Contributions to Desktop Grid Computing

Gilles Fedak

University of Lyon

Habilitation à Diriger des Recherches

Contributions to Desktop Grid Computing

From High Throughput Computing to Data-Intensive Sciences on Hybrid Distributed Computing Infrastructures

Gilles Fedak

28 Mai 2015

Après avis de :

Christine	Morin	Directeur de Recherche, INRIA
Pierre	Sens	Professeur, Université Paris VI
Domenico	Talia	Professeur, University of Calabria

Devant la commission d'examen formée de :

Vincent	Breton	Directeur de Recherche, CNRS
Christine	Morin	Directeur de Recherche, INRIA
Manish	Parashar	Professeur, Rutgers University
Christian	Perez	Directeur de Recherche, INRIA
Pierre	Sens	Professeur, Université Paris VI
Domenico	Talia	Professeur, University of Calabria

Contents

Abstract

Since the mid 90's, Desktop Grid Computing - i.e the idea of using a large number of remote PCs distributed on the Internet to execute large parallel applications - has proved to be an efficient paradigm to provide a large computational power at the fraction of the cost of a dedicated computing infrastructure.

This document presents my contributions over the last decade to broaden the scope of Desktop Grid Computing. My research has followed three different directions. The first direction has established new methods to observe and characterize Desktop Grid resources and developed experimental platforms to test and validate our approach in conditions close to reality. The second line of research has focused on integrating Desktop Grids in e-science Grid infrastructure (e.g. EGI), which requires to address many challenges such as security, scheduling, quality of service, and more. The third direction has investigated how to support large-scale data management and data intensive applications on such infrastructures, including support for the new and emerging data-oriented programming models.

This manuscript not only reports on the scientific achievements and the technologies developed to support our objectives, but also on the international collaborations and projects I have been involved in, as well as the scientific mentoring which motivates my candidature for the Habilitation à Diriger les Recherches.

Acknowledgements

I would like to express my gratitude to the members of the defense committee: Christine Morin, Pierre Sens and Domenico Talia, who accepted to review the manuscript and Vincent Breton, Christian Perez and Manish Parashar, who accepted to participate to the defense.

I would sincerely like to thank the GRAAL/Avalon team and the LIP members for their kindness, support and for all the fruitful discussions we have daily. In particular, I would like to thank Christian Perez, head of the Avalon team for his constant support during these last years spent in Lyon.

A great deal of credit should be given to the PhD students, post-docs and engineers without whom none of this would have been possible: Anthony Simonet, Paul Malécot, Baoha Wei, Asma Ben Cheick, Julio Anjos, Bing Tang, Mircea Moca, Haiwu He, José Saray, Simon Delamare, François Bérenger, Derrick Kondo and more.

This work gave me the privilege to meet many awesome people; some of them with whose I developed over the years a sincere and friendly relationship. Without naming them, I would like to thank all my co-authors and collaborators from around the world: US, China, Japan, Tunisia, Ukraine, Canada, Brazil, Romania and more.

I would like to friendly salute Oleg Lodygensky, working at IN2P3, with whom I have shared a lot of this adventure.

I would like to dedicate this manuscript to my beloved ones : Irina, Lison and Virginie.

Chapter 1

Introduction

> Learning is experience. Everything else is just information.
>
> *(Albert Einstein (1879 – 1955))*

This document presents the research I have done since receiving my Ph.D. in 2003. These activities started during my postdoctoral stay at the University of California San Diego and continued at University Paris XI in the Grand-Large team where I occupied a position of INRIA Researcher between 2004 and 2008. In 2008, I joined the GRAAL/AVALON team at the University of Lyon, whose research activities aim at designing software, algorithms and programming models for large scale distributed computing infrastructures. Pursuing similar objective, I focused my research work on studying infrastructures and technologies for Desktop Grid Computing.

1.1 Historical Context

Thorough this document, the reader will follow a part of the history of parallel and distributed computing. This story illustrates the efforts to build more powerful computing infrastructures by designing *large scale* systems, that is to say, capable of assembling millions of computing nodes [1]. Desktop Grid Computing [2] is a discipline which, by considering *the whole Internet as a possible computing platform*, has pushed this idea to its extreme limit. It's hard to figure out now, how disruptive this approach was, when it emerged at the late 90's. To put things into context, Desktop Grid has appeared at the same time than Cluster Computing, which was aiming at designing parallel computers using off-the shelf components such as regular PC run by Linux and early Grid Computing, where the first experiments were considering parallel applications spanning over several super-computers geographically distributed. Thus, the environment was favorable to research leading to new directions for high performance computing. The characteristics of a Desktop Grid platform is radically different from a traditional supercomputer: nodes are distributed over the Internet, nodes can join or leave the network

at any time without notice, nodes have low network capabilities with many connection restrictions and nodes are shared with end-users.

Our context was so radically different, that we had to invent new paradigms and technologies before being able to explore the full possibility of Desktop Grid Computing. Indeed, Desktop Grid has revisited many traditional aspects of high performance computing: scheduling, file management, communications, programming model and more. Moreover, it has also revealed in advance several problematics that were considered as minor and secondary in the classical field of parallel computing, such as system security and dependability, and which are now widely addressed as a primary concern, at the age of Cloud Computing and Peta-scale High Performance Computing.

This thesis presents my contributions to the domain, both in term of technology and scientific results. One particular aspect of research in Desktop Grid computing is that it requires to develop the technology which enables to build the infrastructure, and at the same time to observe and understand the infrastructure in order to improve the technology. Thus, my research follows two axis which are advanced in parallel. The first area concerns Desktop Grid *algorithms and software*, and addresses a large variety of topics such as performance, programming model, security, data management, fault-tolerance etc. The second direction concerns the Desktop Grid *infrastructures*, which consists in observing the infrastructures and characterizing the computing resources. This helps to develop new experimental methodologies that were rather taken from the P2P world and that we have applied to the study of computing infrastructure.

Finally, since the beginning of the field and up to now, more than a decade has passed. Desktop Grid Computing, and more generally speaking, the landscape of Distributed Computing Infrastructure (DCI) has greatly evolved. The reader will also follow a part of the history and how we have integrated Desktop Grid computing to the existing DCI paradigms, so that we made Desktop Grid a first class citizen.

1.1.1 From High Throughput Computing on Volatile Desktop Computers...

The starting point of the XtremWeb project [3] is an interdisciplinary collaboration between scientists belonging to the Paris Sud University: Alain Cordier (Laboratory of Linear Accelerator) and Franck Cappello (Laboratory of Computer Science). The physicists' lab was participating to an international collaboration called the Pierre Auger Cosmic Ray Observatory, which is studying ultra-high energy cosmic rays, the most energetic and rarest of particles in the universe. A significant part of their simulation campaign was aiming at simulating particles striking the earth's atmosphere, which produces extensive air showers made of billions of secondary particles. While much progress has been made in nearly a century of research in understanding cosmic rays with low to moderate energies, those with extremely high energies remain mysterious. Because such simulation campaign based on Monte-Carlo application is known to be embarrassingly parallel, the physicists were looking for an alternative infrastructure that would not over-use their regular super-computer center. The second motivation was that particles entering the atmosphere is a rare event, which is difficult to observe. It was anticipated that once detected, the observation would provoke a huge demand in computing power

and physicists were looking for the largest pool of resources possible in addition to their computing centers. Inspired by the first successes of distributed computing projects such as distributed.net and SETI@Home, Franck proposed the XtremWeb platform that I further developed during my Ph. D thesis. My main source of inspiration was the P2P file sharing software (Napster, Kazaa), which have drawn the idea that we could design a platform where each user could at the same time be a resource provider and take advantage of other contributors' resources. We rapidly sketched the objectives, features and prototyped the software accordingly. The XtremWeb vision was to provide an execution runtime environment supporting multiple applications and multiple users, and taking its computing power by using idle computing resources, distributed on the Internet and located in existing local area network, such as classrooms or data centers. This vision was translated into an architecture and a set of principles that did not evolve during the following years of research and development: it consists of a client-server-worker architecture, a push-pull scheduler, a security based on sandboxing and a fault-tolerance protocol based on host failure detection. As soon as we succeeded in executing the first applications from the Pierre Auger Observatory, we began to explore new programming models for the platform. An important milestone was the support of message-passing parallel application on volatile nodes, which probably represents the most challenging class of application that could be ported to Desktop Grid Computing. At the end of my Ph. D, I was genuinely convinced that extending Desktop Grid computing to new classes of applications was less an issue of "how-to", i.e. implementing adequate execution environment, than a question of "how good", i.e. finding out the algorithms to make efficient application execution. I took the opportunity of one year postdoctoral fellowship in A. Chien's team at UCSD and work with H. Casanova's, who was opening a new research direction dedicated to scheduling on volatile resources. This post-doc experience allowed me to develop the experimental framework required for the observation and characterization of Desktop Grid resources and the conceptual framework to understand the impact of node volatility on the performance of application execution. Up to now, I'm still relying heavily on this heritage whenever simulations or analytical evaluation of system performances are required.

1.1.2 ... to Data-Intense Processing on Hybrid Distributed Infrastructures

In the meanwhile of our research around Desktop Grid, the technologies for Parallel and Distributed Computing have strongly evolved. One remarkable evolution was the shift from High Throughput Computing (HTC), a computing paradigm that focuses on the efficient execution of Bag-of-Tasks (BoT) applications, i.e. consisting of a large number of loosely-coupled tasks to Data-Intensive Computing, i.e parallel applications which use a data parallel approach to process large volumes of data. If Desktop Grids have been proved extremely efficient systems for HTC, concerning Data-Intensive Computing, everything had to start from scratch.

In order to broaden the use of Desktop Grids, I examined several challenging applications (e.g. data-intensive bag-of-tasks application, long running applications which requires checkpointing, workflow application with tasks dependencies, MapReduce-like

flow of execution, and more) and came to the conclusion that these applications have very strong needs in terms of data management, which where not satisfied by existing Desktop Grid technologies. Most Desktop Grid systems rely on a centralized architecture for indexing and distributing the data, and thus potentially face issues with scalability and fault tolerance. Moreover, many basic blocks have been developed by researchers in P2P systems (Distributed Hash Tables, collaborative file distribution, storage over volatile resources and wide-area network storage), which addresses many challenges relevant to Data Desktop Grids: scalable and resilient data indexing, efficient data distribution, etc. Thus, the challenge was to re-architecture classical Desktop grid systems so that it can integrate some aspects of P2P technologies, while keeping the same level of performance, security and manageability than those allowed by a centralized design.

The second evolution that happened during this past decade was the consolidation and wide availability of Distributing Computing Infrastructures (DCI), in particular Grid and Cloud Computing infrastructures. In the U.S and in Europe, pushed by a strong effort towards standardization and by significant support from international institutions, several Grid infrastructures have been established as the main computing facilities to support e-Science communities. The European Grid Infrastructure is an example of a large computing infrastructure established to support High Energy Physics. Following Grid Computing, the advent of Cloud Computing has made DCI available to a larger audience, including private companies and smaller scientific communities. The result is that scientific users now have at their disposal several kinds of DCIs, that can be used simultaneously; we call this assemblage of Grids, Clouds and Desktop Grids an *Hybrid Distributed Computing Infrastructures*. Because infrastructures are characterized by different attributes such as price, performance, trust, greenness, combining these infrastructures in such a way that meets users' and applications' requirements raises significant scheduling challenges.

Thus, I have identified three main bottlenecks that prevented Desktop Grid to be first class citizen amongst the existing technologies to build Distributed Computing Infrastructures:

- Lacks of tools and methodological concepts to deeply understand the characteristics and performances of Desktop Grid platforms. The first challenge is to design observation and characterization methods for real-world Desktop Grids in terms of computing capabilities, reliability, resources volatility and trust. The second challenge is to establish experimental platforms that allow experiment reproducibility either using real execution platform or by accurately re-creating an execution by simulation or emulation.

- Desktop Grid systems should be as usable as regular Distributed Computing Infrastructure. The first challenge is to integrate Desktop Grid systems in the e-science infrastructure and solve the interoperability issues between Desktop Grids and other DCIs. In particular, this includes implementing the same standards than Grid computing, with respect to job submissions, security, user authentication, resources monitoring and so forth. The second challenge is that user experience

should be similar when using Desktop Grid, Cloud or Grid infrastructures. This implies to provide advanced features to Desktop Grid middleware, such as the support for virtualization technologies to improve the portability of scientific applications, and an improved QoS so that a probabilistic guaranty for the user is given on application completion time. The last challenge is to re-factor applications so that they can be executed on hybrid infrastructures, to mitigate the drawbacks of some infrastructures and enjoy the benefits of others.

- Support for Data-intensive applications is the third bottleneck that prevents Desktop Grid systems for being adopted in a large number of scientific disciplines. This requires that a data management system is able to efficiently execute the main data operations: storage ensuring data availability, security and privacy, efficient distribution of large files to high number of nodes, collective file communication following patterns such as broadcast or gather/scatter and smart user-driven data placement. In addition, an execution environment should be provided as well for programming languages dedicated to data intensive computing, such as MapReduce for example.

1.2 Contributions

The three challenges mentioned in the above sections have been developed into corresponding research directions. In this Section, I summarize the main contributions and briefly indicates the organization of the research (support, collaboration, PhD and postdoc advising) that lead to these results.

