EAI Endorsed Transactions

on Industrial Networks And Intelligent Systems Research Article ICST.ORG

GRAPP&S, a Peer-to-Peer Middleware for Interlinking and Sharing Educational Resources

Thierno Ahmadou Diallo[1,2,*], Olivier Flauzac[2], Luiz-Angelo Steffenel[2], Samba N'Diaye[1]

[1]Département d'Informatique, LMI, Université Cheikh Anta Diop, 5005 Dakar-Fann, Sénégal
[2]CReSTIC Laboratory - SYSCOM team, University of Reims Champagne-Ardenne, Reims, France

Abstract

This article presents GRAPP&S (Grid APPlication & Services), a specification of an E-learning architecture for the decentralized sharing of educational resources. By dealing with different resources such as files, data streams (video, audio, VoIP), queries on databases but also access to remote services (web services on a server, on a cloud, etc.), GRAPP&S groups the resources of each institution in the form of a community and allows sharing among different communities. Educational resources are managed on a transparent manner through proxies specific to each type of resources. The transparency provided by proxies concerns the location of sources of educational data, the processing of queries, the composition of the results and the management of educational data consistency. Furthermore, the architecture of GRAPP&S has been designed to allow security policies for data protection, both within a community and between different communities.

Keywords: E-learning, Peer-to-Peer, Prefix routing, Data proxies

Received on 22 January 2014, accepted on 18 November 2014, published on 08 May 2015

Copyright © 2015 Thierno Ahmadou Diallo et al., licensed to ICST. This is an open access article distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/3.0/), which permits unlimited use, distribution and reproduction in any medium so long as the original work is properly cited.

doi: 10.4108/inis.2.3.e1

1. Introduction

The pedagogical integration of information and communication technologies (ICTs) to all education degrees is promising way to improve the quality of African education systems. Online learning (E-learning) and mobile learning (M-learning) help not only to strength the planning and the management of a democratic and transparent education, but also to extend the access to learning, to improve quality and ensure inclusion.

Thanks to the opportunities they offer in terms of use or adaptation, in particular in environments where there are insufficient resources, free access educational resources constitute an excellent opportunity to achieve the goal of a education of quality for everyone. This is the first motivation of the project GRAPP&S. The objective of this project is to construct an E-learning architecture for sharing and decentralized management of all educational resources formats like files, streams (video,audio, VoIP) and resources from web services, cloud and distributed computing services. These resources are transparently presented to the user (student, teacher) thanks to the use of proxies adapters tailored for each educational resources.

Nowadays, it is very easy to share and learn using Internet. The Internet has contributed greatly to the education system by introducing the concept of E-learning. The latter is now accepted in the various educational institutions to improve the learning process for students and teachers or administrators. An important characteristic of E-learning systems is the sharing and use of educational resources.

Although most of E-learning repositories give free access to their repositories of educational resources, the integration process is still costly [5] as most learning repositories [6, 19] rely on different standards to access the resources [4]. Furthermore, several systems rely nowadays on storage facilities on the cloud, which poses a problem to remote localities due to the access speed.

Therefore, an intuitive approach is to regroup different local learning repositories from each institutes (schools, universities, repositories of research laboratories, etc.) in a "Community", which can foster the aims of reusing and sharing educational resources without costly duplicating them into local learning repositories. Through the use of communities, we can promote the goals of reuse and sharing educational resources. By extending the coverage to different educational resources (files, videos, data in a database, even a video stream for example when subscribing to TV channel for learning languages), we can integrate all tools to improve education in a single infrastructure, with a smaller cost. Following this approach, two major research challenges must be considered to ensure interoperability across the Web:

*Corresponding author. Email: thierno.diallo.sn@gmail.com

1. Compatibility between systems: if today most APIs (Dropbox, Google Drive, etc.) allow to handle simple files, it is less clear how to integrate complex data such as queries on a database, data streams or web services.

2. Decentralized data management: most platforms are migrating to the cloud, but at the expense of losing its proximity to the consumers, as well as poor access speed and privacy threats.

