Distributed Computing Systems: P2P versus Grid Computing Alternatives

A. Cabani*, S. Ramaswamy+, M. Itmi*, S. Al-Shukri+, J.P. Pécuchet*

***Abstract*-Grid and P2P systems have become popular options for large-scale distributed computing, but their popularity has led to a number of varying definitions that are often conflicting. Taxonomies developed to aid the decision process are also quite limited in their applicability. While some researchers have argued that the two technologies are converging [1], in this paper, we develop a unified taxonomy along two necessary distributed computing dimensions and present a framework for identifying the right alternative between P2P and Grid Computing for the development of distributed computing applications.**

1. Introduction

A distributed computing system is defined as a collection of independent computers that appear to their users as a single computing system [2]. Distributed software systems are increasingly being used in modern day software systems development to tackle the issues of geographically separated work groups and increasing application complexities. Often this constitutes the interconnection of autonomous software residing on individual machines through communication networks to enable their users to cooperate and coordinate to successfully accomplish their objectives.

The widespread need for a distributed system based solution is due to the need for resource sharing and fault-tolerance. Resource sharing implies that a distributed system allows its resources - hardware, software and data – to be appropriately shared amongst its users. Fault-tolerance means that machines connected by networks can be viewed as redundant resources, a software system could be installed on multiple machines to withstand hardware faults or software failures [3].

For a distributed system to support active resource sharing and fault-tolerance within its multitude of nodes, it needs to possess certain key properties. These include openness and transparency. Openness in a distributed system is achieved by specifying its key interface elements and making it available to other software developers so that the system can be extended for use. Distributed systems generally tend to provide three forms of transparency. These include: (i) Location transparency, which allows local and remote information to be accessed in a unified way; (ii) Failure transparency, which enables the masking of failures automatically; and (iii) Replication transparency, which allows duplicating software/data on multiple machines invisibly.

To be able to demonstrate the above properties, in turn such a distributed system must provide support for concurrency and be built on a scalable architectural framework. Concurrency refers to the simultaneous processing of requests to multiple interconnected machines / networks. Scalability refers to the adoption of an interconnection network architecture that allows for seamless extendibility to a large number of machines and/or users to support the needs of increased processing power requirements.

In this paper, we compare and contrast two currently popular approaches for distributed computing applications: Grid and P2P approaches. The objective of both P2P and grid computing is the collective, coordinated use of a large number of resources scattered in a distributed environment. However the user communities that have adopted and popularized these two approaches are vastly different, both in terms of their user-level requirements as well as the architectural design of the systems themselves.

This paper is organized as follows: Section 2 and 3 present a brief overview of Grid and P2P computing. Section 4 presents the unified taxonomy along two necessary distributed computing dimensions. Section 5 compares and contrasts Grid and P2P computing using a set of commonly desired criteria for a distributed computing solution. Section 6 concludes the paper.

2. Grid Computing

According to IBM's definition [4]: "A grid is a collection of distributed computing resources available over a local or wide area network that appear to an end user or application as one large virtual computing system. The vision is to create virtual dynamic organizations through secure, coordinated resource-sharing among individuals, institutions, and resources. Grid computing is an approach to distributed computing that spans not only locations but also organizations, machine architectures and software boundaries to provide unlimited power, collaboration and information access to everyone connected to a grid."

Another definition, this one from The Globus Alliance is [5]: "The grid refers to an infrastructure that enables the integrated, collaborative use of high-end computers, networks, databases, and scientific instruments owned and managed by multiple organizations. Grid applications often involve large amounts of data and/or computing and often require secure resource sharing across organizational

* LITIS Laboratory EA4051, INSA-Rouen, 76131 Mt. St-Aignan Cedex, France. This work was supported, in part, by INSA-Rouen - for supporting Dr. Ramaswamy as a visiting professor at INSA during June 2006.
Email: adnane.cabani@insa-rouen.fr / itmi@insa-rouen.fr / jean-pierre.pecuchet@insa-rouen.fr

+ Knowledge Enterprises for Scalable Resilient Infrastructures (KESRI), Computer Science Department, Univ. of Arkansas at Little Rock, AR 72204 USA. This work was supported, in part, by an NSF MRI Grant #CNS-0619069 and the Graduate Institute of Technology at UALR.
Email: srini@acm.org / shalshukri@ualr.edu

T. Sobh (ed.), Innovations and Advanced Techniques in Computer and Information Sciences and Engineering, 47–52.

boundaries, and are thus not easily handled by today's Internet and Web infrastructures."