1.2.1 Research Achievements

- The first research direction aims at providing a solid background for the evaluation of our research by providing *new platforms and methodologies for characterizing and experimenting with Desktop Grids.*

  Although we know that Desktop Grids resources are heterogeneous and volatile, new methods are required to precisely observe and characterize the computing resources in existing and deployed Desktop Grid systems. I have implemented such methods in two different deployment contexts: local enterprise and campus Desktop Grids system (joint work with H. Casanova and D. Kondo at UCSD), and Internet volunteer system, in collaborations with D. Anderson (SETI@Home, UC. Berkeley). Furthermore, activity traces collected during these experiments have been made available to the research community in distributed system through the Desktop Grid Trace Archive, and later through the Failure Trace Archive [4].

  The second main contribution of this research direction addresses the challenge to conduct experiments in controlled and reproducible experimental conditions as close as possible to the reality of an actual Internet-wide deployment. In 2006, I started and lead the project DSLLAB, funded by the French Research Agency, in

partnership with the INRIA MESCAL team in Grenoble, which allowed to design and deploy an innovative platform dedicated to perform experiments on nodes distributed on the broadband DSL Internet.

The main contributors to this research direction are:

- The Ph.D. work thesis of Paul Malécot (co-advised with Franck Cappello, FP6 Grid4all funding, 2006-2010) has been at the center of the DSLLAb project. Paul's contributions are in the design and development of the DSLLAB and the XtremLab platforms.
- With Derrick Kondo, (post-doc, 2006-2007, ANR JCJC DSLLAB funding) we worked on the XtremLab project on characterizing and evaluating Desktop Grid and investigated new algorithms for resource management, error detection and recovery.

- The second research direction aims at the *integration of Desktop Grids within regular and existing e-Science Cyber-infrastructures* with the aim of providing additional computing capabilities at a reduced cost.

  In 2007, I joined an European collaboration involving several academic partners and strongly supported by several European FP7 grants. In particular, I took work package leadership in two FP7 projects: EDGeS (Enabling Desktop Grid e-Science) and EDGI (European Desktop Grid Initiative), which was aiming at set-up the first computing infrastructures based on Desktop Grid technologies that could transparently be used by regular users of the European Grid Infrastructure (EGI).

  The contributions in this research axis are both algorithms and software in the field of security, support for virtualization technologies, quality of service, as well as several scheduling heuristics. Eventually, I started the CloudPower project, funded by the French ANR, to study the possibility to transfer our research results and technologies to create a new innovative start-up company that would offer low-cost, scalable and secure HPC-on demand service for innovative small businesses.

  - Haiwu He, (post-doc, 2007-2009, FP7 EDGeS funding) has contributed to the bridge technologies that allow jobs workload to flow between Grid and Desktop Grid infrastructures.
  - Simon Delamare 's (post-doc, 2011-2012, FP7 EDI funding) main contribution is SpeQuloS, a QoS service which provides to the application executed on EDGI a prediction of their execution time and a probabilistic guaranty that they'll meet the expected completion time.

- The third research direction concerns the *support for Data-intensive science in hybrid distributed computing infrastructure.*

  We first designed and developed the BitDew software, which allows large scale data management on hybrid infrastructures. Besides, I joined the ANR project Clouds@Home and lead the ADT INRIA to support the development of BitDew.

More recently, in collaboration with Matei Ripeanu (UCB, Vancouver, Canada), we started Active Data, a project around data life cyle management.

The second contribution addresses the execution of data intensive applications on hybrid DCI. This research direction started when we were looking at the feasability of executing data-intense applications on Desktop Grid using P2P protocols. Based on BitDew, and in collaboration with Huazong University of Science and Technology (Wuhan, China) and the University of Babes-Bolaj (Cluj, Romania), we explored several research directions around the challenge of executing MapReduce [5] to allow data-centric computing on hybrid infrastructures: middleware design, scheduling, security, performance evaluation. Part of this work has been achieved thanks to the ANR MaReduce project.

- Baohua Wei (PhD co-advised with F. Cappello, 2004-2005, Chinese corporate funding) started his PhD thesis on Data Desktop Grid, until he resigned prematurely for personal reason.
- Bing Tang (Postdoc, 2011-2012, ANR Clouds@home funding) is a key contributor of BitDew and research around storage and MapReduce runtime environment for hybrid infrastructure.
- Anthony Simonet's PhD thesis main proposition (PhD, 2011-2015, ANR MapReduce funding) is Active Data, a programming model that allows to expose and manage data life cycle when the data sets are handled by heterogeneous systems and infrastructures.

Table 1.1 summarizes the involvement of students, postdocs and research engineers (F. Bérenger, J. Saray, and S. Bernard):

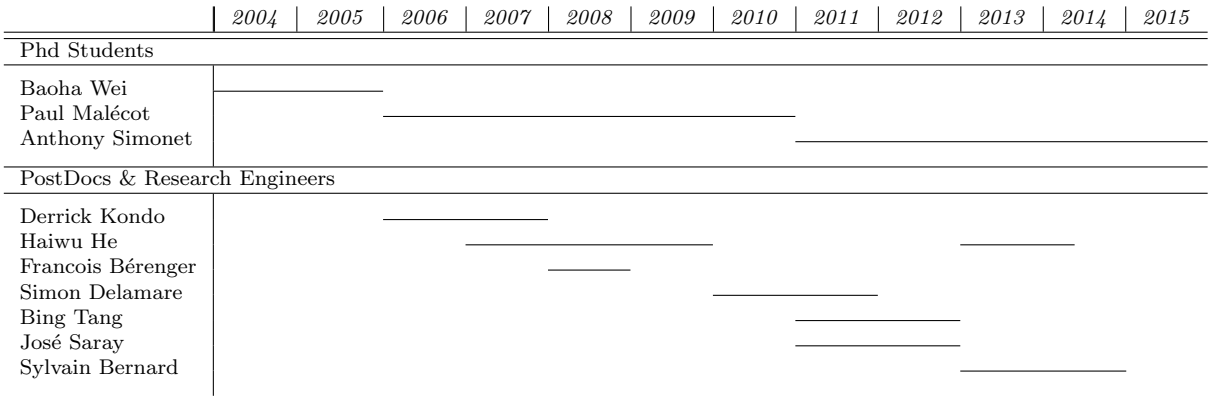


Table 1.1: Mentoring and advising

1.2.2 Experimentation and Development of Software and Toolkits

The methodology to explore and validate our approaches relies extensively on experiments (either in a controlled environment or using real world infrastructures), sometimes on simulations, and marginaly on analytical analysis. The difficulty to address the experimental evaluation of such environments leads us to develop a variety of original solutions, ranging from dedicated experimental platform (e.g DSLLAb) to complex emulation framework on Grid5000, the French experimental Grid [6].

To meet our research objectives, validate our approaches and perform large experiments, we have developed and contributed to the development of several software toolkits:

- XtremWeb-HEP is the result of technology transfer from the original XtremWeb code to the French Institute for Research in High Energy Physics (CNRS/IN2P3). Since 2002, Oleg Lodygensky leads the development of XtremWeb-HEP, with the objective of achieving a production and integration in the EGI Grid infrastructure, for which IN2P3 is one of the main actor. We continuously and closely cooperated not only on software development, but also to prototype new ideas, to validate innovative approaches developed at INRIA, and get feedback from real world use case, eventually leading to open new research directions. This collaboration allowed to address a very large range of scientific applications coming from physics, mathematics, finance, biology, and even multi-media.

- BitDew is the project umbrella under which several software and experiments have been developed to address the issues of data management and distribution for hybrid distributed infrastructures. BitDew is a subsystem, composed of a set of services, that 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. We started the development of BitDew in 2005. Since then, it has been used as a substrate by several PhD students and postdocs to conduct research on data-intensive computing: storage over hybrid infrastructures, MapReduce for Desktop Grid, data life cycle management (Active Data) and more.

1.3 Summary of the Document

The document is organized as follows.

- The Chapter II presents a State of the Art of Desktop Grid and related Distributed Computing Infrastructure technologies. An historical evolution of the technologies is done that covers the topics of resource management, scheduling algorithms, security principle, data management, software, standardization. The Chapter II positions our work with respect to related works and emerging challenges.

- The Chapter III presents our methodology for studying Desktop Grid Computing. The chapter covers our effort to observe and characterize existing Desktop Grid infrastructures, including Volunteer Computing Systems. We also report on our participation to the European Desktop Grid Infrastructure (EDGI), the first international collaboration aiming at providing a sustainable computing infrastructure based on Desktop Grid technologies.

- Chapter IV presents the algorithms and middleware we have developed to improve Desktop Grid in the context of Hybrid Distributed Computing Infrastructures. This chapter covers important areas such as security, virtualization, result checking, scheduling and Quality of Service.

- Chapter V focuses on Data Intense Computing on Desktop Grids and Hybrid Infrastructures. We'll describe the BitDew project as well as the associated developments: MapReduce on Desktop Grids, hybrid storage involving Desktop resources and Cloud storage, and Active Data, a programming model to program applications based on data life cycle across heterogeneous systems and infrastructures.

- Chapter VI presents conclusion and perspectives for this Habilitation thesis.

Chapter 2

Evolution of Desktop Grid Computing

> We described a computational model based upon the classic science-fiction film, *The Blob*: a program that started out running in one machine, but as its appetite for computing cycles grew, it could reach out, find unused machines, and grow to encompass those resources. In the middle of the night, such a program could mobilize hundreds of machines in one building; in the morning, as users reclaimed their machines, the "blob" would have to retreat in an orderly manner, gathering up the intermediate results of its computation. (This affinity for night-time exploration led one researcher to describe these as "vampire programs.")
>
> *(John F. Shoch and Jon A. Hupp, 1982)*

In this Chapter, we introduce the principles of Desktop Grid Computing, the main software realizations implementing this paradigm, and the characteristics of the infrastructures based on these systems. Over the last two decades cyber-infrastructures have lived many revolutions such as the elaboration of Grid production infrastructures and more recently the emergence of Cloud Computing. This Chapter positions our contributions with regard to these developments and outlines the singularities of our approach. We argue for a better integration of Desktop Grid infrastructures in eScience and we report on the requirements and challenges in terms of algorithms and technologies to meet this objective. We identify data management as being one of the key issues for Desktop Grid Computing and propose a new vision to tackle this difficult task. Finally we emphasize on the methodological challenge of conducting research in this field and argue for tools that not only improve the comprehension of existing systems but also allow experimentation on conditions close to the reality. Overall, this chapter aims at giving the reader a comprehension of the field before each of the points of our contribution are further detailed in the following chapters.

2.1 A Brief History of Desktop Grid and Volunteer Computing

2.1.1 Definition

The most widely accepted definition of Desktop Grid Computing is *the principle of using a network of Desktop PCs when they are idle to execute very large distributed applications.*

Taking this definition as a starting point, we can further develop the concept, depending on whether one considers the perspective of the infrastructure or the technology.

- If one takes the point of view of the infrastructure, then there is a set of PCs, distributed over the Internet or several local networks, that cooperate to run large applications. Of course this set of computing resources have their own characteristics, which are very different than computing resources involved in traditional DCIs. Without characterizing these resources extensively, one simply notes that they are numerous, prone to frequent failures, with low communication performance and with a low level of trust.

- The second aspect of the concept is the set of technologies which allows this kind of distributed infrastructure to run parallel applications. Those technologies encompass software, service and algorithms which implement scheduling, security, failure resilience, data management, programming model, and more.

Obviously, in parallel computing, architecture, runtime environments, and languages are strongly linked. Historically, emerging classes of parallel architectures have steered the development of parallel programming paradigms able to exploit the hardware efficiently. As an example, one could remember the development of vector supercomputers, such as Cray and Fujitsu, and the emergence of associated parallel data programming models. In the late 90's, our research team was involved in designing and assembling one of the first French cluster of multi-processors interconnected with a Myrinet Network, at the time where the idea of using off-the shelf components to build supercomputer was almost considered as a silly idea. Once the prototype built, the first challenge was to extend the message passing communication library to take advantage of the dual-processors motherboards. Interestingly, this first prototype and experience have opened many researches around hybrid message passing and shared memory programming model. In contrast, when doing our research about Desktop Grid, we had to follow a different development cycle [7, 8, 9]. We first designed and solved the runtime issues before building the infrastructure by deploying the software prototypes on the Desktop machines. Later, we were able to study the infrastructures, in particular by characterizing the computing resources, which in turn influenced the design of many Desktop Grid components such as the scheduler, or the result certification mechanism.

In the remaining of this manuscript, we name Desktop Grids both the infrastructures and the technologies, and we study this items independently going back and forth between this two notions.

2.1.2 The Origins

Computer enthusiasts often like to think that any outstanding innovation must, somehow, be born in the Xerox Palo Alto Research Center. It turns out that the very first paper, published in 1982, discussing a system similar to Desktop Grid comes from the PARC, which also confirms this cosmogonic belief. In [10], authors introduce the idea of Blob computing and Worm programs, whose principle is given in the chapter dictum. The paper presents several key ideas for distributing a computation over a network of computers: Self replication, migration, distributed coordination, etc.

But what has actually driven the development of Desktop Grids, came from the association of several key concepts: 1) cycle stealing; 2) computing over several administrative domains; and 3) the Master-Worker computing paradigm.

