addition, this metadata provides people awareness about new objects in areas of their interest at a minimal cost. ODS allows selective distribution, subscription and notification, bringing order and economy to the chaotic and redundant traffic circulating nowadays in Internet. Distribution chains are built dynamically to find the most effective way to provide end users a replica of the objects they want to obtain, while making good use of

[3] About Usenet News, [Saltz 92] considered that over 90% of the contents of newsgroups were noise. Generally speaking, a high noise to useful information is only worth a search for topics (awareness), not a systematic reading.

available and always limited resources. Readers and authors have a service specialized in selective and economic distribution of information on large scale.

Replicating resources improves performance and availability by approaching information to readers. By storing copies of shared data on processors where they are frequently accessed, the need for expensive remote read accesses is decreased. By storing copies of critical data on processors with independent failure modes, the probability that at least one copy of the data will be accessible increases. But when resources are replicated the consistency[4] of each copy must be taken into account.

Maintaining consistency and availability of data during network partitions are conflicting goals. Correctness may be guaranteed by allowing operations to take place on only one partition. This requires a reliable mechanism to detect network partitions. In very congested networks, such as Internet, latency of the network cannot be easily distinguished from a partition. This leads to a compromise between detection of failures and availability. If we want to ensure the availability of resources at all time, we should allow normal operation even in presence of partitions. By allowing this, replicas may not always be consistent and it is needed to apply some correcting mechanism once the partition has been solved [Davidson 85].

Requirements for consistency are application dependent: weak consistency may not be acceptable for a brokering service on the stock exchange, but it may be more acceptable for providing new publications to a group of researchers.

Weak consistency replication is a policy to allow replicas to diverge during network partitions, therefore each replica can continue offering service. Once the partition ceases, replicas eventually converge to a consistent state. High significant latencies must be thought of as network partitions.

We believe weak consistency protocols are required to make replication mechanisms scalable in wide area networks, in presence of failures. These protocols have already been used in a great variety of systems, due to their high availability, good scalability and design simplicity[5].

Information dissemination

Information tools deployed on the Internet may be classified as using one of the following techniques to 'disseminate' their information [Weider 94]:

- *Come get it*:

One unique original source of information. It is the most simple and used technique (e.g. FTP, Gopher, Web, etc.), based on a client-server model. It means a waste of time for the user, as a combination of access latency and excess of information (relevant and noise).

- *Send it everywhere*:

Send information to all interested members. News, Mbone [Eriksson 94] and Harvest may be considered to use this technique. They are distributed systems, where service agents cooperate to replicate information. Important details are the topology used to disseminate information and how it is built.

---

[4] Consistency is defined as all the copies of the same logical data item to agree on exactly one value [Davidson 85].

[5] E.g. Grapevine [Birrel 82], Clearinghouse [Oppen 83], Locus [Popek 85], Coda [Satya 92], GNS [Lampson 86], AFS [Satya 93], News, Refdbms [Golding 92a], Harvest [Obraczka 94], OSCAR [Downing 90].

In the Web, the most popular mechanisms to approach information to users while reducing repetitive long distance network transfers are caching and replication.

Caching

A cache is an intermediate memory, slower than processor's registers and faster than main memory or disks. Caching exploits the locality of data: time is saved every time that the cache serves data saving access to the more expensive secondary storage. Caching needs replacement policies to decide the best content for the cache, and it suffers from problems of coherency, in case of changes on the original storage.

On the Web, caching occurs in client's local main memory, in a client local disk, or in a shared cache repository, in the neighbourhood of the client: a proxy cache. All requests "to the outside world" are sent and answered by the proxy: some are found locally (someone requested it recently), some have to be forwarded to the source. All of them should be validated against the original copy.

Generally, caching saves network bandwidth and improves client access latency to documents, but caching suffers from:

* consistency problems when network partitions occur, because validation occurs synchronously with client requests,
* latency or failures on cache misses, when a document has to be synchronously retrieved from the original source.

In other words, the client suffers from any network partition on every request, and suffers from latency on cache misses. On the other side, caching does not redistribute network access: network load keeps following temporal preferences of users (rush hours).

Sophisticated caching mechanisms (e.g. geographical push caching [Gwertzman 94], demand-based dissemination [Bestavros 95], cooperative cache [Malpani 95]) have all the problems just mentioned.

Cache techniques fail to handle large volumes of information (cache replacement policy), and they also fail when information changes rapidly (cache coherency).

In addition, the cache behaviour: synchronous to client requests, imposes a heavy server load with many pending HTTP (i.e. TCP) connections for every piece of a document, for several documents, for every user of the proxy cache.