From these elements, we present GRAPP&S, an architecture designed to connect institutes interested on sharing educational data sources through the network. GRAPP&S brings therefore:

- a decentralized solution for sharing all types of educational resources, not only files (text/xml) but also databases, streams (video, audio), and resources from remote services such as web services and distributed computing, all integrated through the use of data proxies;

- a simpler way to connect a large number of institutes interested into resource sharing. This is obtained through the aggregation of resources from each educational institute in the form of a "community". This will allow quick access to resources due to the proximity to consumers, unlike most of the E-learning solutions based on cloud storage when consumers are penalized because of the slow access speed;

- an access to resources GRAPP&S independent from direct interconnection between nodes. In GRAPP&S, it is always possible to route data transfer by the inverse of the path used during the search.

- the possibility to define security policies for the protection of educational resources within a community, and access policies between different communities through the establishment of *Service-Level Agreements (SLAs)*;

The remaining sections of the paper cover: Section 2 discusses the related work. Section 3 presents an overview of GRAPP&S architecture and describes the elements of this architecture, while Section 4 describes the lookup algorithm used to locate resources inside GRAPP&S. Finally, Section 5 concludes this paper.

2. Related Work

2.1. E-learning Systems

Because GRAPP&S aims at the decentralized management of resources, it seems natural to identify peer-to-peer (P2P) works in this area. Indeed, an architecture for sharing educational resources among different learning institutions is proposed in [4]. This architecture, called LOP2P, aims at helping different educational institutions to create course material by using shared educational resource repositories. Nonetheless, resources of different formats cannot be easily integrated in this platform. Similarly, [9] develops a P2P based E-learning system that uses the video for learning. This system divides multimedia data into fragments managed by assigned agents, and this system allows the sharing of multimedia data, but it cannot deal with other data types.

Several E-learning systems based on cloud are being proposed, like [12] or [2]. In [12], an E-learning ecosystem based on cloud computing and Web 2.0 technologies is presented, and the article analyses the services provided by public cloud computing environments such as Google App Engine, Amazon Elastic Compute Cloud (EC2) or Windows Azure. It also highlights the advantages of deploying E-Learning 2.0 applications for such an infrastructures, and identify the benefits of cloud-based E-Learning 2.0 applications (scalability, feasibility, or availability) and underlined the enhancements regarding the cost and risk management. In addition, [2] is used to run web 2.0 applications, such as video teleconferencing, voice over IP, and remote management, over handheld devices and terminals. As it is targeted towards military usage, the work from [2] has a multi-level security and the network infrastructure is encrypted.

It is quite evident that the cloud-based system would help the educational institutes or universities to share and disseminate knowledge among students, teachers and researchers, but the use of Cloud Computing in the educational system presents many risks and limitations: not all applications run in cloud, there are risks related to data protection, security and accounts management. Also, the access speed to cloud infrastructures may be a critical factor, amplified by the lack of a stable Internet connection that may affect the work methods in some isolated areas.

2.2. Routing in Peer to Peer Systems

Resources in a P2P system are distributed among all peers. These resources must first be searched and located in order to access it. These resources are identified by a key and depending on how a P2P system searches for resources using these keys, and by the relationship between the key and the peer, the type of P2P system is defined differently.

In unstructured P2P Gnutella [8] example uses a method of flood routing queries rather than to a specific node. To return a response to a search query, Gnutella allows direct response back to the requesting node if possible (firewall could prevent such an item). Otherwise, the answer may come back

along the path traversed by the query. However, because of the flood search algorithm is limited to the scale. Systems like KaZaA [15], Gnutella [11] have implemented a partial unstructured P2P, where supernodes concentrate metadata about the surrounding nodes and therefore can be asked before a flooding research, minimize the impact on the network.

Structured P2P networks Chord [16], Pastry [14] are based on DHT (*Distributed Hash Table*). Resource lookup works with a a peer p sending a request for a key k, p to one of its neighbors with the ID closest to k. Each peer receiving the request to repeat the same operation until the peer responsible for k is reached by the query.