Grid computing evolved out of the scientific realm where there was a need to process and analyze increasingly colossal quantities of data such as those needed to perform weather or climatic forecasts [6, 7], to model and calculate the aerodynamic behavior of a plane, in genomics [8], etc. Such application specific needs were not only at odds with the wait times for technological evolution but also were highly intolerant to the advances attained by means of Moore's law [9] with respect to performance (doubling every eighteen months), storage (doubling every twelve months), and networks (double every nine months). Since the speed of network performance outpaces that of processors and storage, it has led to the increasing interconnection of geographically scattered resources by means of a dynamic network that accumulates the capacities of calculation, storage, etc.

Grid computing [10-12] thus allows for better resource sharing between users in an institution to resolve a common problem. From a business perspective, the purpose of grid computing is to minimize the time to market, thereby profiting from the infrastructure costs incurred. Grid computing offers its users' access to processing across diverse storage structures which are transparently distributed geographically. It is based on the concept of on demand data processing wherein a user pays according to their needs and resource consumption.

3. Peer-to-peer Computing

As contrasted from grid computing, peer-to-peer computing refers to a network of equals that allows two or more individuals to spontaneously collaborate without necessarily needing any centralized coordination [13]. P2P computing was made famous by a music sharing frenzy generated by Napster, which was initially a server based centralized architecture. Other P2P systems have since appeared without the limitation imposed by a centralized server. Here, a user seeking a file (song, video, software) sends their query which is incrementally forwarded by the nodes of the network, thereby creating an ad-hoc chain between the requester's PC requester to the supplier's PC – culminating in the transfer of the requested file. Examples include Limewire [14], Kazaa [15], eDonkey [16], BitTorrent [17]; which are some popular P2P systems. When designing P2P applications, it is important to assume that peers are untrustworthy [18]. While they are designed to interconnect and communicate with each other they can join or quit dynamically from the P2P network. When a node quits, there will be communication failures. This makes the development of P2P applications a very challenging task.

While P2P technology can be applied to many application domains, most current utilization is customer targeted with the primary focus on file sharing. These systems allow files to be easily shared and quickly propagated through the Internet without powerful host servers. Other applications include:

1. Personal productivity applications: Collaboration between individual users, i.e. sharing address books, schedules, notes, chatting, etc. allows improvements in productivity. Connecting such desktop productivity software systems together enables collaborative e-business communities to form for flexible, productive, and efficient working teams. For example, Java developers have used OpenProjects.net to collaborate. On a broader scale, hundreds of thousands of uses use instant messaging, one of the most popular P2P applications to date.
2. Enterprise resource management: These systems allow the coordination of workflow processes within an organization thereby leveraging the existing network infrastructure for improvements in business productivity. For example, Groove [19] enables an aerospace manufacturers to post job order requests to partner companies and route the completed requests from one department to the next.
3. Distributed computation: A natural extension of the Internet's philosophy of robustness through decentralization is to design peer-to-peer systems that send computing tasks to millions of servers, each one possibly also being a desktop computer.

In [20, 21] the authors present a taxonomy for P2P applications and distinguish three specific categories of P2P applications. The specific classes include:

1. Parallel applications: In such applications, a large calculation is split into several small independent entities that can be executed independently on a large number of peers - SETI@Home [22], genome@Home, etc. Another possibility is making the calculation of the same operation but with different data sets or parameters. Such computation kind called a parametric study computation. Example: fluid dynamics simulation. The goal is to solve computational problems and cycle-sharing. P2P cycle-sharing and grid computing are converged but its origins are different [11, 23]. In P2P cycle-sharing the whole application runs on each peer and no need communication between the peers.
2. Content and file management: Content encapsulates several types of activities and refers to anything that can be digitized; for example, messages, files, binary software. It essentially consists of storing, sharing and finding various kinds of information on the network. The main application focus is content exchange. Such projects aim to establish a system of files for distribution within a community and

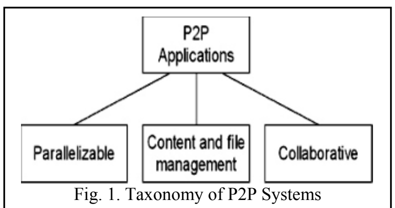

Fig. 1. Taxonomy of P2P Systems