Replication (Mirroring)

The aim of replication is to increase the availability of data, and to reduce (balance) the load on document-serving machines. Users must know and use those nearby mirror sites. If used, client access latency and flash-crowd phenomena are diminished and network traffic is equalized in time.

Mirroring consists on making an exact local copy of a remote site using the FTP protocol. This is repeated periodically affecting new, changed or deleted documents.

Mirroring has several problems:

1) There is a need to maintain consistency between original server and mirrors: the majority of documents are read-only, and changes usually occur at the originating site by the author of the document.

But users usually do not trust the source of information and they doubt about the information being kept updated. They would need to know the update policy of each mirror site, and the history of changes on the documents of interest.

[Baentsch 97] proposes a partial solution: if a client-side proxy directs HTTP request to a given server, a list of replicate servers could be included in a special HTTP header on the response. That information could be used transparently on future requests, or presented to the user to let him decide.

2) Users ignore the existence of many mirror sites and they are not able to decide which is best (closest and/or less loaded). They would have to know the quality of service (and the update policy, etc.) of a mirror site before deciding which mirror (or the original) site is best.

Current efforts on protocols for resource reservation and quality of service negotiation on Internet may help on this matter.

3) Keeping mirror sites does not prevent access to a document at the original site, so most users keep accessing only to that site.

4) Replication is carried out in a manual and centralized manner for some specific large and static collections of information, because there is no criteria to determine which data will provide the most gain from replication, and there is no automatic mechanism to replicate that data.

Object Distribution

It is a model based on many distribution models used in everyday life (e.g. food distribution chains, publications). Consumers don't go to places where goods are produced (e.g. factories, author's home). Goods are purchased in the closest retail shop, where most products are on stock waiting for customers (even though sometimes goods are back-ordered). Factories produce at a near optimal pace supplying distributors and retailers. This system works because consumers trust their retail shops: shops provide fresh products at a reasonable price, probably better deal than one could try to get from the factory.

Some consumers may try to obtain the goods directly from the producers, but those requests may be rejected. Even when producers accept direct purchasing, it is not easier for costumers to buy this way than going to retail shops. (It is not the producer main bussiness, distance, volume of purchase, working hours, product presentation, customer service, etc.).

In addition, there are classification schemes (determining distribution channels) that allow consumers and producers to optimise the distribution of objects. The telephone Yellow Pages, or the universal classification scheme in libraries are examples of that.

This model is adequate for ultra large scale and it does not exist on the current Internet community, but it may be introduced over the existing networking infrastructure, without modifying protocols and standards. Their progressive introduction provides immediate advantages for their users.

- Traffic may be even more equalized, organized and automated because users may express their interest in certain documents by subscription, and documents may be labelled under one or several categories. When labelling is done under a quality criteria by some institution, that may help to improve the quality of information perceived by clients.

3. Object Distribution System (ODS)

The Object Distribution System (ODS) is formed by two independent virtual networks: an Object Distribution Network (ODN) and an Object Routing Network (ORN). ODN brings objects close to readers according their interests, and ORN builds the distribution chains that ODN needs to do his work in a near optimal way.

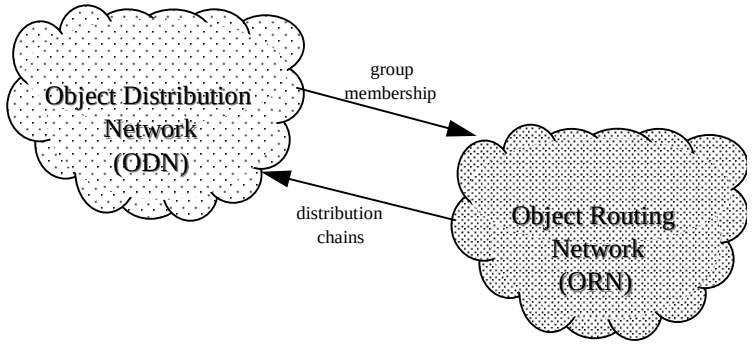

Fig. 3: Object Distribution System

ODN handles objects that are persistent and replicated in every interested service agent. ODN can handle different collections of objects, determined by their authors or some classification authority.

Our objects are write-by-one/read-by-many (e.g. Web documents, FTP archives, etc.). This means that each object is only modified in the service agent where it is *registered*. This way, the worst thing that may happen is that someone is accessing in a read-only manner a version of an object that may not be the last one. It must be noted that in a finite and not bounded time the objects will be updated (i.e. as soon as the network permits). Accessing a version of an object different from the last one, is not acceptable in certain applications. This can be clearly seen in stock exchange information for real time decisions or a videoconference. We have designed ODN for objects that do not suffer constant changes, and in unreliable networks.