Desktop Grids inherit the principle of aggregating inexpensive, often already in place, resources, from past research in cycle stealing. Roughly speaking, cycle stealing consists in using the CPU's cycles of other computers. This concepts is particularly relevant when the target computers are idle. Mutka and al. demonstrated in 1987 that the CPU's of workstations are mostly unused [11], opening the opportunity for high demanding users to scavenge these cycles for their applications. Due to its high attractiveness, cycle stealing has been studied in many research projects like Condor [12], Glunix [13] and Mosix [14], to cite a few. In addition to the development of these computing environments, a lot of research has focused on theoretical aspects of cycle stealing [15].

Early cycle stealing systems where bounded to the limits of a single administrative domain. To harness more resources, techniques were proposed to cross the boundaries of administrative domains. In the early 90's, the WWW has become increasingly popular. Beside the publication usage, the idea of using the Web as a technology to build distributed computing systems became a reality by designing client/server application using HTTP technologies. A first approach was proposed by Web Computing projects, such as Javelin [16]. These projects have emerged with Java, taking benefit of the virtual machine properties: high portability across heterogeneous hardware and OS, large diffusion of virtual machine in Web browsers and a strong security model associated with bytecode execution. At the end of the 90's these projects have proved that DG was a successful proof of concept. They have paved the way for several research works in the fields of programming model, results certification and scheduling.

The Master-Worker paradigm is the third enabling concept of Desktop Grids. The concept of Master-Worker programming is quite old [17], but its application to large scale computing over many distributed resources has emerged few years before 2000 [18]. The Master-Worker programming approach essentially allows for the implementation of non trivial (bag of tasks) parallel applications on loosely coupled computing resources. Because it can be combined with simple fault detection and tolerance mechanisms, it fits extremely well with the Desktop Grid platforms that are very dynamic by essence. The combination of web technologies with the Master/Worker programing model has enabled the first generation of Desktop Grid systems.

The Great Internet Mersenne Prime Search (GIMPS) [19] is one of the oldest computation using resources provided by volunteer Desktop Grid users. It started in 1996 and

is still running. The 45th known Mersenne prime has been found in August 2008. Since 1997, Distributed.net [20] tries to solve cryptographic challenges. RC5 and several DES challenges have been solved. The first version of SETI@Home [21] has been released in may 1999. SETI is an acronym for the Search for Extra-Terrestrial Intelligence, whose purpose is to analyze radio signals collected from the Arecibo radio telescope. The data is digitized, stored, and split in work-units both by time (107s long) and frequency (10KHz) to search for any signals, that is, variations which cannot be ascribed to noise, and contain information. The crux of SETI@home is to have work-units distributed to a large base of home computers, so that data analysis is massively distributed. At the moment, projects such as SETI@Home have formed among the largest distributed systems in the world, involving millions of volunteers to reach huge computational power at the fraction of the cost of a traditional supercomputer.

However, this success has been obtained at the price of simplification as they were built around a single application and only the project administrator could use the computing power provided by the whole Desktop Grids.

2.1.3 Influence and Impact of P2P Computing

In the second half of the 90's, a new approach, called *peer-to-peer* (P2P) has revolutionized the design of distributed systems [22]. By contrast with the traditional Client/Server architecture, P2P systems can be summarized by three guidelines: *i*) the organization and the coordination of the system is totally decentralized avoiding the bottleneck and the single point of failure of the server; *ii*) the system relies on nodes which are located at the edge of the Internet instead of relying on servers, which are usually located on the Internet backbone; and *iii*) any nodes in the system can play at the same time the role of client and server. Driven by the objective of providing more scalable and reliable distributed systems, the literature has seen a flowering of papers proposing distributed indexing structures, such as Distributed Hash Tables (DHT). To name a few of the most successful propositions, we can cite Chord [23], Pastry [24], or Tapestry [25]. In parallel with these research works, many end-user applications, mainly dedicated to file-sharing have seen an increasing popularity. The principle was that all participants of the system were both able to propose a list of multi-media files to download, and to download the files from the other participants. These applications have shown that it was possible to build actual systems relying on P2P principles and those systems where able to implement scalable databases containing multi-billion content documents and to distribute very large files to large number of nodes [26]. In the beginning of the 2000's, soon after being made available to the Internet users, these systems where already consuming a significant fraction of the Internet bandwidth.

Strongly influenced by the P2P vision, Desktop Grid computing has followed a similar evolution. Indeed, the P2P systems have outlined the lack of genericity of the early distributed application, mainly built around a single application. Making Desktop Grid Systems more generic is one of the fundamental motivations of the second generation of Global Computing systems like BOINC [27] and XtremWeb [28]. These systems are designed so that various applications can be integrated, and several users or projects

can share the aggregated resources. The consequence is the establishment of a general architecture for Desktop Grid computing, which can be described as three components: the *client* that submits computing requests, the *worker* that accepts requests, performs the computation and returns the results, and the *coordinator* that schedules the client requests to the workers. Thus, depending on how the clients/workers/coordinators network is organized, one can build very different types of infrastructures.

2.1.4 Taxonomy of Desktop Grid Systems

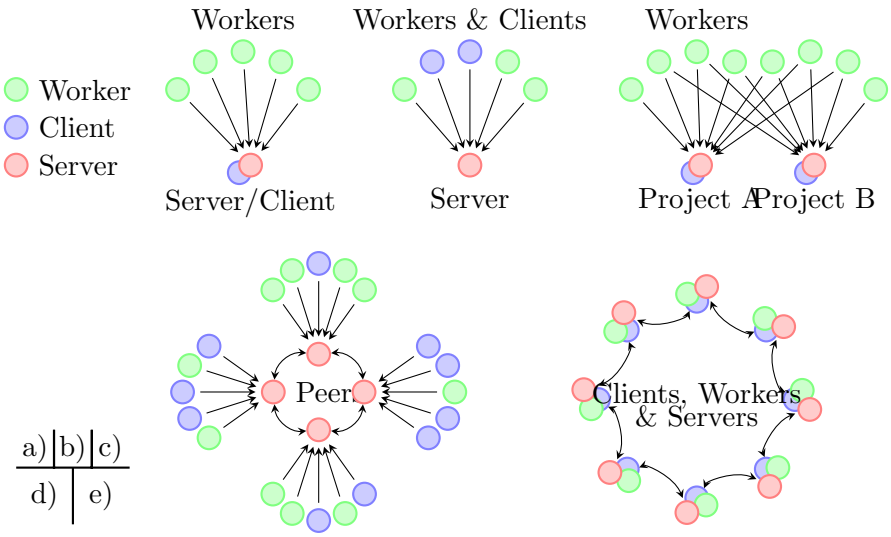


Figure 2.1: Different architectures of Desktop Grid project: a/ Distributed Application, b/ Global Desktop Grid, c/ Volunteer Computing, d/ Collaborative Desktop Grid, e/ P2P Desktop Grid

Desktop Grid systems can be classified according to their deployments and resource organization as shown in Figure 2.1:

Distributed Application (Fig. 2.1-a), corresponds to the early 90's first generation of Desktop Grid systems, including Web computing projects such as Jet [29], Charlotte [30], Javelin [16], Bayanihan [18], SuperWeb [31], ParaWeb [32] and PopCorn [33].

Internet Desktop Grid Fig. 2.1-b) and *Volunteer Desktop Grid* 2.1-c) correspond to the second generation of Desktop Grid systems; sometimes referenced as *Enterprise Desktop Grids* when the deployment consists of Desktop computers hosted within a corporation or University belonging to the same administrative domain. The Condor [12] system is a pioneering work in this domain, whereas several companies such as Entropia [34], United Devices, or Platform, have specifically designed products to implement this paradigm.

XtremWeb is an open source research project originally developed by the LRI and LAL (CNRS, Université Paris XI and LAL) that belongs to the class of Internet Desktop Grid systems [28, 35, 36, 3, 37]. The development started in 1999, and it was primary designed

in order to explore scientific issues about Desktop Grid, Global Computing and Peer to Peer distributed systems. Since then, it has been deployed over networks of common Desktop PCs and regular Grid nodes, providing an efficient and cost effective solution for a wide range of applicative domains: bioinformatics, molecular synthesis, high energy physics, numerical analysis and many more. XtremWeb-HEP [38], is the production version developed in the context of EGI Grid by Oleg Lodygensky and his team in the Laboratory of the Linear Accelerator (LAL/IN2P3/Université Paris Sud).

The Berkeley Open Infrastructure for Network Computing (BOINC) [27] is the main platform for Volunteer Computing. More than 900,000 users from nearly all countries participate with more than 1,300,000 computers, to more than 40 public projects. BOINC is an asymmetric organization, in the sense that a project is using resources from many resource volunteers. In addition, volunteers have the possibility to collaborate to several projects simultaneously. SZTAKI-DG [39] is a set of extension for BOINC to allow for a seamless and secure integration in Grid environments.

Collaborative Desktop Grids 2.1-d) consists of several Local Desktop Grids which agree to aggregate their resources for a common goal. The OurGrid project [40] proposes a mechanism for laboratories to put together their local Desktop Grids. When scientists need an extra computing power, this setup allows them to access easily their friend universities resources. In exchange, when their resources are idle, they can be given or rented to others universities. A similar approach has been proposed by the Condor team under the term "flocks of condor" [41].

2.2 Algorithms and Technologies Developed to Implement the Desktop Grid Concept

Over this past decade of development of the Desktop grid concept, an impressive set of technologies has been developed we are going to overview now.

2.2.1 Job and Resource Management

The functionalities required for job management include job submission, resource discovery, resource selection and binding. With respect to job submission, some systems have an interface similar to batch systems, mainly because it is well adapted to Bag-of-Tasks workload.

After a job is submitted to the system, the job management system must identify a set of available resources. This process is called Resource discovery. The classic method is via *matchmaking* [42] where application requirements are paired with compatible resource using a ClassAds description of the requirements.

Several distributed approaches to resource discovery have been proposed. The challenges of building a distributed resource discovery system are the overheads of distributing queries, guaranteeing that queries can be satisfied, being able to support a range of application constraints specifed through queries, and being able to handle dynamic loads on nodes. Several works [43, 44, 45] have investigated how to implemented distributed

resource discovery using a DHT in the context of a P2P system. In [46], the authors proposed a rendezvous-node tree (RNT) where load is balanced using random application asssignment. The RNT deals with load dynamics by conducting a random-walk (of limited length) after the mapping. In [47], the authors use a system where information is summarized hierarchically, and a bloom filter is used to reduce the overheads for storage and maintenance. And alternative to the DHT is to use a self-stabilized tree. In [48], a decentralized resource discovery using a self-stabilized tree have been proposed. BonjourGrid [49] is an interesting alternative to decentralized index protocols inspired by P2P network. As the name suggests, BonjourGrid leverages the Zero configuration networking and publish/subscribe protocol. Nodes advertize their role (i.e., worker/scheduler/client) using the Bonjour implementation, and the system can deploy on-the-fly instances of Desktop Grid middleware if it is observed that some roles are missing.

After a set of suitable resources have been determined, the management system must then select a subset of the resources and determine how to schedule tasks among the resources.

2.2.2 Scheduling

The majority of application models in Desktop Grid scheduling have focused on jobs requiring either high-throughput [50] or low latency [51, 52]. Some works have considered providing a response in a limited time [53, 54], others have considered guaranting some level of QoS [55]. These jobs are typically compute-intensive.

The pull nature of work distribution and the random behavior of resources in Desktop Grids introduce several limitations on scheduling possibilities. First, it makes advance planning difficult as resources may not be available for task execution at the scheduled time slot. Second, as task requests are typically handled in a centralized fashion and a server can handle a maximum of a few hundred connections, the choice of resources available is always a small subset of the whole. This platform model deviates significantly from traditional grid scheduling models [56, 57, 58, 1].

There are four complementary strategies for scheduling in desktop grid environments, namely resource selection, resource prioritization, task replication, and host availability prediction. In practice, these strategies are often combined in heuristics.

With respect to resource selection, hosts can be prioritized according to various static or dynamic criteria. Surprisingly, simple criteria such as clock rates has been shown to be effective with real-world traces [52]. Other studies [59, 50] have used probalistic techniques based on a host history of unavailability to distinguish more stable hosts from others.

With respect to resource exclusion, hosts can be excluded using various criteria, such as slowness (either due to failures, slow clock rates, or other host load), unreliability, or hosts which return erroneous or faked results. Thus, excluding those hosts from the entire resource pool can alleviate the performance bottleneck.

With respect to task replication, schedulers often replicate a fixed number of times. The authors in the studies [59] and [50] investigated the use of probabilistic methods for

varying the level of replication according to a host's volatility. When triggered at the end of the computation, replication can avoid the slowdown resulting from the host taking an unexpected amount of time to return their results. This strategy is somewhat similar to the speculative execution that we can find now in MapReduce runtime environments [5].

With respect to host availability prediction, the authors in [60] have shown that simple prediction methods (in particular a naive bayesian classifier) can allow one to give guarantees on host availability. In particular, in that study, the authors show how to predict that N hosts will be available for T time. In [61] classification methods are used to predict and avoid computing on likely to fail resources.