All structured P2P network and unstructured are all horizontal design without hierarchical routing. All peers are identical in the sense that all peers use the same rules to determine the routing of a query. This approach is very different from that of the Internet, where the hierarchical routing is used. Hence, Kleinrock *et al.* [10] and *Peng et al.* [13] proposed the hierarchical routing to reduce the routing information in large scale networks. This technique consists on each node keepin a complete routing information for a set of closest nodes.

Bakker *et al.* [1] introduced the routing prefixes to minimize the routing information in dynamic networks. The idea of routing prefix is to assign each node n a label $\alpha(n)$, which is a word over an alphabet Σ containing at least two symbols. Each link is also labeled with a word Σ^* is a prefix of some labels of nodes. A message to then node n is transmitted on the link whose label is the longest prefix $\alpha(n)$. These authors demonstrated that any network admits a routing prefix and propose a method of constructing functions routing prefixes.

3. The GRAPP&S Architecture

The GRAPP&S framework is an E-learning solution for the decentralized sharing all types of resources. As stated before, our main objective is to propose a resource sharing middleware to support E-learning tools and applications that allies both flexibility, security and performance on remote locations. We observe indeed that current sharing tools tend to externalize data (on a dedicated server or in the cloud), but this solution is not adapted to remote locations whose computational resources are limited and the access speeds are reduced. To circumvent these limitations we to propose an alternative architecture inspired on the grid environments, where individual resources are gathered and shared among the grid members.

By relying on the GRAPP&S framework, we allow pooling of data from each institution in the form of community, and allow different communities to share resources based on predefined access rules. For this reason, GRAPP&S can also be extended to other areas of the school, by creating a community in the administrative part, separated from the education part, with safety rules and resource protection. GRAPP&S also allows a flexible deployment of its components, adapting therefore to the capacity of the computing nodes. Finally, the data proxy abstraction that composes the framework leafs (see the Data Manager component below) allow the integration of different data sources, making therefore GRAPP&S an unified resource sharing platform.

In the following sections, we present the different elements of our framework GRAPP&S for the decentralized sharing of educational resources.

3.1. Model

We consider a model of communication represented by an undirected and connected graph $G = (V, E)$, where V denotes the set of nodes in the system and E denotes the set of communication links between nodes [3]. Two nodes u and v are said to be adjacent or neighbors if and only if u, v is a communication link of G. $u_i, v_j \in E$ is a bidirectional channel connected to port i for u and to port j for v. Thus nodes u and v can mutually send and receive messages. Nodes communicate by using asynchronous messages.

A message m in transit is denoted $m(id(u), m', id(v))$ where $id(u)$ is the identifier of the node that sends the message, $id(v)$ is the identifier of the receiving node and, m' the message content. Each node u of the system has a unique id and has two primitives: **send(message)** and **receive(message)**.

3.2. Nodes of GRAPP&S

In order to present our architecture, we introduce some notations first. A community (C_i) is an autonomous entity, which includes educative resources sharing some properties: same location (resources institute, university, research laboratory), same administration authority, or same application domain (administrative resources vs teaching resources). A community contains one communicator process noted (c) and at least one *Resource Manager* process noted *RM* and one process *Data Manager* noted *DM* and these processes are hierarchically organized in the Community.

Communicator(c). nodes play an essential role that is related to information transmission and interconnection between different communities, such as when passing messages through firewalls. A Communicator is the community entry point and assures its security towards the outside, through establishment of *Service-Level Agreements (SLAs)* with other communities. The communicator also defines the security rules (access) for the protection of educational resources inside the

community (for example the administration community can see the data on the educational system, but not the reverse, thanks to this access rules defined by the communicator).

Resource Manager(RM). processes ensure indexing and organization of educational resources in the community. The RM_i processes are involved in the search and indexing of data in the community c_i, and by receiving queries from its neighbors communities. Given the important role of RM processes in research and indexing of resources, we choose a RM among ordinary nodes that have good performance levels in both CPU, memory size and communication speed.