An ODN is composed by a number of cooperating *service agents* that join several groups, or collections of objects, according to the interests of their users. In every ODN group service agents cooperate to obtain an efficient replication inside the group, providing a selective replication of objects restricted to interested agents only. In this way we also want to put some order in the chaos that is brought about by having information that is not classified.

ORN builds distribution chains dynamically for each group. To build the chains the *routing agents* (members of ORN) take into account the type of membership to a group of each service agent and the underlying network state. Even if systems such as News or GNS distribute objects in a hierarchical manner, they do not build distribution paths dynamically.

The routing mechanisms used in ORN for building distribution chains is completely independent of the class of objects that are being handled by ODN. Both networks were designed to work independently, defining a clear interface between them so that ORN can provide services to ODN in a transparent way.

Classification schemas

schema documents. They are identified by a unique email address.

- Classification Schemas Authors : people or organisations that prod

an unique name of the authority \ category: org.yy\aa.dd.c

assigned to documents by the author or by classification authorities. They

documents in order to classify them. According to the working scenar

- Classification Schema: an special kind of document used to classify o

order to fulfill more optimally users readness, as it provides a framew

selective distribution. A classification schema has labels or categories th

author\unique_hash: xx@yy.org\zz\ff.ps

of files, a postcript file, a Latex document, etc. Documents ar

- Document: a file or a collection of files containing multimedia text. Exam

xx@yy.org

author through several user agents. Authors are identified by a unique

to allow different people to act as an author, a service agent can be acce

could be a working team. An author is restricted to work in only one serv

- Document Author: Everyone that produces documents in DDN. We consi

Following we give the complete list of object classes:

time since the last update from the document.

- Blaze [Blaze 95] states that the probability that a document changes gets

supset of them.

‚globally popular', is at most 0.2% per day, even more this amount is res

- Bestseller [Bestseller 95b] states that the amount of updates on docum

change often based on the following statistics:

persistent, can be classified and do not change often. We state that documents

documents. Even if they are not the only class of objects DDN handles

This network is a special case of ODN, where the main objects to p

4. Document Distribution Network (DDN)

Fig. 4: Protocols for Object Distribution

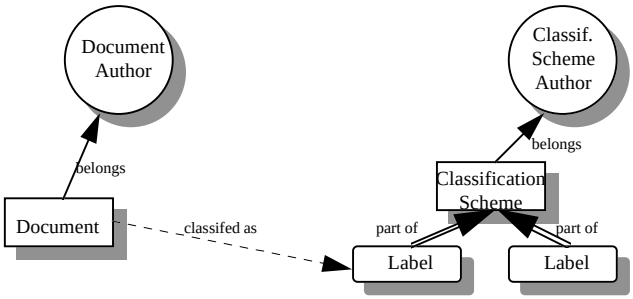

Fig. 5: Relation between DDN objects

We propose subscription to objects of DDN to be done by specifying one of the following, depending if we are interested in documents, events or classification schemes:

–Documents: with a Label; of an Author; of an Author with a Label; a given document.
–Information (events): idem as documents; about Document Authors; about Classification Scheme Authorities.
–Classification Schemes: of an Authority; a given scheme.

4.1 User Agent

Each user agent acts as an interface between users and the distribution network. It depends to the work environment and needs of users, allowing them to configure it according to their preferences. User agents need not be the same everywhere. We mention here some possible user agent implementations:

- Access (potentially off-line) to publications by a client application in a personal workstation. The user initiates a connection to new information of interest to the user. New documents or changes produced since the last connection are also submitted.
- CGI application running in an http server located at the DDN server agent node. Our reference implementation uses this.
- A process that cooperates with a proxy server [Luotonen 94] that allows HTTP requests from users to be redirected to local requests to the service agent. This only uses part of the functionalities provided by a service agent, but is transparent to users, that need not even know about the existence of DDN.

4.1 Service Agent

In addition to handle publication, subscription and cooperation with other service agents, some specific service agent can be specialized with internal functionalities that configure it's behavior in various ways. Some examples:

- [scheme] repository of collections of classification schemes;
- [meta-information] repository of meta-information about documents in general (without content), about documents in paper, etc;
- [contents] publisher (editorial) with different policies of document admission, charge and delivery.
    - They can be thematic, such as the IEEE or Prentice-Hall, with specific acceptance policies: peer review, decision of the editor, etc.
    - They can be regional, such as university or city libraries, with thematic and organisational policies: select topic of interest to the university or the city.
    - A generic publishing service provider that would charge the author and/or the reader to distribute documents.

- [contents] Processor that provides abstracts, search indexes, tables of contents, cross references, citations, digitalization of paper documents, registry of document circulation, payment agents, copyright clearance centers, etc.