2.2.3 Volatility Resilience

Because nodes can join and leave the system at any time, Desktop Grid systems have been designed in such a way that resilience to volatility handles failures as a normal events whereas, in traditional systems, failures were considered as exceptional events which require special treatment to restore a correct state of the system.

Volatility detection is usually implemented following two approaches whether the system considers host or task failure. The first one, followed by systems such as XtremWeb [28] and Entropia [34], relies on heartbeats sent from the computing nodes, which periodically signal their activity to the server. BOINC [27] implements the second approach and uses job deadlines as a indication of whether the job has permanently failed or not. Unfortunately few works have investigated alternative failure detectors.

When a failure has been detected, one can resolve the failure in a number of ways. Task checkpointing is ones means of dealing with task failures since the task state can be stored periodically either on the local disk or on a remote checkpointing server; in the event that a failure occurs, the application can be restarted from the last checkpoint. In combination with checkpointing, process migration can be used to deal with CPU unavailability or when a "better" host becomes available by moving the process to another machine.

As a consequence, several distributed checkpointing systems have been designed to schedule and store the execution snapshots. The authors in [62, 63] develop a distributed checkpoint system where checkpoints are stored in peers in a P2P fashion using a DHT, or using a clique. StdChkpt [64] is a checkpoint storage system that gathers local storage desktops. StdChkpt is specialized in many ways for managing of checkpoint images: handling of write series for high-speed I/O, support for data reliability as well as versioning, incremental checkpointing, and lifetime management of checkpoint images. [65] proposes to cluster nodes in a hierarchical topology and uses replication to improve checkpoint images reliability and performance when getting checkpoint images. An idea proposed in [66] is to compute a signature of checkpoint images and use signature comparison to eliminate diverging execution. Thus, indexing data with their checksum as commonly done by DHT and P2P software permits a tolerance to basic sabotage even without having to retrieve the data.

Another approach for masking failures is replication. The authors in [67, 59, 50] use probabilistic models to analyze various replication issues. The question of failure is even

more important when executing parallel applications, as a single node failure may slow down the whole parallel execution. Several works have investigated these issues with various platform models and tightly-coupled or loosely coupled applications. [67] and [68] examine analytically the costs of executing task parallel applications in desktop grid environments. In this context, understanding and modeling the availability of a collective group of resources is of great importance [69]. This research axis is still a hot topic in the community.

2.2.4 Data Management

Although Desktop Grids have initially been built for high throughput computing, since the middle of the 2000's, a large research effort has been done to support *data-intensive* applications in this context of massively distributed, volatile, heterogeneous, and network-limited resources. Data-intensive applications require secure and coordinated accesses to large datasets, wide-area transfers, and broad distribution of TeraBytes of data while keeping track of multiple data replicas.

Most Desktop Grid systems, like BOINC, XtremWeb, Condor, and OurGrid rely on a centralized architecture for indexing and distributing the data, and thus potentially face scalability and fault tolerance issues. However, researches around DHTs [23, 70, 24], collaborative data distribution [26, 71, 72], storage over volatile resources [73, 74, 75], and wide-area network storage [76, 77] offer various tools that can be leveraged to circumvent entirely or partly this bottleneck.

Parameter-sweep applications composed of a large set of independent tasks sharing large amounts of data are the first class of applications which has driven a lot of efforts in the area of data distribution. In the initial works exploring this idea [78], we have shown that using a collaborative data distribution protocol such as BitTorrent over FTP can improve the execution time of parameter-sweep applications. In contrast, it has also been observed that the BitTorrent protocol suffers a higher overhead compared to FTP when transferring small files. Thus, one must be allowed to select the correct distribution protocol according to the size of the files and the level of "sharability" of data among the task inputs. Later, similar studies have been conducted with the BOINC middleware and led to similar conclusions [79, 80, 81, 82, 83]. If the P2P approach seems efficient to distribute large data, it assumes that volunteers would agree that their PC connects directly to another participant's machine to exchange data. Unfortunately, this could be seen as a potential security issue and is unlikely to be widely accepted by users. This drawback has so far prevented adoption of P2P protocols by major volunteer computing projects.

The alternative approach to Bittorrent is to use a content delivery approach where files are distributed by a secure network of well-known and authenticated volunteers [84, 85]. This approach is followed by the ADICS project [86] (Peer-to-Peer Arhictecture for Data-Intensive Cycle Sharing). Instead of retrieving files from a centralized server, workers get their input data from a network of identified cache peers organized in a P2P ring.

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 the local storage of the Desktop PC [87, 88, 75] with adequate scheduling strategies [89, 78] to minimize data transfers can thus improve overall application execution performance.

2.2.5 Security Model

A key point is the asymmetry of Volunteer Computing security model: there are few projects well identified and which belong to established institutions (by example, University of Berkeley for the SETI@Home project) while volunteers are numerous and anonymous. The notion of users exists in BOINC. It allows users to participate to forum and receive credits according to the computing time and power given to the project. The security mechanism used to authenticate the project and its applications is simple and based on asymmetric cryptography. If volunteers trust the projects, the reverse is not true. To protect against malicious users, result certification mechanism [90] is needed to ensure that results are not tampered by malicious users. Result certification has been subject of many researches that have investigated several combination of strategies such as black-listing, replication, vote or probabilistic spotting [91, 92, 93, 94, 95, 96, 97, 98]. More recently, these works have addressed the issues of nodes that would cooperate to launch attacks [99, 54, 100].

The second aspect of security is the protection of the computing resources which must be insured in the case where any user has the possibility to submit his/her own applications and input data. This protection is known as *sandboxing*. It consists either in confining the execution the program, so that attempts to corrupt the host system can be detected and corrective actions can be taken to stop, isolate, or modify the execution, so that attacks would be directed to a system different than the host system. If early works have relied on specific solutions, such as kernel security extensions [3], this technique has become implementable since the development of Virtual Machine (VM) technologies, which are widely available both in the server market (XEN) and the Desktop market (VMWare, Virtual Box). VMs offer many desirable features such as security, ease of management, OS customization, performance isolation, checkpointing, and migration to improve security and deployability.

There are several ongoing works that aim at bringing virtualization technologies to Desktop Grid. Daniel L.G et al. [101] present a method to run legacy applications on BOINC using the standard BOINC Wrapper and a special starter application to set up the environment for the application. The capability of VMs to save and resume their state image is used to provide a checkpoint/restart mechanism. LHC@Home [102] chooses to use virtualization to increase the portability of the Atlas [103] physics application. Atlas requires the Athena framework which is around 8GB and it is closely tied to a specific Linux distribution. LHC@Home provides VMware images with Linux and the whole software stack needed to run Atlas plus a BOINC client executed within the virtual machine.

2.3 Evolution of the Distributed Computing Infrastructures Landscape

The landscape of distributed computing has known several radical evolutions during the past two decades, in particular with the rise of Grid and Cloud systems [1, 104]. Somehow, Grid, Cloud, and Desktop Grid systems share the same concept of using *on-demand* remote computing facility. They ultimately aim at allowing for a "flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources" [105]. However, at the beginning of their existence, Desktop Grid systems and in particular Volunteer Computing systems were adopting rather radical principles that were far from other Grid or Cloud systems, which has made them to evolve autonomously. In this Section, we situate Desktop Grid Computing in the context of the recent evolution of Distributed Computing Infrastructures and while explaining the differences and contrasts, we give reasons and trends of their convergence.

2.3.1 From Grid Computing...

The history of Grid Computing starts with the availability of wide-area high speed networks, which has allowed to integrate computing resources from different sites. Metacomputing was the name given to the computing model aiming at running parallel applications using several super-computers. Amongst the first experiments, I-Way [106] was the first large project to establish the model and gave a basis for both the main concepts and problematics as well as a the middleware and infrastructure foundation.

Grid Computing has since then been developed following four directions: infrastructure, virtual organizations, software, and standards.

One key scientific communities which has pushed forwards Grid Computing is the High Energy Physics and in particular the international collaboration around the Large Hadron Collider (LHC). The LHC experiments were anticipated to generate yearly petabytes of data, and only a large worldwide infrastructure, federating storage and computing resources from hundreds of sites would be adequate to allow thousands of scientists to collectively analyze these data sets. This has lead to the creation of the Data Grid [107] project in Europe, and the Grid Physics Network [108] in the US, which later continued their existence as the Enabling Grids for E-sciencE (EGEE), followed by the European Grid Infrastructure (EGI) and the Open Science Grid, and the LHC Computing Grid (LCG) [109]. Similar infrastructures, but geared towards different scientific communities or built on top of different regional collaborations, have continuously spread worldwide. One can cite for example: ChinaGrid [110] in China, NorduGrid [111] in the northern european countries, TeraGrid [112] in the US, EEULA in South America, Naregi [113] in Japan to name a few of them. Some of these infrastructures have been set up to coordinate access to high-end supercomputers, such as Deisa [114] while others, such as Grid5000 [6] and FutureGrid [115] have been designed to explore, develop and study very large distributed systems.

Alongside with infrastructures, *virtual organization* (VO) has been the cornerstone concept to drive the organization of the Grids. The term VO denotes a set of individuals

or institutions that agrees to federate their resources and on a set of rules that define "clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs" [105].

Supporting these large and distributed infrastructures and VOs has required complex software stacks to enable resource management, running application across multiple sites, users authentication and authorization, resource access logging and bookkeeping, organizing data movement, storage and replication, ... The Globus Toolkit [116], is a popular collection of software components, which have been enhanced and extended by several other Grid toolkits such as ARC [117] or gLite [118]. In these toolkits, security is usually achieved through X.509 proxy certificates [119] and VO manager VOMS [120], GridFTP [121] implements high performance data management service, and job submission and management relies on GRAM [116] or CREAM [122]. In contrast with Grid toolkits, Unicore [123], XtreemOS[124] and DIET [125] are integrated Grid middleware.

It is eventually the need for interoperability between Grid middleware that has pushed for a global standardization process. The Global Grid Forum, and later the Open Grid Forum, have been established to foster cooperation and agreement on document defining the standards. The most noticeable results have been the definition of X.509 certificates that underpins the Grid Security Infrastructure [126], the job description languages (JDL and JSDL [127]), the Basic Execution Service (BES) [128] and the DRMAA [129] specification for resource management. Eventually, this standardization effort has facilitated interoperability between software by the emergence of portable multi-software toolkits, such as CoG [130] or SAGA [131] and Grid applications portals such as P-Grade [132] or GridPort [133].

2.3.2 ... To Cloud Computing

Recently, with the emergence of Cloud computing platform, a new vision of distributed computing infrastructures has appeared where the complexity of an IT infrastructure is completely hidden from its users. Cloud Computing provides access through web services to commercial high-performance computing and storage infrastructure (*IaaS* – Infrastructure-as-a-Service), distributed services architecture (*PaaS* – Platform-as-a-Service) or even application and software (*SaaS* – Software-as-a-Service). This vision is achieved through the extensive use of virtualization technologies which allow users to deploy and manage virtual images of their computing environment directly on the Cloud.

A Cloud computing platform can dynamically configure, reconfigure, purvey, and deprive computing resources. The interfaces to access resources, provided by Amazon Web Services have emerged as the *de facto* standards and most IaaS vendors sells compatible platforms. Eucalyptus [134], OpenStack [135], and OpenNebula [136] are Cloud computing software based on web services.

The economical model asssociated with Cloud Computing makes this approach competitive for scientific communities compared with traditional Grid computing. Compute resource consumers can eliminate the expense inherent in acquiring, managing, and op-

erating IT infrastructure and instead lease resources on a pay-as-you-go basis. Some studies also compared Clouds and Desktop Grids, and also conclude to the attractiveness of Cloud for small scale projects [137]. In [138], authors investigate the viability of using Amazon S3 storage service for science.

As a consequence, Cloud Computing becomes more and more attractive, which push forwards the design of a Cloud framework dedicated to scientific usages. The Nimbus [139] Cloud computing platform is an early representative of such trends. Following a complementary approach, StratusLab [140] is a project of a distributed Cloud for scientific simulations which can be used by the Grid communities. ElasticSite [141] is a project which enables to supplement Globus infrastructures with EC2 Cloud resources to meet user peak demands. They investigate several strategies to decide when to provision such additional Cloud resources.

Desktop Clouds, or Clouds made of Desktop PCs is a new and on-going research direction that combines virtualization and Desktop Grid technologies. This consists in building virtual cluster such as Violin [142], WoW [143] or PVC [144] on top of volunteers PCs and which could be deeply configured on demand by DG users. The expected benefit is that a much broader range of applications could run on Desktop Grids. Furthermore DG users would have the ability to tune the OS of their own VM images and deploy their prefered set of services (e.g., scheduler, file system, monitoring infrastructure) along with the application. Desktop Clouds raise many research challenges: deployment of VM images on DG resources, scheduling heuristics allowing for the reservation of DG resources, establishment of virtual cluster despite the network protection of firewalls and NAT, transparent checkpoint/restart of networked VM and, finally replication of communicating VM to ensure a better reliability.

