Journal of Network and Computer Applications 32 (2009) 961–975

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

BitDew: A data management and distribution service with multi-protocol file transfer and metadata abstraction

Gilles Fedak [a,b,*], Haiwu He [a,b], Franck Cappello [a,b]

[a] INRIA Saclay, Grand-Large, Orsay, F-91893, France
[b] LRI, Univ Paris-Sud, CNRS, Orsay, F-91405, France

ARTICLE INFO

Article history:
Received 5 November 2008
Received in revised form
1 April 2009
Accepted 3 April 2009

Keywords:
Content network
P2P
Cloud computing
Desktop Grid

ABSTRACT

Desktop Grids use the computing, network and storage resources from idle desktop PCs distributed over multiple-LANs or the Internet to compute a large variety of resource-demanding distributed applications. While these applications need to access, compute, store and circulate large volumes of data, little attention has been paid to data management in such large-scale, dynamic, heterogeneous, volatile and highly distributed Grids. In most cases, data management relies on ad hoc solutions, and providing a general approach is still a challenging issue. A new class of data management service is desirable to deal with such a variety of file transfer protocols than client/server, P2P or the new and emerging Cloud storage service.

To address this problem, we propose the BitDew framework, a programmable environment for automatic and transparent data management on computational Desktop Grids. This paper describes the BitDew programming interface, its architecture, and the performance evaluation of its runtime components. BitDew relies on a specific set of metadata to drive key data management operations, namely life cycle, distribution, placement, replication and fault tolerance with a high level of abstraction. The BitDew runtime environment is a flexible distributed service architecture that integrates modular P2P components such as DHTs (Distributed Hash Tables) for a Distributed Data Catalog and collaborative transport protocols for data distribution. We explain how to plug-in new or existing protocols and we give evidence of the versatility of the framework by implementing HTTP, FTP and BitTorrent protocols and access to the Amazon S3 and IBP Wide Area Storage. We describe the mechanisms used to provide asynchronous and reliable multi-protocols transfers. Through several examples, we describe how application programmers and BitDew users can exploit BitDew's features. We report on performance evaluation using micro-benchmarks, various usage scenarios and data-intense bioinformatics application, both in the Grid context and on the Internet. The performance evaluation demonstrates that the high level of abstraction and transparency is obtained with a reasonable overhead, while offering the benefit of scalability, performance and fault tolerance with little programming cost.

1. Introduction

Enabling Data Grids are one of the fundamental efforts of the computational science community as emphasized by projects such as EGEE (Enabling Grids for E-Science in Europe) and PPDG (PPDG, 2006). This effort is pushed by the new requirements of e-Science. That is, large communities of researchers collaborate to extract knowledge and information from huge amounts of scientific data. This has lead to the emergence of a new class of applications, called *data-intensive* applications which require secure and coordinated access to large data sets, wide-area transfers and broad distribution of TeraBytes of data while keeping track of multiple data replicas. The Data Grid aims at providing such an infrastructure and services to enable data-intensive applications.

Our project, BitDew[1] (Fedak et al., 2008a, b), targets not only traditional Grids (which connect computational and storage resources usually at universities or enterprises through virtual organizations), but also a specific class of Grids called Desktop Grids. Desktop Grids use computing, network and storage resources of idle desktop PCs distributed over multiple LANs or the Internet. Today, this type of computing platform forms one of

* Corresponding author at: INRIA Saclay, Grand-Large, Orsay, F-91893, France.
E-mail addresses: gilles.fedak@inria.fr (G. Fedak), haiwu.he@inria.fr (H. He), fci@lri.fr (F. Cappello).

[1] BitDew can be found at http://www.bitdew.net under GPL license.

1084-8045/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2009.04.002

the largest distributed computing systems, and currently provides scientists with tens of TeraFLOPS from hundreds of thousands of hosts. Despite the attractiveness of this platform, little work has been done to support data-intensive applications in this context of massively distributed, volatile, heterogeneous, and network-limited resources. Most Desktop Grid systems, like BOINC (Anderson, 2004), XtremWeb (Fedak et al., 2001), Condor (Litzkow et al., 1988) and OurGrid (Cirne et al., 2006) rely on a centralized architecture for indexing and distributing the data, and thus potentially face issues with scalability and fault tolerance.

However, we believe that the basic blocks for building BitDew can be found in P2P systems. Researchers of DHTs (Distributed Hash Tables) (Stoica et al., 2001; Maymounkov and Mazires, 2002; Rowstron and Druschel, 2001) and collaborative data distribution (Cohen, 2003; Gkantsidis and Rodriguez, 2005; Fernandess and Malkhi, 2006), storage over volatile resources (Adya et al., 2002; Butt et al., 2004; Vazhkudai et al., 2005), wide-area network storage (Bassi et al., 2002; Kubiatowicz et al., 2000) and Cloud computing (Amazon web services) offer various tools that could be of interest for Data Grids. To build Data Grids from and to utilize them effectively, one needs to bring together these components into a comprehensive framework. BitDew suits this purpose by providing an environment for data management and distribution in Desktop Grids.