Scenarios of use

Through DDN we want to prevent that the volume of available documents becomes too big for the user to handle it and to provide the user with a reference about their quality. As seen, this must be done in a way that improves access to documents and uses resources efficiently.

A reader may express their reading interests in terms of subscriptions. These may be expressed in terms of one or several classification schemes, a document selector, or a meta-information selector. Typically it would be a list of <item, mode>, where item is <name of classification authority, category>, and mode is <document or metainformation>. For instance, interest in any document from the SIGOIS of the ACM, and interes on being aware of any event about distributed systems from the ATI organisation could be expressed as follows:

```
//acm.org/sigois/; document
//x.org/distributed_systems/; metainformation
```

In the following examples, two comlete scenarios are outlined. In a literary environment, DDN would distribute books produced by writers to readers through publishers and bookstores. Interactions writers-publisher, and bookstore-reader are local (user agent to server agent), while interactions between publisher and bookshops are far (distribution among a number of service agents of the DDN). A similar model may be applied to scientific publishing in a university research environment.

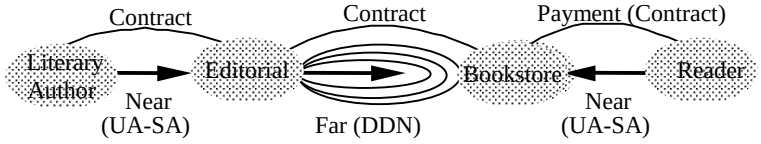

Fig 6: Literary DDN

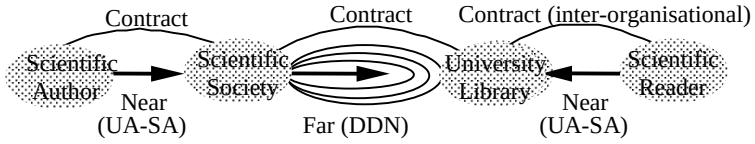

Fig. 7: Academic DDN

5. Object Distribution Network (ODN)

In general we define an Object Distribution Network (ODN) as a set of *service agents (SA)* that cooperate in order to replicate objects for users in different sites. The users need not know which is the original site of any object, or when it is reachable. Each user accesses a replica of the collections of objects he subscribes to, in the nearest service agent conected to the ODN, and also it registers there the new objects he wants to disseminate through ODN.

Users access ODN through *user agents (UA)* that work as interfaces. User agents communicate with service agent using the *User Agent Protocol (UAP)*. User agents are not part of the system, so the way users interact with them is not defined.

SA: Service Agent
UA: User Agent
ODN: Object Distribution Network
UAP: User Agent Protocol
SAP: Service Agent Protocol

Even if a distribution chain provides more than one path between two SA, objects do not cycle. Server agents reject versions of an object that are not newer than the one they have. When they are offered a newer version, they accept it, and once they receive it, they offer it to it's clients.

A distribution chain is only build with service agents from one group. Service agents only receive objects they are interested in consuming. Each consumer keeps a persistent local replica of all objects in the group and of the objects that are locally produced. In this way service agents can offer once and again the same object to their clients. They can also determine if the version of an object that is being offered to them is newer than the ones they have seen, without this meaning any burden to them. In other networks, such as Mbone, maintaining information to determine if an object was already flooded may become too expensive, so it is needed that distribution chains do not have alternate paths [Semeria 97].

Some authors propose the use of IP multicast [Deering 89] for massive replication (e.g. DPM [Donnelley 95]). Using IP multicast implies best-effort sending of datagrams to a group of hosts that share a unique IP address. This fits correctly to real time application for audio and video, where the loss of a datagram is not an important problem and receiving datagrams too late or replicated is much worse. But using directly IP multicast does not work when data consistency is needed. Message delivery is not reliable and all the members of the group must be active to receive an update, because the distribution is done directly by the producer of the object.

Multicast routing algorithms (e.g. CBT [Ballardie 93], DVMRP [Waitzman 88], MOSPF [Moy 94], PIM [Deering 96], etc.) may be studied separately from the use of IP multicast. As ours is an application level problem it is not right to depend on the network level to solve it. Our proposal is to adapt routing techniques used on network level to application level.

Networks have a physical and a logical topology. The physical topology is determined by connections between physical components. Internet's logical topology is a completely connected graph, because IP hides the physical topology to allow all the hosts to communicate.

We must find distribution chains over this logical topology. Distribution chains must allow the objects produced in the group to reach all the consumers in the group, optimizing the distribution cost.

The tags each service agent has, defines it's role in every group. When building distribution chains, each service agent is treated according to it's role. Roles are not exactly the same as tags, even if they are defined by the tags (e.g. there is no tag for mere consumers, but there is a role as consumer that implies that the service agent is either an object consumer or a meta-information consumer).