2.4 Emerging Challenges of Desktop Grid Computing

In the previous section, we have seen how the research efforts have led to substantial developments and innovations. However, considering actual large scale applications, one can observe that, in practice, Desktop Grid systems have little diverged from the original concepts. Although the most popular deployments have strengthened the model, showing that it was indeed able to provide effective computing infrastructures at a very competitive cost, the model was still restricted to a limited class of applications, namely Bag-of-Tasks applications with very few I/Os. More detrimental, none or very few of the existing Desktop Grid systems that have been largely deployed have been able to engage a broad base of scientific users, such as the Grid systems have been doing for years. In this section, we review some the main challenges Desktop Grid have to address before being a first-class citizen in the e-science infrastructure landscape.

2.4.1 Broadening the Application Scope of Desktop Grids

The first research challenge is to broaden the applicability of Desktop Grid systems to applications that require more coordination, storage and better quality of service. Figure 2.2 presents a simplified, incomplete, arbitrary, and somewhat naive view of the main

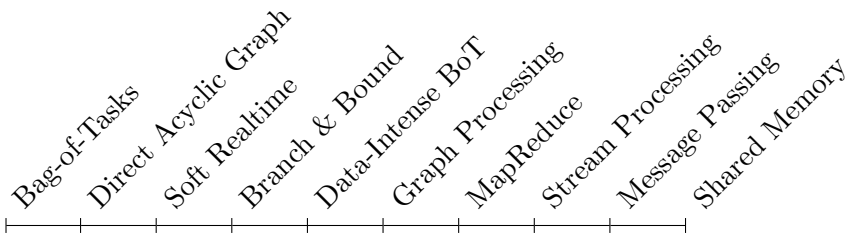


Figure 2.2: Parallel Applications Complexity Scale

typology of parallel applications. Nevertheless, it will help us to figure out the effort needed to fulfill our objectives. The gradient view orders parallel application classes according to their execution requirements and complexity. On the left, we find the Bag-of-Tasks applications, which constitute a large part of nowadays Grid applications and the main type of applications supported by Desktop Grid. The more we move to the right of the Figure, the more applications require stronger coordination, such as control between a master and its workers, managing dependencies between the tasks, steering iterative and branch and bound computations or enforcing deadline constraint for soft real-time applications. Data-intense BoT requires the ability to distribute efficiently large amounts of data, usually using P2P protocols. On the rightmost part, we find first MapReduce computations which add collective communications, and storage and then we find under the term MPI applications, the SPMD applications which rely on the message passing communication paradigm. All along the past decade, many research advances, on the front of algorithms (e.g., scheduling, programing model) and middleware (e.g., data management, communication libraries, QoS services) have allowed to move the cursor to the right. However, there are still intrinsic limitations. The network capacity of Desktop Grids is orders of magnitude less efficient than super-computers. The communication links of broadband Internet are asymmetric and usually do not allow for direct connections between peers. The result of this gap is that while knowledge has greatly progressed, we have not seen actual Desktop Grid infrastructures supporting advanced parallel applications. In my research activities, I focused mainly on two classes of applications at both ends of the spectrum: (*i*) Bag-of-tasks applications, because already highly prevalent, progress in this area will have an immediate impact on the performance and efficiency of existing infrastructures, and (*ii*) the MapReduce programming model, because of the potential it could open to the DG to perform data-intensive applications.

The second factor to greater acceptance of Desktop Grid systems is the ease to program and port applications to these infrastructures. Developed in the context of the European EDeGS/EDGI projects, the EDGeS Application Development Methodology (EADM [145]), is a framework offering a clear methodology to help the porting of applications. EADM consists of several stages. The first one is the identification of potential applications that could benefit from the EDGeS/EDGI infrastructure and consists of analyzing the candidate applications in terms of type of parallelism and computing platform used before the platform, data access and volume, licencing, programming lan-

guage, execution environment, computational load, type of user interface, information confidentiality and security requirements, etc. The next stages concern application design and development and ensure that an optimum service is provided to the end-user, within the existing technical constraints and limitations of the platform. In addition, the DCI-API [39] is a programming library which helps the programmers to develop their applications for the EDGI infrastructure by addressing the portability across the various DG middleware. I always considered that the use of virtual machines can greatly improve the portability of application [146, 147], and I still conduct several researches in this area. Flying Grids [148] and GBAC [149] are two frameworks based on virtualization technologies to ease the porting of complex applications to XtremWeb-HEP and BOINC respectively. The first results with Flying Grid indicate that it is now possible to run complex applications such as High Energy Physics (HEP) ones on non specialized and/or dedicated ressources. First, those HEP applications rely on complex and software libraries such as Geant4[1] and ROOT[2], and second, may be especially designed to run on Scientific Linux OS. The capability of deploying and using specific VM such as the CernVM, allows us to target major HEP applications and software stacks such as SuperNemo[3] and Atlas[4].

2.4.2 Integration of Desktop Grid Computing in the Cyber Infrastructure

Desktop Grid technologies are now considered as a viable solution tocomplement regular Grid infrastructures. For instance, the European Union, through several FP7 projects (EDGeS, EDGI, DIGISCO) [150], supports a large research and development effort to make Desktop Grids easily available to the scientific public and in particular through the EGI community. Several Memorandum of Understanding have been signed between EDGI, the European Middleware Initiative (EMI), and the European Grid Infrastructure (EGI) to sustain the development of Desktop Grid technologies and bridge middleware.

Because these DCIs have different characteristics, there are many scenarios which advocate for hybrid infrastructures which assemble Clouds, Desktop Grids, and Grids to mitigate the disadvantages of certain aspects of some infrastructure and enjoy the benefits of others. Actually, it is reasonable to think that such hybrid infratructure could become a next trend in distributed computing, as an extension of the concept of *Sky Computing*, introduced by Keahey et al. in [141] to denote an infrastructure composed of multiple Clouds.

Desktop Grid middleware plays an "enabler" role in hybrid DCI, because they have several desired features to manage resources: resilience to node failures, no reconfiguration when new nodes are added to the system, task replication or task rescheduling in case of node failures, and push/pull protocols that help with deployment issues. Although there are many solutions for BoT execution on cross-infrastructure deployments, we have found that in several cases, a Desktop Grid middleware is used to schedule tasks

[1] `http://geant4.fnal.gov`
[2] `http://root.cern.ch/drupal`
[3] `http://nemo.in2p3.fr/supernemo`
[4] `http://atlas.web.cern.ch/Atlas/Collaboration`

on the computing resources. For instance, early in the development of XtremWeb, we have used the prototype to schedule jobs on several Condor pools [151] using a mechanism known as PilotJobs. The XtremWeb worker is scheduled as a regular Job on the Condor pool. Once the XtremWeb worker is executed on the Condor resource, it retrieves jobs from the server, execute the job and send results to the XtremWeb server.

The first motivation for assembling DCIs is to obtain greater computing power. In this configuration, Desktop Grids can play a supplementary role for Grid users by offering a vast amount of computing power for a little additional cost. Unsurprisingly, several projects have created bridge technologies which allow Grid users to use resources provided by Desktop Grids. GridBot [152] puts together Superlink@Technion, Condor pools, and Grid resources to execute both throughput and fast-turnaround oriented BoTs. The Desktop Grid middleware used is BOINC augmented with the matchmaking mechanism of Condor to implement more sophisticated scheduling policies. Following a similar path, the Latin America EELA-2 Grid has been bridged with the OurGrid infrastructure [153].

In the scope of the European FP7 projects EDGeS (Enabling Desktop Grids for E-science) [150] and EDGI (European Desktop Grid Infrastructure), we developed the bridge technologies to make BOINC [154] and XtremWeb [155] transparently available to any EGI Grid users as a regular Computing Element. The cornerstone of the EDGeS and EDGI projects is the 3G bridge software which implements bi-directional jobs transmissions between Service Grids and Desktop Grids [156] (see Section 3.4).

Then, at the end of the EDGI project, I was convinced that Desktop Grid middleware could be an excellent scheduler capable of efficiently using hybrid DCIs at the condition that specific scheduling heuristics would take into account the differences between the infrastructures. This is the goal of the *Promethee scheduler* [157], presented in Section 4.3, which allows for multi-criteria and satisfaction oriented scheduling for hybrid DCIs.

The next scenario, known as *Cloud bursting* [141], is a mechanism which offload part of the Grid workload to the Cloud when there is a peak demand. It is noteworthy that several studies have compared the cost of running large scientific on the Clouds. In [138], authors investigate the cost and performance of running a Grid workload on Amazon EC2 Cloud. Similarly, in [137], the authors introduce a cost-benefit analysis to compare Desktop Grids and Amazon EC2. In [158], authors propose a Pareto efficient strategy to offload Grid workload, which consists of Bag-of-Tasks application whith deadlines on the Cloud.

However, because Desktop Grids trade reliability for lower prices, they offer poor Quality of Service (QoS) with respect to traditional DCIs [159]. Besides Desktop Grid, other particular usages of an existing infrastructure can also provide unused computing resources without any guarantee that the computing resources remain available to the user during the complete application execution. For example, the Grid resource managers such as OAR [160] manage a best effort queue to harvest the idle nodes of a cluster. Tasks submitted in the best effort queue have the lowest priority. At any moment, a regular task can steal the node and abort the on-going best effort task. In Cloud computing, Amazon has recently introduced EC2 Spot instances [161] where users can bid for unused Amazon EC2 instances. If the market Spot price goes under the user's bid,

a user gains access to available instances. Conversely when the Spot price exceeds his bid, the instance is terminated without notice. Similar features exist in other Cloud services [162] as well. In Section 4.2, we will present SpeQuloS, a framework which provides QoS for applications executed on Desktop Grids by offloading a part of the application execution on reliable resources provided by Clouds.

2.4.3 Data Desktop Grid

Enabling e-Science has been one of the fundamental objective pursued by the computational science community. This effort aims at allowing large communities of researchers, which collaboratively 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 datasets, wide-area transfers and broad distribution of TeraBytes of data while keeping track of multiple data replicas. Despite the attractiveness of Desktop Grid 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 (eg. BOINC, XtremWeb, OurGrid or Condor) rely on a centralized architecture for indexing and distributing the data, and thus potentially face issues with scalability and fault tolerance.

As we have seen in Section 2.2.4, P2P protocols can be efficient at distributing large data to large number of nodes. Experiments on Grid5000 confirmed that the BitTorrent was protocol was significantly improving Desktop Grid performances when executing Data-intensive BoT [79, 163, 164, 165, 78] .

However, at the time of those experiments, the P2P protocols were only able to cope with a one-to-many file distribution pattern. This limitation convinced me that it was necessary to dispose of high level abstractions to precisely steer the distribution of data, which would allow the execution of more advanced parallel computations. After conducting a deep analysis of application requirements in terms of data management, I proposed the BitDew middleware, presented in Section 5.1. BitDew relies on a set of services to implement data scheduling, indexing, transfer, and storage.

Innovative in several ways, BitDew promotes the idea of using meta-data (called *attributes*), so that users can control data distribution, resilience, and life time. Once, the behavior of a data item has been set, the runtime environment implements the necessary actions to manage the data accordingly. Using these attributes allows the user to implement complex data management, such as resilient collective file operation that were infeasible before. In the meantime, several services for storage and data management appeared that address the issue of data management on multiple infrastructures. For instance, MetaCDN [166], proposes to unify several Cloud storage providers in a single namespace and provides smart placement of users' content based on QoS, coverage, and budget requirements.

At the same time, the challenge of executing data-intensive applications on Desktop Grid became a major concern in the community. In particular, the MapReduce programming model [5] started to attract a lot of attention [167, 168, 169, 170, 171]. Supporting

MapReduce on Desktop Grid [172] would allow to take advantage of storage resources of volunteer PCs to run application that process very large amounts of data, opening the way to large and scalable Big Data processing on Desktop Grids.

Leveraging BitDew, we proposed the first implementation of MapReduce for Internet Desktop Grid computing [173, 174, 175], which relies on a set of optimizations dedicated to this kind of platforms: overlap of Map and Reduce computations with data transfers, distributed results checking, collective file transfers and more. Moreover this early research has opened the way to MapReduce on hybrid DCIs [176, 171, 177, 178, 179], where the long term objective is to take advantage of multiple data storage services, each of them having different characteristics in terms of reliability, cost, and performance. We can then design storage strategies which offer the desired level of reliability, durability, and cost, and the system will coordinate the different services to transparently implement the policy [180].

2.5 Conclusion

In this Chapter, we have reviewed the evolution of the Desktop Grid concept following a chronological and thematic order. This encompasses several topics around algorithms and technologies, such as: scheduling and resource management, fault tolerance, security, and data management. We have seen that it is a rich area, where many innovations have influenced all distributed systems.

This Chapter outlined only some of the most interesting aspects of Desktop Grid Computing. Of course, it would not be possible in this Habilitation thesis to give an extensive chronology of the domain, neither to cover all the research topics and results. I would rather invite the reader wishing to deepen their knowledge of the domain to refer to the book "Desktop Grid Computing" at CRC Press, co-edited by Christophe Cérin and myself [2].

High performance and distributed computing is a fast evolving domain, which has radically changed over the last decade. We also introduced other Distributed Computing Infrastructures, in particular Grid and Cloud Computing. Detailing out these different paradigms has allowed us to understand the differences as well as the possible convergences. We observed a remarkable trend, which is the emergence of Hybrid DCI, i.e., the assemblage of Grid, Cloud, and Desktop Grid in a single infrastructure.