Data Manager(DM). processes interact with sources of educational data such as databases, file, email servers, WebDAV servers, FTP servers, disks, or cloud services. A DM node is a service that has the following components (see Figure 1):

- a proxy interface adapted to the various formats of educational data,
- a query manager that allows to express queries on local or global educational resources, and
- a communications manager that allows the DM node to communicate with the RM node to which it is connected.

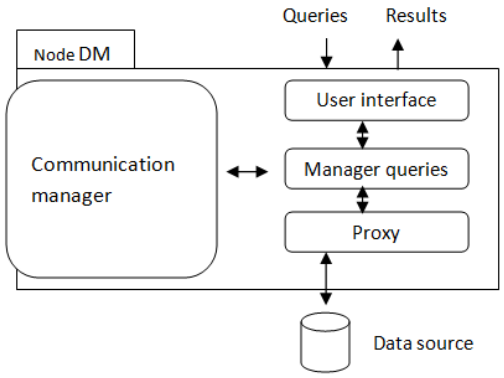

Figure 1. DM node architecture

3.3. Management of the Community

GRAPP&S can be deployed in several ways, depending on the placement of the nodes. For example, we can find the following deployment topologies;

1. nodes can be grouped into a single physical machine (see Figure 2a). This is an example of a machine of a teacher who wants to host a resource server and set different access rights to the users. With this organization, users from other communities (see Figure 4) depend on inter-community SLAs to access the server data. An example of inter-community lookup and transfer is presented later in the next sections.

2. the nodes are organized in a server farm such as a cluster, which is characteristic of an HPC network or a classroom (Figure 2b). In this case, different nodes belong to the community and present different behaviors. Some of the nodes may act as data servers, while others act as clients. As in the previous item, inter-community SLAs can be defined to give access to nodes from other communities.

3. nodes can be distributed over different machines, which corresponds to a grouping of educational resources in a university or a school with remote sites. These resources share the same administration entity (see Figure 2c). This allows a better scaling of the resources and integration of several data sources, as well as clients.

These scenarios correspond to typical environments found on educational organizations. The nodes arrangement will depend on several factors such as storage capacity and computing power of the devices, as well as the security policies that must be implemented. While special access rights can always be implemented in a per-DataManager basis, we consider that nodes inside a community are considered to have full access to its resources, and that inter-community SLAs implement the access policies.

3.4. Hierarchical Addressing Nodes on GRAPP&S

In order to correctly identify each resource inside a GRAPP&S community (and between different communities), each node on GRAPP&S needs its own unique identifier (ID). The IP or the MAC addresses are not sufficiently accurate because they do not identify uniquely different nodes that can reside on a same machine (eg RM and DM). Thus, we rely on the identification method proposed by JXTA [18], which uses a string of 128 bits. Each node has a unique string $ID-local$, in the form $"urn:name-community:uuid:string-of-bit"$. As GRAPP&S is hierarchically structured, the expression of hierarchical addressing is done by concatenating the IDs as a prefix, i.e., ID c_i node is equivalent to its $ID-local$, the node ID is formed by RM_i $ID\text{-}c_i/ID\text{-}RM_i$, and DM_i node ID has the form $ID\text{-}c_i/ID\text{-}RM_i/ID\text{-}DM_i$.

An advantage of using an addressing model specific to GRAPP&S is that it is independent of the overlay network addressing model that is implemented. Indeed, Figure 3 presents an illustration of GRAPP&S implemented over the Pastry P2P middleware. Thus,