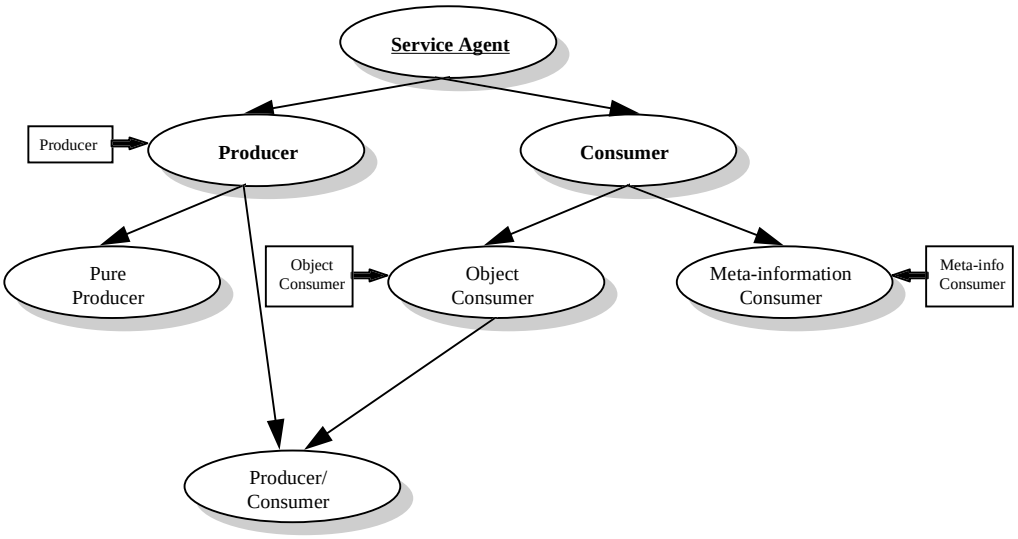

Fig. 9: Service Agent's roles

Distribution chains are build by an Object Routing Network (ORN). The way this network works is not known by the service agents. ORN is formed by routing agents that cooperate to build distribution chains that adequate to the requirements of the service agents, through a protocol that is known generically as *Object Routing Protocol (ORP)*.

Each service agent is user of a routing agent that provides it with the set of clients it must have in each group it joins. Service agents communicate with the routing agents using the *Routing Agent Protocol (RAP)*.

SA: Service Agent
RA: Routing Agent
ORN: Object Routing Network
RAP: Routing Agent Protocol
ORP: Object Routing Protocol

We can present ODN as a graph, where each service agent is a node. As we are working over a logical network that allows all nodes to communicate, the graph is complete. Each arc in the graph has a weight assigned that represents the cost of using the logical link. A possible way to determine costs is using statistic measures of the state of the link (e.g. using 'ping').

Nodes in this graph are classified according to their roles in each group. We will handle this graph as a set of graphs in different levels, one for each group. Computations will be done using this resulting graphs.

Routing agents must build distribution chains taking into account that the members of a group may change and paying special attention to the fact that the amount of them may grow significantly. On the other hand they must also consider the state of the underlying network.

We introduce here a distributed routing mechanism that may be used in a flat or hierarchical routing topology. We first introduce the basic algorithm to be used in a flat topology and afterwards the changes that must be done to adapt it to a hierarchical topology. At last we show a protocol suite to be used in a hierarchical topology.

Routing agents determine dynamically the structure of groups. Each routing agent announces it's peers the tags of the service agents it represents. With this information, each routing agent can know the members of each group. Routing agents must make computations for all the groups where at least one of the service agents they represent is involved.

Each service agent has information according to it's role in the group. In the graphs for each group, nodes (service agents) with the same role form a *layer*. We show here the information that objects of each layer have:

1)**Pure Producers:** only locally produced objects.
2)**Producers/Consumers:** objects that are locally produced, because they are producers, and all objects in the group, because they are consumers.
3)**Object Consumers:** all objects in the group.
4)**Meta-Information Consumers:** meta-information of all objects in the group.

We build the distribution chain of a group in layers. Each node has clients in the same layer or in higher layers (see fig. 7). In this way we do not allow certain distribution chains that may be optimal, but do not obey this layer division. In each layer global distribution cost is minimized.

Producers/consumers form a completely connected graph, that we call *core*. Inside the core there is a path between every pair of nodes. So each node inside the core eventually has all objects produced in the core.

Each pure producer chooses as client a node inside the core. In this way every node inside the core will have also all objects produced by the pure producers of the group. In other words every node in the core eventually has all objects in the group.