Several evolutions in DCIs may pose significant challenges to Desktop Grid Computing. The first challenge is the switch towards data oriented science, and the advent of data-intensive applications, which requires to re-think Desktop Grid architectures. The second challenge is the integration of Desktop Grid computing in the e-science cyber-infrastructures, which requires considerable technological and algorithmic advances.

However, to address these complex issues, one needs a new methodology that allows to precisely understand the characteristics of Desktop Grid platforms and that allows to test, develop, and evaluate experimentally our propositions. This is the subject of the next Chapter.

Chapter 3

Research Methodologies for Desktop Grid Computing

> The best way to observe a fish is to become a fish
>
> *(Jacques Cousteau (1910-1997))*

In this Chapter we explore the various methodologies at our disposal to study Desktop Grid Computing. We start with the observation of existing infrastructures in order to characterize computing resources. Simulation is a classical way to investigate distributed systems. We report on several solutions for simulating and emulating a Desktop Grid on the Grid5000 experimental platform. Then we present the DSL-Lab experimental platform that will allow us to perform experiments on the DSL broadband Internet using real software. Finally, the European Desktop Grid Infrastructure (EDGI) is introduced, which is a unique effort to establish an international computing infrastructures based on Desktop Grid technologies.

3.1 Observing and Characterizing Desktop Grid Infrastructures

Observing existing Desktop Grid systems has been the primary methodological approach for characterizing these systems. Although we know that these projects have attained considerable computing power for high-throughput applications, we still need to understand what would be their potential applicability to more demanding applications in terms of storage or communication.

3.1.1 Observing a Volunteer Computing platform: SETI@Home

In collaboration with D. Anderson from the University of Berkeley, we have studied the computing resources provided by the participants to the SETI@home project [181]. SETI@Home database collects a great number of host information, which can either report about the hardware (e.g. CPU, RAM, used and free storage, host location),

about the performance (e.g CPU Whestone and Dhrystone benchmarks, average network throughput) and about users' preferences (preferred applications, resource sharing, period of availability). When we started this research, we knew very few about host availability characterization. Fortunately, SETI@Home contains several relevant information: *i*) *host lifetime* is the interval from creation to last communication for hosts that had not communicated in at least one month, *ii*) *on-fraction* is the fraction of real time during which the BOINC client is running on the host, *iii*) *connected-fraction* is the fraction (of the time that BOINC is running) that a physical network connection exists, *iv*) *active-fraction* corresponds to when the host is allowed to compute and communicate, for instance because no mouse/keyboard activity has been detected, and finally *v*) *CPU efficiency* measures the percentage of CPU load, which has been allocated to execute a BOINC application. Clearly there are several levels of availability which are somehow nested: the host has to be turned on, then connected, then active to be able to join a computation. Combining the various factors, and assuming that the factors are statistically independent, we have the following expression for the total computing power X available to a project:

$$X = X_{arrival}.X_{life}.X_{ncpus}.X_{flops}.X_{eff}.X_{onfrac}.X_{active}.X_{redundancy}.X_{share}$$

Where $X_{arrival}$ is the average arrival rate of hosts, X_{life} is the average lifetime of hosts, X_{ncpus} is the average number of CPUs per host, X_{flops} is the average FLOPS per CPU, X_{eff} is the average CPU efficiency, X_{onfrac} is the average on-fraction, X_{active} is the average active-fraction, $X_{redundancy}$ is the reciprocal of the average redundancy, and X_{share} is the average resource share (relative to other CPU-intensive projects). Although this work presents several limitations, in particular regarding the modeling of each of this variable, this was a first step towards understanding the components on which depends performance of Desktop Grid systems.

3.1.2 Characterizing Host Availability

As we can see from the previously presented formula, one of the key aspect impacting performance is host availability. In the previous work, we obtained aggregate measurements in time, which did not allow to understand host availability variability during time. During my postdoctoral stay in San Diego, I collaborated with Derrick Kondo who proposed a new methodology for availabilities studies [182]. It consists in gathering traces by submitting measurement tasks to a Desktop Grid system that are perceived and executed as real tasks. These tasks perform computation and periodically write their computation rates to file. This method requires that no other Desktop Grid application is running, and allows us to measure exactly the compute power that a real, compute-bound application would be able to exploit. Our measurement technique differs from previously used methods in that the measurement tasks consume the CPU cycles as a real application would, thus measuring both host and CPU availability. Also, this approach captures all the various causes of task failures, including but not limited to mouse/keyboard activity, operating system and hardware failures, and the resulting trace reflects the temporal structure of availability intervals caused by these failures.

Moreover, the method takes into account overhead, limitations, and policies of accessing the resources via the Desktop Grid infrastructure. Together, we gathered traces from four different Enterprise Desktop Grid deployments relying on the Entropia and XtremWeb systems [183, 184].

3.1.3 The XtremLab Project

Later, following my return to France as a newly appointed INRIA research, I started the DSLLAB project supported by a ANR Young Researcher Grant. The objectives were to provide accurate and customized measures of availability, activity and performances in order to characterize and tune the models of the ADSL resources and to provide a validation and experimental tool for new protocols, services and simulators and emulators for these systems. Derrik Kondo and Paul Malécot joined the project respectively as Postdoc and PhD student, with the task of setting up XtremLab: a volunteer project based on BOINC in order to obtain traces from an Internet Desktop Grids [185].

The findings of our characterization studies cannot be summarized easily. We considerably increased our knowledge about those platforms by allowing to get new insights about correlation between tasks failure rates, host and user profile, machine activity and availability, as well as determining sources of correlated failures. This knowledge in turn, allowed us to improve scheduling and fault tolerance algorithms in terms of optimal tasks duration, optimal checkpointing interval during tasks execution, and building improved performance model that takes into account host availability, resource selection based on host characteristics and many more.

To highlight one particular result, which received the Best Paper Award at the Europar conference in 2007, [185] has been authored in collaboration with Filipe Araujo, Patricio Domingues and Luis Moura Silva from the University of Coimbra, Portugal. One critical issue of Desktop Grid is the errors in results, which are almost inevitable because of the high number of unreliable nodes involved. Errors can stem from different sources, such as, to give few examples: computational, an error could result from a CPU miscalculation due to overclocking and overheating; related to failures during application input or output (I/O), if a machine crashes when the application is writing to an output file or checkpoint; or sabotage, if computed by a malicious host. To study error rates, we analyzed the XtremLab computation results to give quantitative and empirical characterization of errors stemming from input or output (I/O) failures. We find that in practice, error rates are widespread across hosts but occur relatively infrequently: about 35% of hosts will commit at least a single error over time and the mean error rate over all hosts is 0.0022. A large fraction (e.g. about 70%) of errors result from a small fraction (e.g. 10%) of hosts. Moreover, we find that error rates tend to not be stationary over time nor correlated between hosts.

In light of these characterization findings, we evaluated and compared the effectiveness of several error prevention and detection mechanisms namely blacklisting, majority voting, spot-checking, and credibility-based methods. We concluded that when one requires a small error rate (less then $2\text{x}10^{-4}$) and can afford high redundancy, then majority voting should be considered. For greater error rate then spot-checking with

blacklisting should be strongly considered, as long as workload can be formed as relatively long batches. Fluctuations in error rates over time may limit the effectiveness of credibility-based systems.

Beside direct analyze, such traces of Desktop Grid system can be used for a broad range of studies: feeding trace-driven distributed system simulator, statistical generative availability model, and predictive model for availability, host clustering according to collective availability, and more. Thus, those traces could benefit the whole community and we made them available to the general public through the Desktop Grid Trace Archive [186]. Later, when Derrick Kondo joined INRIA, he greatly developed this research direction, in particular by initiating the Failure Trace Archive[4] project.

3.2 Simulating and Emulating Desktop Grid Systems

For researchers in large scale distributed systems, simulation remains one the fundamental methodological approach. There are several unavoidable reasons which motivate the development of Desktop Grid simulators. First, it allows to quickly obtain results on the hypothesis investigated in the simulation. Second, the environment for the experiment is reproducible and can be fine-tuned, for instance to create special condition corresponding to specific scenarios. In particular, a requirement is to simulate very large platforms with very large number of nodes. Last, it allows to explore large set of parameters, which helps to understand the interactions between several mechanisms and determine the best combination of parameters. In our research, we have never relied solely on simulations for performance. In contrast, simulations are part of a toolbox, mainly used in concert with experimentation on real infrastructure, to explore and evaluate/compare extensively various algorithms (e.g scheduling, result certification) on various scenarios. For instance, when evaluating our strategies to provide Quality of Service to Desktop Grids (SpeQuloS is presented in detail in Section 4.2), we developed a simulator which was able to simulate SpeQulos, two Desktop Grid middleware (XtremWeb and BOINC), running on three kinds of infrastructures: Amazon EC2 using Spot instances, Grids in Best effort mode, and Desktop Grids. Our simulation campaign lasted several months, corresponding to more than 30.000 executions on Grid5000. More recently, in the scope of the ANR MapReduce project , we developed a simulator for hybrid MapReduce environment based on the SimGrid framework [187]. BigHybrid [178] simulates two MapReduce middleware: Hadoop over BlobSeer [188] and MapReduce/BitDew on two different computing environments Clouds and Desktop Grids and is being validated on Grid5000.

However, although simulation allows to validate easily and rapidly new ideas for algorithms, it is not always sufficient for validating full and complete technological solutions, in particular when it comes to software, services and infrastructures. Often papers are published in the literature on Desktop Grid computing without presenting experiments that reflects the problematics of an Internet-wide deployment. To evaluate distributed systems, the tool of choice for the French community is the Grid5000 scientific instrument [6], which is an infrastructure dedicated to support experiment-driven research in all areas of computer science related to parallel, large-scale or distributed comput-

ing and networking. Physically, it consists of more than 7.000 cores distributed among about 1.500 nodes located in ten different sites. Grid5000 experimenters benefit from a controlled environment and reproducible experimental conditions. To experiment on Grid5000, users configure the complete software stack using virtual operating system images and deploy these images on each node. Obviously, this is not enough when evaluating a Desktop Grid system, because the computing nodes and networks have very different characteristics. Thus we proposed a new methodological approach aiming at assessing the feasibility of running a system in a real world Desktop Grid infrastructure. The experimental protocol consists of ten experiments, all very simple to set-up and run on Grid5000, which individually test one aspect of a Desktop Grid deployment, and all together give a good perspective on how a system would behave when deployed in a real environment. We call these experiments the *Desktop Grid checklist* because to be validated, a system has to pass successfully all the tests that address network connectivity (firewall, NAT), node and network failures, sabotage, heterogeneous network and computing nodes, stragglers and more. We applied this method broadly to fill the gap between in-house development and real-world deployment. For example, it has been used in [173] to compare between regular Hadoop and our own implementation of MapReduce for Desktop Grid when deployed on WAN.

3.3 DSL-Lab: a Platform to Experiment on Domestic Broadband Internet

Experimental platforms such as PlanetLab [189] and Grid'5000 are promising methodological approaches to study distributed systems. However, both platforms focus on high-end service and network deployments only available on a restricted part of the Internet, leaving aside the possibility for researchers to experiment in conditions close to what is usually available with domestic connection to the Internet. High-speed Internet access has become common in home families; ADSL (Asymmetric Digital Subscriber Line) lines are wide-spread and fiber optic communication is now gaining significant market penetration. The progress realized by these technologies allows Internet provider to offer their customer an Internet connection comparable, in term of bandwidth, to local area network (up to 1Gb/sec). However, the architecture of a network of home PCs interconnected by ADSL presents special characteristics: *i*) the physical characteristics of the network differ substantially from the LAN characteristics, already well studied, because of the asymmetric communication performance (download/upload) and the internal ISP topologies; *ii*) within each family home, users share their Internet connection between several machines, using wired and/or WiFi local network as well as NAT and Firewalls to protect their network; *iii*) new classes of network appliance, beside the regular PC join this network: wifi phones, media center and IPTV, Network Attached Storage, networked gaming console etc. Furthermore, the network resource might be shared between several communication demanding applications (VOIP, P2P, gaming).

In 2007, I coordinated the DSL-Lab project, which was aiming at establishing a platform to experiment on distributed computing over broadband domestic Internet [190].

DSLLAB was a collaboration with Laurent Lefevre and Jean Patrick Gelas from the INRIA Reso team at Lyon and Oliver Richard and George Da Costa from the MESCAL team in Grenoble. The two main contributors for this platform were two PhD students: Lucas Nussbaum, advised by Olivier Richard, and Paul Malécot, co-advised by Franck Cappello and myself.

DSL-Lab is a complementary approach to PlanetLab and Grid'5000 to experiment with distributed computing in an environment closer to how Internet appears, when applications are run on end-user PCs. DSL-Lab is a set of 40 low-power and low-noise nodes, which are hosted by participants, using the participants' xDSL or cable access to the Internet. The objective is to provide a validation and experimentation platform for new protocols, services, simulators and emulators for these systems. DSLLab features:

- *Hardware and Network*: we had to select specialized hardware so that it would be powerful enough for conducting all our experiments, but low profile enough so that it won't disturb volunteers. We selected the Neo CI852A-4RN10 barebone (Celeron M 1GHz, 512MB RAM, 2 Gb Compact Flash storage), which belongs to the Mini-ITX class of PC, characterized by a small size form factor, an absolute silence, thanks to the absence of fan or moving part, and low power processor. In January 2009, 32 DSLnodes were distributed on the French major DSL providers (Orange, Free, Neuf, Tele2), giving a good perspective on the broadband heterogeneity, as 4 technologies are allowed in France: ADSL, ADSL2, ADSL2+ and ReADSL.