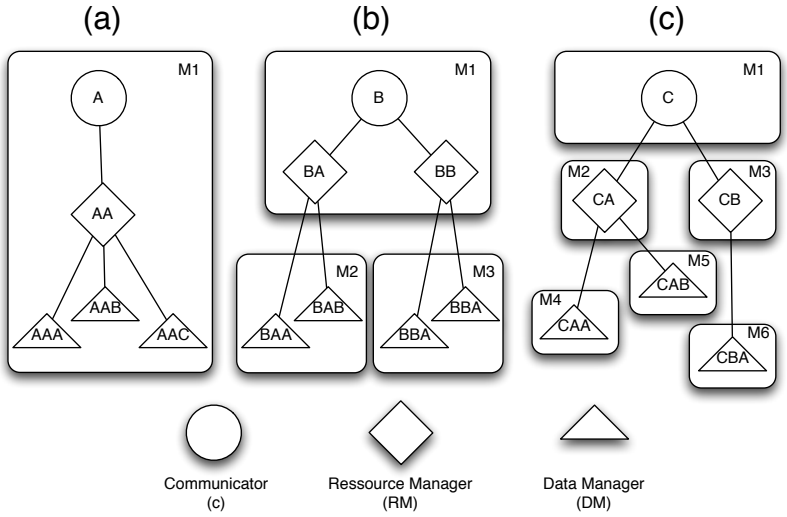

Figure 2. Organization of the nodes in a machine (a), in a cluster (b) and in a network (c)

two communities GRAPP&S implemented on different middleware can still be compatible, once the connection is established between their communicators.

The other advantage of hierarchical addressing scheme is to allow easy up structure GRAPP&S, defining the pathways by which transit requests. This will prevent flooding the network links.

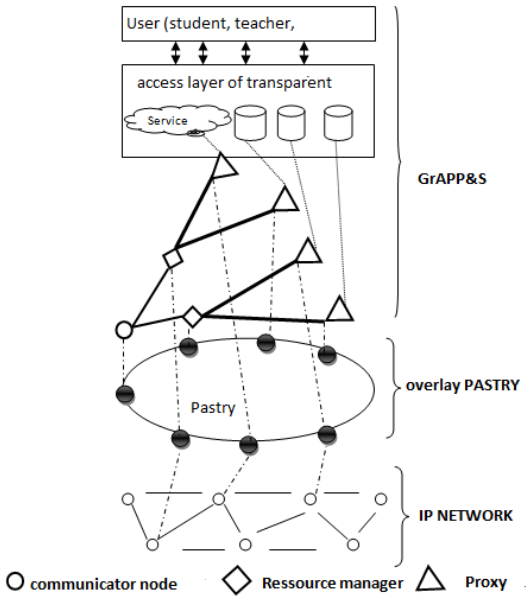

Figure 3. Deployment of nodes GRAPP&S on Pastry

4. Accessing the Resources

A GRAPP&S community is an hierarchical network with an addressing system independent from the underlying network. This addressing scheme is used to help data lookup and also to help route data during transfers. This hierarchical addressing also simplifies the integration to another network community. In the following paragraphs, we will propose a routing algorithm and a method of data lookup in our system.

4.1. Routing Algorithm

Let T be the tree of a community GRAPP&S. Thanks to the results of Fraigniaud *et al.* [7]; Thorup *et al.* [17] we can construct a routing scheme in the tree T.

Let T be a tree and a numbering of the vertices of T following a DFS [7] in the tree. For each vertex X identifier Id_X address of X consists of the binary string representing Id_X concatenated by the binary strings of parent vertices of X in T. The address of each vertex X is represented by a binary string $PATH\_X$ more than 3 bytes and an integer $Lpath\_X$ is the length of the chain.

For any vertex X of T:

Let $mask\_F$ the mask constituted by a binary string $Lpath_X$+8 bits including $Lpath\_X$ first bits are all 0 and the last 8 bits are 1 (e.g.: for a vertex X as $Lpath\_X$= 16, the mask will $mask\_X$= 00000000. 00000000. 11111111).

If Y is a descendant of X in T, then $PATH\_X$ is a substring of $PATH\_Y$ and applying a logical AND between $mask\_X$ = 00000000. 00000000. 11111111 and $PATH\_Y$ was number Y.

If Y is not a descendant of X in T, then it must go through the father of X to go to Y.

Let $is\_parent(X, Y)$ the function for any pair of vertex returns TRUE or FALSE depending on whether X is the parent of Y in the tree T or not.