As long as there are producers in the group, there must be a core. If there are no producers/consumers, an object consumer is chosen. If there is no object consumers either, then a meta-information consumer is chosen.

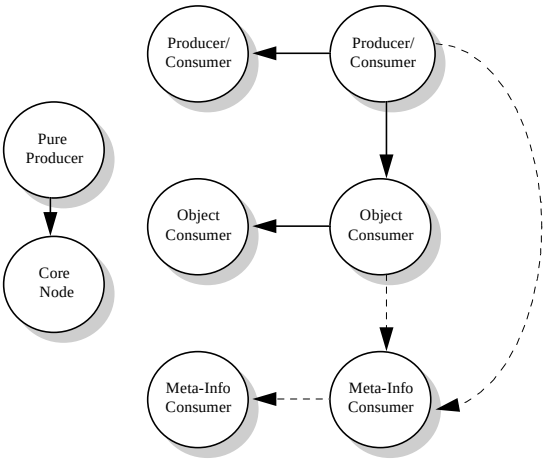

Fig. 11: Communication between layers

```
Define a set S of nodes included in
the tree
S = {1} /* original version */ (#)
Define a set LIST, that will have the
arcs chosen for the tree
LIST = ∅
While |S| ≤ n
 Choose arc (i, j) of minimum cost in
 the cut [S, Sc] (i ∈ S y j ∈ Sc)
 Add j to S
 Add (i, j) to LIST
End While
```

Fig 13: Prim's Algorithm

The service agents that are represented by the nodes of the producers tree must consider clients all their adjacent nodes in the tree. Each pair of adjacent nodes in this tree see each other as supplier and client.

Routing agents that represent object consumers compute a minimum spanning tree where all object consumers are included. We call this tree, *consumers tree*. It is build using Prim's algorithm with S={producers/consumers} (see (#) in fig. 9). In a similar way, routing agents that represent meta-information consumers must compute another tree, called *meta-information tree*, that includes all nodes interested in consuming meta-information. To build this tree we use S={all nodes that consume objects (producers/consumers and object consumers)} (see (#) in fig. 9). See fig. 8 that illustrates a hole distribution chain.

When a node is a pure producer and a meta-information consumer, it's routing agent must carry out computations as if it should be for two different nodes. The node will appear twice in the distribution chain, sending objects to the core and receiving meta-information through the meta-information tree.

3.2.2 Hierarchical Routing

In a hierarchical topology nodes are divided into domains. In each domain an *intra-domain routing protocol* is used, generally called *internal routing protocol*. This protocol may be different in each domain. To build paths between domains, an *inter-domain routing protocol* (or *external routing protocol*) is used.

Service agents should not even note that the routing topology is hierarchical, because the way distribution chains are build must be transparent to them. Something similar happens in Internet with IP that hides all details about division in autonomous systems.

We can also view domains as nodes of a graph. This graph is also complete. It's arcs are assigned weights in a static way. Arcs represent logical links that communicate two domains. As all nodes inside a domain are able to communicate with all nodes in other domains, the weight of an arc can be taken as an estimation of the average behavior of the logical links between all nodes in the two domain the arc joins.

Each domain has it's own manager that handles subscriptions of the routing agents of the domain and an entity, that we call *external router*, in charge of inter-domain routing. External routers summarize the tags from the domain nodes and assign this summary to the domain as a set of tags. The tags from the domain are informed to the peer external routers. External routers also flood their peers with the external link states of their domains.

Using this information each external router computes the algorithm we proposed in each group with local members. The algorithm is computed over the graph that represents domains to obtain inter-domain routing chains. As a result each external router knows which domains must be considered clients of it's domain, but service agents do not know anything about domains and need information about service agents.

Each domain must provide a node from it's core for each group with local consumers. This is only needed for the groups that have an inter-domain routing chain. A core must be formed even if there is no intra-domain distribution chain, because of lack of producers in the domain. The node chosen to represent the core externally is the one with the smallest identifier.

When there are no nodes that consume objects in the domain (e.g. there are only pure producers), there is no core with objects. There may be a core with meta-information if the domain has meta-information consumers, but a node from this core is not useful to supply other domains with the objects produced in the domain. In this case each producer in the domain must take as clients the selected nodes from the client domains.

The routing agent from the chosen service agent, provides information about this SA to the external router. The external router forwards this information to the external routers of the supplier domains. This external routers finally pass this information to the routing agents of the service agents that will act as suppliers (a node from the core or the pure producers from the domain).

Prototyping

We have developed a prototype where all the protocols are being tested. The topology for distribution is based on a quasi-static schema, instead of using a dynamic routing protocol.

Agent administrators configure manually the distribution chain. The prototype then determines automatically the kind of documents to exchange among agents depending on client subscriptions. Currently, the lack of a proper ORN implies that there is no guarantee that documents reach all nodes that are interested in them, if the administrators did not define the appropriate agent interconnection.