- *Remote OS Deployment* The DSL-Lab system is able to deploy remotely a new OS on every DSLnode without asking for volunteer intervention. One of the major concerns when designing the DSL- Lab platform was to avoid as much as possible volunteer intervention on the nodes. So, we needed to be able to re- install (in case an experimenter breaks the installed system by mistake) and upgrade (for security reasons, or to install additional software) the whole software stack, including the operating system installed on DSLnodes.

- *Connectivity and Security* The platform is managed in such a way that only identified experimenters have access to it. Experimenters first log into a central DSLLab server, which acts as a gateway and provides remote access to each DSLnode within a VPN through SSH.

- *Resources and Power Management* Most of the DSL-Lab experimenters are familiar with the Grid'5000 platform. To leverage their knowledge acquired on Grid'5000, we have adapted the Grid'5000 batch scheduler, called OAR [160], to the DSL-Lab platform so that: *i*) experimenters have a similar work environment and *ii*) it would eventually facilitate the connection of both platforms. Thanks to OAR, several experimenters may reserve some nodes in advance and deploy their experiments simultaneously. Because DSLnodes are hosted on a volunteer basis, a request of the volunteers is that the DSLnode does not waste power. Besides selecting thrifty hardware, the system ensures that the DSLnodes are powered-off when not used, thus reducing electricity consumption. The node stays up if the

next experimentation starts soon enough, or re-schedules its next wake-up time accordingly. Even if not reserved, the nodes wake up on a regular basis to check if new reservations have been created.

We evaluate the power consumption of DSLnodes, in order to price DSLnode hosting as our volunteers are not refunded for the electricity consumed. A node consumes around 1.5W when turned off (due to wake-on-LAN and software power switch); 9-10W when the CPU is idle and 13-14W at 100% CPU load. According to the French regulated electricity price, the cost per hour is 0.00182€ if the DSLnode is used and 0.000195€ otherwise. If we assume usage of the platform to be 8 hours a day, 5 days a week, 11 months a year, it costs 3.57€/year for a volunteer to host a DSLnode.

3.4 The European Desktop Grid Infrastructure

In the previous sections, we introduced the experimental methods and platforms on which we relied to investigate, design and build Desktop Grid systems. Obviously systems shouldn't restrict to "in-vitro" study, but eventually should follow their faith and reach the real world. In this Section, we report on a unique attempt to promote Desktop Grid infrastructures as an effective solution to enable the scientific community to solve their grand-challenge problems: *the European Desktop Grid Infrastructure* (EDGI).

E-infrastructures play a distinguished role in enabling large-scale innovative scientific research in Europe. In 2007, the European Union started to fund several FP7 projects with the aim of establishing a new kind of distributed computing infrastructure to provide a large amount of affordable computing resources that would supplement the already established EGI (European Grid Initiative). The aim of the project was several folds :

- Establish a global and unified infrastructure based on the interconnection of existing local institutional Desktop Grids and public Volunteer Computing systems. Two Desktop Grid technologies were supported: SZTAKI Grids (based on BOINC) [39] and XtremWeb-HEP [38]. The goal was to reach more than 100K connected machines on a dozen different sites.

- Bridge the EDGI infrastructure to the main Grids existing in Europe so that jobs submitted by regular Grid users can possibly be executed transparently on Desktop resources provided by EDGI. The technologies to connect the Service Grids and Desktop Grid aim at the following properties: secure, transparent to use, scalable, provides QoS, keep traceability and accountability, compliant with Grid standards, able to transmit large data. In addition, EDGI aims at being connected to several European Grid Infrastructures, which implies to be compliant with the major Grid middleware developed in Europe.

- Although scientific users of EGI are already well engaged in Grid technologies, EGI communities are restricted by the capacity of the VOs they can access. An important objective of the project is to raise awareness of the availability of increased capacities among new user communities and to attract them and actively engage

them in exploiting this enhanced e-infrastructure. New users should be trained to "gridify" their applications for the combined EGI/EDGI e-infrastructure.

In total, four projects were funded by the E.U.: EDGeS (Enabling Desktop Grid for e-Science), EDGI (European Desktop Grid Initiative), DEGISCO (Desktop Grid for International Scientific Collaboration), IDGF-SP (International Desktop Grid Federation Support Program) that were focusing respectively on enabling the infrastructure, developing advanced technologies and providing support to user. Thus, I've enthusiastically joined this large collaboration lead by Peter Kacsuk, which involves directly several partners: SZTAKI in Hungary, INRIA and CNRS in France, University of Coimbra in Portugal, University of Cardiff and Westminster University in the U.K., CIEMAT, Ibercivis and Fundecyt in Spain, AlmereGrid in the Netherland, University of Paderborn in Germany and University of Copenhagen in Denmark. I took leadership of two key Joint Research Activities (JRA) work packages: *"Bridge Service Grid to Desktop Grid (3G Bridge)"* (EDGeS) and *"Quality of Service for the EDGI infrastructure (SpeQuloS)"* (EDGI).

Under the scope of these projects, two people joined me as research engineers: Haiwu He, who worked on the 3G bridge, and Simon Delamare, who worked on the SpeQuloS middleware. In addition, we closely collaborated with Oleg Lodygensky and Etienne Urbah at LAL, for all the activities that implied XtremWeb-HEP. I report now on the main achievements of the project with a focus on the scientific results on which my group was actively involved.

The Bridging of Grids and Desktop Grids : The Generic Grid to Grid bridge (3G Bridge), developed in collaboration with SZTAKI institute and other members of the consortium, [191, 156, 150, 155, 192, 154, 193, 194, 195] allows to route jobs between Grid, Clouds and Desktop Grid infrastructures. Architecture of the 3G Bridge is presented in Figure 3.1, it's a generic middleware that interconnects infrastructures through plug-in adaptors for the various Grid (gLite, Unicore, ARC), Desktop Grid (BOINC, XtremWeb-HEP) and Cloud (OpenStack, OpenNebula) middleware. An example of integration is shown in the Figure: on the left-hand side of the figure is shown typical EGI VO, while on the the right-hand side, one can see three Desktop Grid systems: one public BOINC (SZDG: SZTAKI Desktop Grid), one private BOINC (UoW: Univ. of Westminster) and one public XtremWeb-HEP (CNRS). To route jobs between infrastructures, the 3G bridge appear as one regular computing element (DG-CE) of the Grid. Jobs, submitted through gLite, are first check to ensure that the application to be executed is part of the Application Repository, which contains a managed list of applications eligible to run on the EDGI infrastructure. If so, the 3G bridge submits the job to the corresponding Desktop Grid, ensuring key tasks of events logging, resource monitoring and conveying security information, in particular X509 proxy.

Security model The security concepts of the two bridged Grids are different. For instance, on one hand, BOINC does not require X509 certificates, but permits adding of new work units only to the BOINC project owner, and on the other hand, EGI

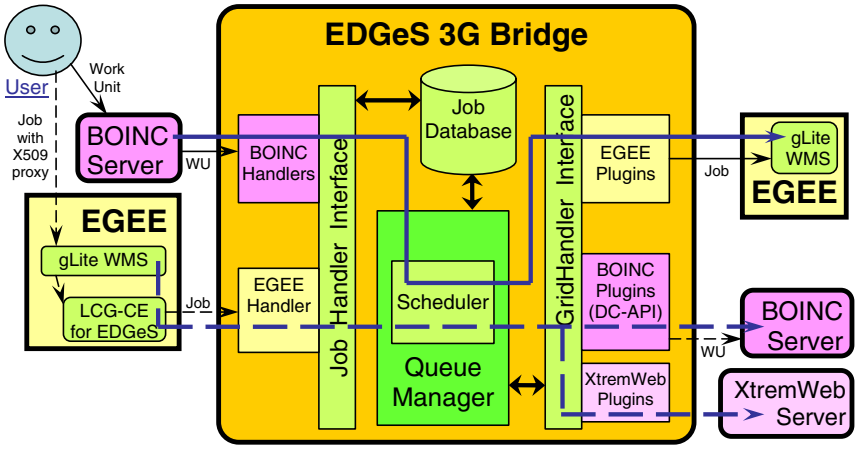


Figure 3.1: Architecture of the EDGeS 3G bridge

provides access to every user with a registered X509 certificate [156, 196]. In order to gain access to EGI resources and run the project's work units on EGI, the BOINC project owner has to send (just only once) the DN of his X509 proxy to the 3G bridge administrator. Then, following standard EGI rules, he stores his X509 proxy inside a MyProxy server which unconditionally trusts the EDGeS 3G bridge. As a consequence, the EDGeS 3G bridge must be operated with at least the same security level as a MyProxy server. The bridge components generate detailed logging, so that the bridge administrator can quickly identify the BOINC project and work unit for an incidentally maliciously working EGI job started by an EGI plug-in.

Quality of Service An important challenge with provide QoS support for those applications that require a faster execution in the public DG part of the infrastructure. For example, a public DG system enables clients to return work-unit results in the range of weeks. We developed the SpeQulos system (detailed in section 4.2) [197, 55, 191], which provides a probabilistic guarantee of job throughput during the execution. The principle is that SpeQuloS will dynamically deploy fast and trustable clients from some Clouds that are available to support the EDGI DG systems. SpeQuloS offers strategies that take the right decision about assigning the necessary number of trusted clients and Cloud clients for the QoS applications.

Handling Data-intensive Application Applications that have high data requirements are challenging to execute for EDGI [164, 78]: *i*) large files have to be distributed efficiently on a very large number of nodes, *ii*) data input files cannot be directly transferred from the Grid storage elements to the Desktop Grid nodes for security reasons. The solution developed in the scope of the project is AtticFS [86, 163, 79], which is a dedicated distributed storage system. This protocol is implemented by having a group of computers on the Internet that act as "Data Centres" that

replicate the files amongst themselves (in a P2P fashion) and distribute them when needed as requests from clients are processed. Depending on the security needs of an individual project, these Data Centres can either be known and secure hosts, or (potentially less secure, however, more flexible and freely available) volunteered resources donated by people in the community.

Virtualization Support for the Desktop Grids Porting application to the Desktop Grid infrastructures can turn into a complex and tedious task due to the: *i*) the necessity to instrument the source code using the BOINC API, *ii*) provide the application binaries for different architectures (e.g. Intel, ARM) and operating systems (e.g. Linux, Windows, and Mac OS X). Fortunately, thanks to virtualization technologies, Desktop can execute applications in a customized environment regardless of the host operating system. We gave BOINC and XtremWeb-HEP such capabilities, so that EDGI users can provide an execution context (OS, librairies, tools) as VM images for their application [146]. Thanks to this new capability, we have been able to execute extremely complex applications, such as the ROOT high energy physics applications [148].

Besides these technical developments, the EDGI consortium has advanced towards the Grid community on several fronts: *i*) standardization. Following several round tables organized by the consortium at the Open Grid Forum, discussions happened to decide of the relevant GRID standards that should be followed by EDGI, or, alternatively, if the consortium should propose new ones. The results can be found in the implementation of the 3G bridge, which supports several standards and common API such as, for instance the HPC profile job submission mechanism which ensures that the EDGI Bridge will be compatible with the middleware supported by EGI; *ii*) by providing Grid users a clear methodology to easily port their applications to the infrastructure. This methodology includes a questionnaire to match application requirements with infrastructure capabilities plus a sandbox testbed to develop and try the application [145], *iii*) an application repository and a Grid portal to facilitate usage of the infrastructure by Grid end-users.

At the end of the project, the infrastructure allowed to inter-connect 9 Desktop Grid systems, both local and public, providing more than 200.000 computing nodes; 5 Clouds systems providing around 300 cores; connected to 17 EGI VOs, 3 ARC VOs and 2 UNICORE VOs. Several high profile applications were able to take advantage of this very powerful infrastructure. One of the killer application was the Docking portal for biologists and chemists which comprises the Autodoc and Vina applications, which was used by more than 70 registered users submitting more than 60.000 jobs [198, 199, 200, 201, 202].

3.5 Conclusion

The problematic of evaluation is one of the most difficult challenge of Desktop Grid research. It concentrates many difficulties such as reproducing experimental conditions,

experimenting in a controlled environment, lack of knowledge of the underlying infrastructure.

In this Chapter, I have presented our methodology to develop, validate and evaluate the products of our research. This broad range of tools concerns: *i*) observation of existing Desktop Grid platform with the main goal of understanding the characteristic of the infrastructure and characterizing the computing resources in terms of availability and dependability, *ii*) simulation with the aim of designing algorithms and testing them over a wide range of parameters and scenarios, *iii*) emulation of Desktop Grid infrastructure over Grid5000 in order to develop and improve our software in conditions close to the one of a real Internet deployment. In addition, we developed a new experimental platform called DSLLab, specifically designed for experimenting on broadband Internet. DSLLab presents several original innovations that are not found elsewhere; in particular it has been developed with the objective of maximizing its energy efficiency, both by the selection of specific hardware and by the resource management strategies that we have developed.