The shipping method in our hierarchical system is modeled after [1]. The basic routing algorithm (see Algorithm 1) allows the transmission of messages between two vertices knowing the identifier of the source node and the destination node.

```
m : message sent from a source node to the destination node;
source : source node;
destination : destination node;
add : address of a node;
Procedure Route(m,source, destination)
    If (areNeighbors(source, destination)) then
        add ← getAdd(destination);
        send(m, add);
    else
        If (is_Parent(source, destination)) then
            add ←getAdd(mask_source AND destination);
            send (m, add);
        else
            add ← getAdd(father(source));
            send(m, Add);
            source ← add;
            Route (m, source, destination);
        end If
    end If
End
```

Algorithm 1: Basic forwarding method

4.2. Research Data in GRAPP&S

Among main functionalities on GRAPP&S there exists research and data access. This section describes our data search algorithm in a community GRAPP&S. The nodes that intervenes in this research has a few primitives:

- *Route_request* allows a source node to send a request to a destination node. *Route_request* uses the function *Route*() (cf. Algorithm 1) to send the message m so its signature is $Route\_request(m, source, destination)$.
- *Route_reply* allows to send a reply to m that follows the reverse path to the initiating node Message *Route_request*. The method *Route_reply* has almost the same structure as the *Route_request* function except that it does not use the same message in *Route_request*. The structure of the method *Route_reply* is $Route\_reply(m', source, destination)$.
- *Route_RequestDirect*, enables a source node to send a message directly to the recipient, if the communication mechanism allows it.
- *Save*() allows a node receiving a message *Route_reply* to store it on a non-volatile support(for example a file on the hard-drive, on Dropbox, on Gmail etc.).

When a customer searches any data on GRAPP&S, it contacts a DM_i proxy that sends a request Y containing the characteristics of the data. This research is conducted in steps so as to respect the hierarchical organization of the network, as follows:

- $DM_i \in C_i$ sends its request Y to node $RM_i \in C_i$ through *Route_request*() (cf. Algorithm 1);
- RM_i uses the method $Verif\_of\_RM_i()$ (cf. Algorithm 2) to check on its index if among his neighbors there is at least a DM that contains the requested information, thanks to the *Local_Search*() function (cf. Algorithm 3).
  - If yes, then the node RM_i returns to node DM_i a list of nodes DM that contain the requested information. An example is shown in Figure 4 where the node RM_i identifies id(RM_i)=AA and returns a list of nodes DM to the node Id(DM_i)=AAA.
  - Otherwise, the RM_i node forwards the request to the node $c_i \in C_i$ for broadcasting to all other nodes $RM_k \in C_i$ such that $RM_k \neq RM_i$ (cf. Algorithm 4). Figure 4 illustrates this distribution, where the node RM_i identifier $id(AA)$ forwards the request to its c_i identifier $id(A)$ for a broadcast to other nodes RM_k ($id(AB)$, $id(BA)$ and $id(BB)$).
- When a node $RM_k \in C_i$ finds a match for the searched information, it sends back an answare to sender node DM_i following the reverse path through *Route_reply*() function (cf. Algorithm 1). This is illustrated in Figure 4 where RM_k node identifier $id(BB)$ responds to the node DM_i identifier $id(AAA)$ taking the opposite route.
- If the requested data is not available in the community C_i, then c_i node may forward the request to other communities C_j. This is illustrated in Figure 4 where c_i node identifier $id(A)$ forwards the search request to the community C_j of node c_j with the specified $id(c_j)$ B.

4.3. Data Transfer in GRAPP&S

If successful in the search, the customer gets the node identifier DM_x responsible for the data. In this case, there are two possibilities for contacting $DM_x$and retrieve the data: using a direct connection or using a routed connection.

In the case of direct connections, the customer sends the request directly to DM_x by a simple *Send*() call (cf. Algorithm 6) in order to access the data. The customer can view or store the data with the *save*() function. In case this direct connection is not possible, then the data transfer is done by hierarchical routing in GRAPP&S,

based on the prefixes of identifiers of nodes to back up the hierarchy.