The network behaves in a more co-operative manner, as each agent accepts documents, even though it doesn't have clients interested in them. This fact can lead to whole chains of agents consuming documents they are not interested in. Meta-information is handled in a similar way, propagating the information through the chains of producers.

This prototype was developed before defining the routing protocol and it' s poor routing behaviour was an incentive to carry on with our work on the routing protocol.

It provides a user interface through WWW with the following services: document publishing, information about documents, meta-information, agent administration, local document administration done by their authors, and links to other agents and documents published in the network.

The prototype was written in C over Linux or Solaris. Each node is formed by a set of concurrent tasks. All the tables are handled through a database manager and communications are carried out using TCP streams.

We are testing the prototype between nodes in the National Industrial Technology Institute (INTI) in Argentine, the Universitat Politecnica de Catalunya (UPC) in Spain and the Universidad de Buenos Aires (UBA) in Argentine. Even if the communication from Europe to South America is not very reliable and the delay is very big, the system shows robustness and stability for document replication, and it provides to users reliable and fast access to local copies of documents.

7. Conclusions

We have presented a weak consistent replication system for Object Distribution (ODS). It uses the currently available network resources in Internet. An arbitrary number of independent

document distribution networks may be established using the ODS with different objects, participants, classification schemes, etc.

Therefore ODS or DDS increases the capacity to access information for a local community: the capacity to select and obtain documents globally; and the capacity to announce, index, publish and distribute globally documents produced locally. At the same time, external network resources are used more efficiently for the transport of meta-information and objects asynchronously among distribution agents.

This alleviates the indiscriminate use of international links as if they were local connections, avoiding the bottleneck produced by the difference of bandwidth and reliability between local and external networks, and redistributing traffic out from peak hours.

Classification authorities and schemes provide very useful meta-information to objects. This allows to do selective distribution, and it enables service agents to provide many value-added services (indexing, ToC databases, circulation statistics, etc.) that are currently found in the current paper-based document distribution system.

The identification of two independent virtual networks in ODS: distribution (ODN) and routing (ORN) allow us to continue developing each network separately. We also wish to generalize the routing mechanisms and the routing service provided by ORN for other applications.

Comparing ODN with FTP and HTTP, we found out that some documents could never be transferred between Argentine and Spain either using FTP or HTTP, due to the latency and high packet loss, but were successfully replicated using ODN. This documents were not as big as may be thought, just over 1Mb, but FTP and HTTP close the connection before completing the transfer. We must add that the reference model we are now using is just a prototype and is being improved to work in a more resilient and efficient way.

We are now studying how to incorporate support for copyrights, signatures, certification, payment and distribution of registers (e.g. number of users that access a replicated object, amount of replicas, etc.); the reference implementation is further refined and ported to other platforms; and many cooperative aspects related to publication, awareness and collaborative classification of documents are also being studied.

References

[Baentsch 97] M. Baentsch et al. “Enhancing the Web’s Infrastructure: from Caching to Replication”, IEEE Internet Computing, Vol 1, #2, 1997.

[Ballardie 93] Ballardie, A.J.; Francis, P.F.; Crowcroft, J. “Core Based Trees (CBT)”. Proc. of the ACM SIGCOMM ‘93. San Francisco, CA. August 93.

[Berners 95] Berners-Lee, Tim. “Propagation, Replication and Caching”. Web Consortium. MIT. December 1995.
http://www.w3.org/member/WWW/Propagation/Activity.html

[Bestavros 95] Bestavros, Azer. “Demand-based dissemination for Distribute Multimedia Aplication”. Proceedings pf the ACM/ISMM/IASTED International Conference on Distributed Multimedia Systems and Aplications. Stanford, CA. August 1995 .

[Birell 82] Birell, Andrew; Levin, Roy; Needham, Roger; Schoroeder, Michael. “Grapevine: An Exercise in Distributed Computing”. Communication of ACM. Vol 25-num 4. April 1982. Pp 260-274.

[Blaze 93] Blaze, Matthew. “Caching in Large Scale Distributed File Systems”. PhD Thesis, Technical Report 397-92. Princeton University, Dept. of Computer Science. January 1993.

[Bowman 94] Bowman, M.; Danzing, Peter; Mander, Udi; Schwartz, Michael F. “The Harvest Information Discovery and Acces System”. Computer Networks and ISDN Systems. Vol 28. December 1995. Pp 119-125

[Danzing 94] Bowman, M.; Danzing. Peter; Mander Udi, Schwartz Michael F. “Scalable Internet Resource Discovery : Research Problems and Approaches”. Comm. of the ACM. Vol 37 - num 8. August 1994. Pp 98-107.