BitDew is a subsystem which could be easily integrated into other Desktop Grid systems. It offers programmers (or an automated agent that works on behalf of the user) a simple API for creating, accessing, storing and moving data with ease, even on highly dynamic and volatile environments.

BitDew leverages the use of metadata, a technique widely used in Data Grid (Jin et al., 2006), but in more directive style. We define five different types of metadata: (i) REPLICATION indicates how many occurrences of data should be available at the same time in the system, (ii) FAULT TOLERANCE controls the resilience of data in presence of machine crash, (iii) LIFETIME is a duration, absolute or relative to the existence of other data, which indicates when a datum is obsolete, (iv) AFFINITY drives movement of data according to dependency rules, (v) TRANSFER PROTOCOL gives the runtime environment hints about the file transfer protocol appropriate to distribute the data. Programmers tag each data with these simple attributes, and simply let the BitDew runtime environment manage operations of data creation, deletion, movement, replication, as well as fault tolerance.

The BitDew runtime environment is a flexible environment implementing the APIs. It relies either on centralized or and distributed protocols for indexing, storage and transfers providing reliability, scalability and high performance. In this paper, we present the architecture of the prototype, and we describe in depth the various mechanisms used to provide asynchronous, asynchronous and multi-protocol file transfers. We give several examples of protocol integration: client/server such as HTTP and FTP, P2P or collaborative content delivery such as BitTorrent, and Wide Area Storage such as IBP or Amazon S3. We also provide detailed quantitative evaluation of the runtime environment on two environments: the GRID5000 experimental Grid platform, and DSL-Lab, an experimental platform over broadband ADSL.

Through a set of micro-benchmarks, we measure the costs and benefits, components by components, of the underlying infrastructures. We run communication benchmark in order to evaluate the overhead of the BitDew protocol when transferring files and we assess fault-tolerant capabilities. And finally we show how to program a master/worker application with BitDew and we evaluate its performance in a real world Grid deployment.

The rest of the paper is organized as follows. Section 2 presents the background of our researches. In Section 3, we present the API and the runtime environment of BitDew. Then in Section 4, we conduct performance evaluation of our prototype, and Section 5 presents a master/worker application. Finally we present related work in Section 6 and we conclude the paper in Section 7.

In this section we overview Desktop Grids characteristics and data-intensive application requirements. Following this analysis, we give the required features of BitDew.

1.1. Desktop Grids characteristics

Desktop Grids are composed of a large set of personal computers that belong both to institutions, for instance an enterprise or a university, and to individuals. In the former case, these home PCs are volunteered by participants who donate a part of their computing capacities to some public projects. However, several key characteristics differentiate DG resources from traditional Grid resources: (i) performance; mainstream PCs have no reliable storage and potentially poor communication links, (ii) volatility; PCs can join and leave the network at any time and appear with several identities, (iii) resources are shared between their users and the Desktop Grid applications, (iv) resources are scattered across administrative domains with a wide variety of security mechanisms ranging from personal routers/firewalls to large-scale PKI infrastructures.

Because of these constraints, even the simplest data administration tasks, are difficult to achieve on a Desktop Grid. For instance, to deploy a new application on a cluster, it is sufficient to copy the binary file on a network file server shared by the cluster nodes. After a computation, cluster users usually clean the storage space on the cluster nodes simply by logging remotely to each of the compute nodes and by deleting recursively the temporary files or directories created by the application. By contrast, none of the existing Desktop Grids systems allows such tasks to be performed because: (i) a shared file system would be troublesome to setup because of hosts connectivity and volatility and volunteers churn, and (ii) remote access to participant's local file system is forbidden in order to protect volunteer's security and privacy.

1.2. Requirements to enable data-intensive application on Desktop Grids

Currently, Desktop Grids are mostly limited to embarrassingly parallel applications with few data dependencies. In order to broaden the use of Desktop Grids we examine several challenging applications and outline their needs in terms of data management. From this survey, we will deduce the features expected from BitDew.

Parameter-sweep applications composed of a large set of independent tasks sharing large data are the first class of applications which can benefit from BitDew. Large data movement across wide-area networks can be costly in terms of performance because bandwidth across the Internet is often limited, variable and unpredictable. Caching data on local workstation storage (Iamnitchi et al., 2006; Otoo et al., 2004; Vazhkudai et al., 2005) with adequate scheduling strategies (Santos-Neto et al., 2004; Wei et al., 2005) to minimize data transfers can improve overall application execution performance.

Moreover, the work in Iamnitchi et al. (2006) showed that data-intensive applications in high energy physics tend to access data in groups of files called "filecules". For these types of applications, replication of groups of files over a large set of resources is essential to achieve good performance. If data are replicated and cached on local storage of computing resources, one should provide transparent fault-tolerance operation on data.

In a previous work (Wei et al., 2005), we have shown that using a collaborative data distribution protocol BitTorrent over FTP can