The steps in this routing prefix are as follows:

1. the customer sends a request $Y\ (m,\ id\ (DM_i),\ id(DM_x))$ to its RM_i, which forwards the request to the node communicator c_i;
2. using prefixed routing, node c_i sends the request Y to the RM node that has indexed the data. Using the same prefixed routing, the RM forwards the request to the node DM_x responsible for the data;
3. once the data is found on DM_x, it is sent back with the function $Route\_reply$(m', id(DM_x),id(DM_i)), returning the correct answer to DM_i through the prefixed tree.

```
Index_of_RM[0...n] : index node RM;
X ←{} : list of nodes DM;
TypeMime : mime type information;
TypeMimeRec : search term;
Function Verif_of_RM(TypeMimeRec) : X ←{}
    For (i from 0 to n) do
        [ If (Index_of_RM[i].TypeMime=TypeMimeRec ) then
            X1 ← DMi;
        end If
        ]
    end For
    return X;
End
```

Algorithm 2: Checking indexes RM

```
Procedure Local_Search()
    If (RM.Verif_of_RM()≠ empty) then
        Route(m', source, destination);
    end If
End
```

Algorithm 3: Local Search

```
For (each neighbor RMk of ci such as RMk ≠ RMi) do
    [ Route_request(m, source, destination) ]
end For
For (each neighbor RMk of ci such as RMk ≠ RMi) do
    [ If ( RMk.Verif_of_RM() ≠ empty ) then
        Route_reply(m', source, destination);
    end If
    ]
end For
```

Algorithm 4: Broadcasting message to nodes RMs

```
Function CanReach(source, destination) : boolean
    If (connect_direct(source, destination) is possible) then
        return True;
    else
        return False;
    end If
End
```

Algorithm 5: Method CanReach()

```
m : message sent from a source node to the destination node;
source : source node;
destination : destination node;
Procedure Get()
    If (CanReach(source, destination)=True) then
        send_RequestDirect(m, source, destination);
        send_reply(m', source, destination);
        source.save();
    else
        send message through route();
        Route_request(m, source, destination);
        resend message through route();
        Route_reply(m', source, destination);
    end If
    source.save();
End
```

Algorithm 6: Getting data in GRAPP&S

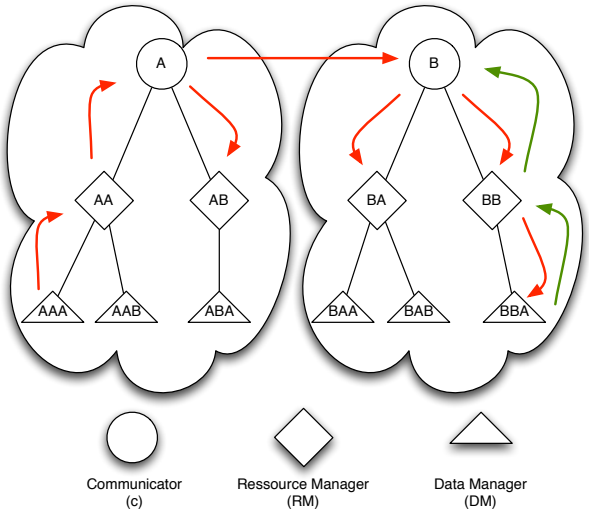

Figure 4. Resource lookup in a routing GRAPP&S

5. Conclusions

In this article we present an E-learning architecture specification named GRAPP&S, which is a decentralized solution for managing and sharing educational resources. GRAPP&S has been constructed as an hierarchical network to allows a pooling resources of each institute under the concept of "community", and to take advantage of the proximity of data and the users (students, teachers, administrative, etc.). One can even extend the use of a community of GRAPP&S to other sectors of an institute such as a community just for the administrative departments, separated from the educational users. In addition, GRAPP&S uses security rules to protect the resources of each community, and defines access policies between different community. The use of proxies for each specific type of data allows sharing a transparent manner not only files but also databases, streams (video, audio, VoIP), resources from the web services, cloud services, distributed computing.