[Davidson 85] Davidson, S.; García-Molina, H.; Skeen, D. "Consistency in Partitioned Networks". ACM Computing Surveys 1985. Vol 17 - num 3. September 85. Pp 341-370.
[Deering 96] Deering, S.; Estrin, D.; Farinacci, D.; Jacobson, V., Liu, C., Wei, L. "Protocol Independent Multicast (PIM): Motivation and Architecture". <draft-ietf-idmr-pim-arch-04.ps>. September 1996.
[Deutsch 94] Deutsch, P.; Emtage, A. "Publishing Information on the Internet with Anonymous FTP". <draft-ietf-iiir-publishing-01.txt>. May 1994.
[Donnelley 95] Donnelley, James
"WWW media distribution via hopwise realiable multicast". Computer Networks and ISDN Systems. Vol 27-num 6. April 1995. Pp 781-788.
[Downing 90] Downing, A.R.; Greenberg, I.B.; Peha, J.M. "OSCAR: A systems for weak-consistency replication". Proceedings Workshop on the Management of Replicated Data. Houston Texas. November 1990. Pp 26-30
[Eriksson 94] Eriksson, Hans. "MBONE: The Multicast Backbone". Comm. f the ACM. Vol 37 - num 8. August 1994.
[Golding 92a] Golding, Richard. "Weak Consistency group communications and memberships". Ph.D. Thesis. University of California, Santa Cruz. December 1992.
ftp://ftp.cse.ucsc.edu/pub/ucsc-crl -92-52.ps.Z
[Golding 92b] Golding, Richard. "End-to-end performance prediction for the internet". Technical Report UCSC-CRL-92-26. University of California, Santa Cruz. June 1992.
URL:ftp://ftp.cse.ucsc.edu/pub/ucsc-crl -92-26.ps.Z
[Guyton 95] Guyton, J.; Schwartz, M. "Locating Nearby Copies of Replicated Internet Servers". Technical Report CU-CS-762-95. Dep. Computer Science, Univ. of Colorado - Boulder. February 1995.
[Gwertzman 94] Gwertzman, James; Seltzer, Margo. "The case for geographical push-caching". HotOZ Conference. 1994.
ftp://das-ftp.harvard.edu/techreports/tr-3494.ps.gz
[Lampson 86] Lampson, B. W. "Dessigning a Global Name Service". Proceedings of the Fifth ACM Annual Symposium on Principles of Distributed Computing. Calgary, Canada. August 1986. Pp 1-10.
[Luotonen 94] Luotonen, A.; Altis, K. "World Wide Web Proxies". 1st. International Conference on the World Wide Web. May 1994.
[Malpani 95] Malpani, Radhika; Lorch, Jacob; Berger, David. "Making World Wide Web Caching Servers Cooperate". 5th. International Conference on the World Wide Web. December 1996.
[Mc Quillan 80] Mc Quillan, John; Richer, Ira; Rosen, Eric C. "The New Routing Algorithm for the ARPANET". IEEE Transaction on Communications. Vol 28-num 5. May 1980.
[Moy 94] Moy, John. "Multicast Extensions to OSPF". RFC 1548. March 1994.
[Nielsen 95] Nielsen, J. "Multimedia and Hypertext". Academic Press San Diego. 1995.
[Obraczka 94] Obraczka, Katia. "Massively replicating services in wide area internetworks". Ph.D. Thesis. University of Southern California. Deciembre 1994.
[Oppen 83] Oppen, D. C.; Dalal, Y. K. "The Clearinghouse: A decentralized agent for locating named objects in a distributed enviroment". ACM Transactions on Office Information Sytems. Vol 1 - num 3. July 1983. Pp 230-253.
[Popek 85] Popek; Walker; Kiser; English; Matthews; Butterfield; Thiel . "The LOCUS Distributed System Architecture". The MIT Press Cambridge London England. 1985.
[Satya 92] Satyanarayanan, M.; Kistler, James. "Disconnected Operation in the CODA file system". ACM Transactions on Computer Systems. Vol 10 - num 1. February 1992. Pp 3-25.
[Semeria 97] Semeria, C.; Maufer, T. "Introduction to IP Multicast Routing". <draft-ietf-mboned-intro-multicast-00.txt>. 3Com Corporation. January 1997.
[Villes 95] Viles, C.L.; French, J.C. "Availability and Latency of World Wide Web Information Servers". Computing Systems. Vol 1. 1995. Pp 61-91.
[Waitzman 88] Waitzman, D.; Partridge, C.; Deering, S. "Distance Vector Multicast Routing Protocol". RFC 1075. November 1988.