The next step towards the validation of this specification is its use under real situations. We are starting to develop a prototype of GRAPP&S that will be used for testing and deployment on educational institutions. This prototype uses P2P Pastry underlying network to distribute the nodes of a community (local network institution) of GRAPP&S as illustrated in Figure 3. The choice of Pastry allows a community GRAPP&S a P2P network, which will allow a community institute include a large number of nodes and to perform scaling tests.

References

[1] Bakker, E.M., van Leeuwen, J. and Tan, R.B. (1993) Prefix routing schemes in dynamic networks. *Computer Networks and ISDN Systems* **26**(4): 403–421.

[2] Cayirci, E., Rong, C., Huiskamp, W. and Verkoelen, C. (2009) Snow leopard cloud: a multi-national education training and experimentation cloud and its security challenges. In *Cloud Computing* (Springer), 57–68.

[3] Chalopin, J., Godard, E., Métivier, Y. and Ossamy, R. (2006) Mobile agent algorithms versus message passing algorithms. In *Principle of distributed systems* (France: Springer), **4305**: 185–199.

[4] De Santiago, R. and Raabe, A. (2010) Architecture for learning objects sharing among learning institutionsâĂŤlop2p. *Learning Technologies, IEEE Transactions on* **3**(2): 91–95.

[5] Dietze, S., Yu, H.Q., Giordano, D., Kaldoudi, E., Dovrolis, N. and Taibi, D. (2012) Linked education: interlinking educational resources and the web of data. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (ACM): 366–371.

[6] Fardoun, H.M., Lopez, S.R., Alghazzawi, D.M. and Castillo, J.R. (2012) Education system in the cloud to improve student communication in the institutes of: C-learnixml++. *Procedia-Social and Behavioral Sciences* **47**: 1762–1769.

[7] Fraigniaud, P. and Gavoille, C. (2001) Routing in trees. In *Automata, Languages and Programming* (Springer), 757–772.

[8] Frankel, J. and Pepper, T. (2003) The gnutella protocol specification v0. 4.

[9] Hayakawa, T., Higashino, M., Takahashi, K., Kawamura, T. and Sugahara, K. (2012) Management of multimedia data for streaming on a distributed e-learning system (IEEE): 1282–1285.

[10] Kleinrock, L. and Kamoun, F. (1977) Hierarchical routing for large networks. *Computer Networks* **1**: 155–174.

[11] Klingberg, T. and Manfredi, R. (2002) The gnutella protocol specification v0. 6. *Technical specification of the Protocol* .

[12] Ouf, S., Nasr, M. and Helmy, Y. (2010) An enhanced e-learning ecosystem based on an integration between cloud computing and web2. 0. In *Signal Processing and Information Technology (ISSPIT), 2010 IEEE International Symposium on* (IEEE): 48–55.

[13] Peng, Z., Duan, Z., Qi, J.J., Cao, Y. and Lv, E. (2007) Hp2p: A hybrid hierarchical p2p network. In *Digital Society, 2007. ICDS'07. First International Conference on the* (IEEE): 18–18.

[14] Rowstron, A. and Druschel, P. (2001) Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware 2001* (Springer): 329–350.

[15] Shin, S., Jung, J. and Balakrishnan, H. (2006) Malware prevalence in the kazaa file-sharing network. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement* (ACM): 333–338.

[16] Stoica, I., Morris, R., Karger, D., Kaashoek, M.F. and Balakrishnan, H. (2001) Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM Computer Communication Review* (ACM), **31**: 149–160.

[17] Thorup, M. and Zwick, U. (2001) Compact routing schemes. In *Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures* (ACM): 1–10.

[18] Traversat, B., Arora, A., Abdelaziz, M., Duigou, M., Haywood, C., Hugly, J.C., Pouyoul, E. *et al.* (2003) Project jxta 2.0 super-peer virtual network. *Sun Microsystem White Paper. Available at www. jxta. org/project/www/docs* **92**.

[19] Xu, C.Z., Rao, J. and Bu, X. (2012) Url: A unified reinforcement learning approach for autonomic cloud management. *Journal of Parallel and Distributed Computing* **72**(2): 95–105.