Overall, EDGI has been one of the first real-world infrastructure that went so far in integrating Desktop Grid in every-day life of scientists and HPC users. Several conclusions can be drawn from the EDGI experience. First, being used by actual users community has helped us to understand deeply what is expected from such users. Even if the workload can flow from the Grid to the Desktop Grids transparently, thanks to the huge effort of development, the user experience is still significantly different because of the lack of performance model for Desktop Grid computing. This has lead me to considerably reconsider how performances of Desktop Grid should be measured and assessed. Second, the EGI/EDGI combined e-infrastructure has been one of the first hybrid DCI, combining several distributed computing infrastructures relying on different characteristics and paradigms. We'll see in the next Chapter that this emergence of hybrid infrastructure has also a strong impact on the design of algorithms in particular with respect to resource management.

Chapter 4

Algorithms and Software for Hybrid Desktop Grid Computing

Why Computer Science is No Good
In the past, computer scientists have found it convenient and productive to adopt a model of the computational universe that was very different from our models of the physical universe. This is changing. As we build bigger computers out of smaller components, our models of computation are forced to change. There is reason to hope that our new models for specific systems will be similar to the models of physics.
A computer designer is constrained by mundane problems that have no counterparts in the theoretical models of computer science: the size of connectors, the cost and availability of components, the mechanical layout of the system. Recently these factor shave dictated a dramatic change in the way we design computers. Things don't look the same. Wires cost more than gates, software costs more than memory, and the air conditioner takes up more room than the computer. Our current models of computation are inadequate for designing or even describing our new architectures. An abstract model is powerful only when it allows us to pay attention to certain aspects of a situation while ignoring others. Our current models seem to emphasize the wrong details.

(The Connection Machine, Daniel Hillis (1956-))

The previous Chapter presented our methodology toolbox to experiment and evaluate Desktop Grid Computing research. Based on this methodological stand, we now present the proposed algorithms, their implementation, and the main evaluation results. The goals of our research have been to improve Desktop Grid systems and to enlarge their application domain, by overcoming the barriers that prevented the execution of a new range of applications and by improving user experience when using Desktop Grid infrastructure. Explored research topics include: scheduling, security, quality of service

and virtualization. Over the years, the general context of Desktop Grid computing has evolved greatly with the advent of hybrid DCIs, which has led us to strongly reconsider many aspects of the techniques and algorithms for resources management. The challenge now is not only to make Desktop Grid efficient, but also to combine several types of infrastructures together. For instance, we revisited several of our work around resource management to consider stable and volatile resources. This allowed us to develop new QoS policies that rely on the stable resources provided by Cloud platforms to compensate for the volatility of Desktop Grid resources. This evolution led us to explore multi-criteria scheduling that takes into account both application requirements and user preferences and match them with infrastructure characteristics. This Chapter completes with a short description of the CloudPower technology transfer project, which aims at building a business model based on the Desktop Grid technologies co-developed by INRIA and CNRS.

4.1 Scheduling for Desktop Grids

Research in scheduling and resource management has always been an active topic in Desktop Grid computing for the following reasons: *i*) in order to overcome the loss of performance due to host volatility, extreme heterogeneity, lack of trust and reliability, *ii*) to support new classes of applications beyond the Bag-of-Tasks applications: workflows with tasks dependency, data-intense applications, parallel applications with communications, application with hard or soft-realtime constraints and more.

Our first investigation on resource management began as a continuation of host availability characterization [184, 183, 3]. In XtremWeb, the detection of faulty nodes is ensured by a heart-beat mechanism, and the faulty task is rescheduled to the next available host. We explored several strategies to improve task redundancy and replication and showed that such strategies were effective, in particular at the end of the computation, when the application execution termination only depends on the few last remaining tasks and there are still available computing resources. There are several reasons for task replication in such case. First, it avoids the cost penalty to reschedule the task in case of node failure. Second, it preemptively prevents the lagger effect, where a very slow computing resource would be attributed a task. Last, there is a greater chance to obtain a fastest computing resource, which is important in case of strongly heterogeneous platform.

Going further, with Derrick Kondo and Bruno Kindarji, student at Ecole Polytechnique, we looked at executing soft real-time applications on Enterprise Desktop Grids – soft real-time applications often have a deadline associated with each task but can afford to miss some of these deadlines. A number of soft real-time applications ranging from information processing of sensor networks, real-time video encoding, to interactive scientific visualization could potentially benefit from Desktop Grid platforms. While this challenge entails a myriad of issues (such as timely data transfers), the contribution we developed in [53] is to achieve probabilistic guarantees on task completion rates via buffering. That is, we determined how large a buffer must be allocated to ensure that

fraction of tasks meet their corresponding deadlines. Thus, we developed a model of the successful task completion rate as function of the server's buffer size, and showed that the aggregate compute power of Desktop Grid systems can be modelled using a normal probability distribution. Our model can be used by system developers who wish to determine an adequate buffer size for their (soft real-time) application to guarantee a certain task completion rate.

It is sometimes the support for new classes of application that has driven the research on scheduling algorithms. Executing BoT applications that have very large input data was the contribution of Baoha Wei during the beginning of his PhD thesis. Preliminary experiments on using a P2P content distribution protocol, such as Bittorrent [78, 163, 79], have shown that the protocol was efficient at distributing large files to great number of nodes. In [163], we worked with Fernando Costa and Luis Silva from University of Coimbra and Ian Kelley from the Cardiff University to validate this approach on the BOINC platform. However, the Bittorrent protocol suffers from a high overhead when transmitting small files. In [164, 165], we explored performance model that allows one to select the best file transfer protocol according to the file size and the number of receiving nodes. Furthurmore, we proposed an enhancement of the built-in incentive mechanism of BitTorrent in order to implement more predictive and deterministic communication ordering. By constraining the BitTorrent protocol to accomplish the file transfer in a pre-determined sequence, we were able to calculate a prediction of the communication cost. Therefore, we proposed BitTorrent dedicated variants of well-known scheduling heuristics such as MinMin, MaxMin and Sufferage [203], that obtained a speed-up of 3 when compared to classical round robin algorithm with the FTP protocol.

4.2 SpeQuloS, a QoS Service for Best-Effort DCIs

Although the aforementioned research work has focused on Desktop Grid, some of the results could be generalized to other *Best Effort DCIs*. Best-Effort DCI (BE-DCI) is an infrastructure or a particular usage of an existing infrastructure that provides unused computing resources without any guarantees that the computing resources remain available to the user during the complete application execution. For instance, the OAR [160] Grid scheduler manages a best effort queue to submit tasks on the idle nodes of the cluster with the lowest priority. At any moment, a regular task can steal the node and abort the on-going best effort task. In Cloud computing, Amazon proposes EC2 Spot instances [161] where users can bid for unused Amazon EC2 instances. If the market Spot price goes under the user's bid, a user gains access to available instances. Conversely when the Spot price exceeds his bid, the instance is terminated without notice.

Because BE-DCIs trade reliability against lower prices, they offer poor Quality of Service (QoS) with respect to traditional DCIs. This loss of QoS was particularly noticeable on the EDGI infrastructure, where Grid users' jobs could be executed either on a Grid or on a Desktop Grid. In the former case, the completion time for similar jobs would have a greater variability, with sometimes very long execution time.

The main source of QoS degradation in BE-DCIs is due to the *tail effect* in BoT

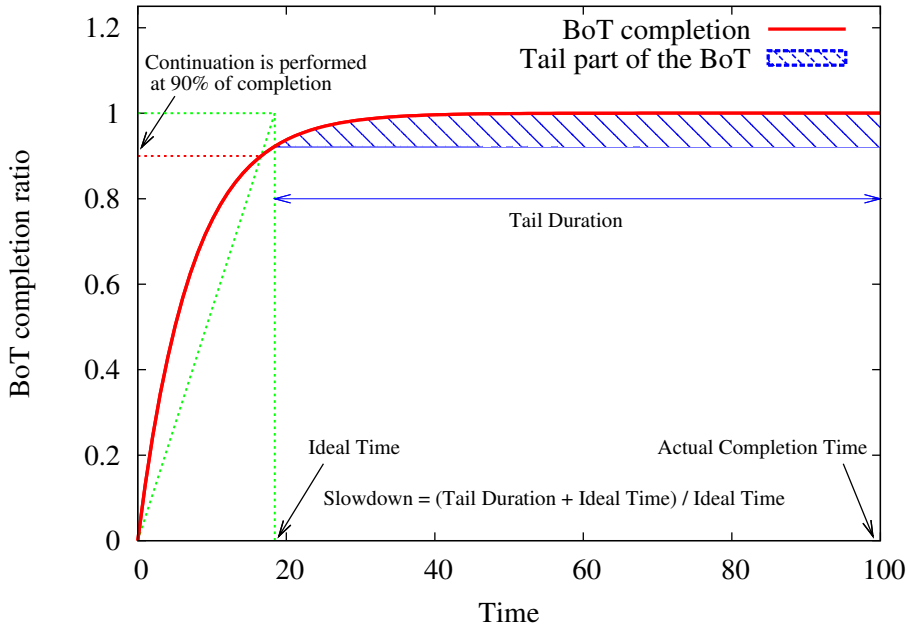


Figure 4.1: Example of BoT execution with noteworthy values

execution. That is, the last fraction of the BoT causes a drop in the task completion throughput as illustrated in Figure 4.1. To characterize this tail effect, we investigated the difference between the BoT actual completion time and an ideal completion time. The ideal completion time is the BoT completion time that would be achieved if the completion rate, calculated at 90% of the BoT completion, was constant. Intuitively, the ideal completion time could be obtained in an infrastructure, which would offer constant computing capabilities. We define the *tail slowdown* metric as the ratio between ideal completion time and actual BoT completion time. The tail slowdown reflects the increase factor of the BoT completion time resulting from the tail effect.

We observed and characterized the tail slow-down on several Best-effort DCIs. About one half of BoT executions are not extremely affected by the tail effect, meaning that the execution is slowed by less than 33%. Other cases are less favorable; the tail effect doubles the completion time from 25% of executions with XtremWeb-HEP middleware to 33% with BOINC. In the worst 5% of execution, the tail slowdown is up to 400% with XtremWeb-HEP and 1000% for BOINC. Moreover, a few percent of BoTs' tasks belong to the tail, whereas an significant part of the execution takes place during the tail. These results are mostly due to host volatility and the fact that Desktop Grid middleware has to wait for failure detection before reassigning tasks.

To enhance QoS of BoT execution in BE-DCIs, we proposed a complete framework called SpeQuloS [55, 197] that addresses the tail effect issue. SpeQuloS improves the QoS in three ways: *i*) by reducing time to complete BoT execution, *ii*) by improving BoT execution stability and *iii*) by informing user about a statistical prediction of BoT

completion.

SpeQuloS is a service which provides QoS to users of Best Effort DCIs managed by Desktop Grid middleware, by provisioning stable resources from Cloud services.

SpeQuloS implements various strategies to ensure efficient usage of Cloud resources and provides QoS features to BE-DCI users. As access to Cloud resources is costly, SpeQuloS provides a framework to regulate access to those resources among users and accounts for their utilization.

SpeQuloS collects information on BoT executions, which is relevant to implement QoS strategies. Careful exploitation of history of collected BoT execution traces as well as real-time information about the progress of BoT execution enable to compute a predicted completion time for the running BoT. The statistical uncertainty returned to the user is the success rate (with a $\pm$ 20% tolerance) of predictions performed on previous BoT executions, observed from the historical data.

We designed several different strategies to decide when and how many Cloud resources should be provisioned to handle the tail execution. Simplest strategies launch Cloud instances when the number of tasks which has completed or are assigned to workers reaches a threshold, corresponding to a fraction of the total BoT size. More elaborate ones monitor the execution variance, i.e the difference between the task completion rate and the task assignation rate to anticipate that the system in no longer in steady state. The second leverage is how many Cloud resources should be allocated. We propose two approaches: a greedy one maximizes the Cloud resources and a conservative one estimates the remaining time to complete the tail assuming a constant BoT completion rate to adjust the number of provisioned Cloud resources. Finally, SpeQuloS has several options in the way of using Cloud resources, depending if the Cloud workers are not differentiated from any regular workers by the DG server.

We evaluated combinations of these strategies not only using Desktop Grid trace-driven simulations, but also against Grids traces and simulated Amazon spot instances. All the strategies are able to significantly address the tail effect: the tail has disappeared in one half of the BoT executions and for 80% of the BoT executions the tail has been at least halved, which is satisfactory. The impact on the BoT execution is significant, leading in the best cases to a speed-up of 2. Still, these strategies consume few Cloud resources, actually, less than 2.5% of the BoT workload is executed in the Cloud. In terms of QoS, SpeQuloS increases the execution stability, meaning that BoTs executed in similar environments will present similar performance. Furthermore, SpeQuloS can accurately predict the BoT completion time and provide this information to BE-DCI users. The predicted completion time given by SpeQuloS is correct within $\pm$ 20% in 9 cases out of 10, which is remarkable given the unpredictable nature of BE-DCIs.

I proposed SpeQuloS as the QoS JRA leader in EDGI. Simon Delamare, recruited as a postdoc on this project, is a major contributor. It is also a joint work with Derrick Kondo and Oleg Lodygensky. With Oleg Lodygensky, we validated the framework on a complex infrastructure of the Desktop Grid IN2P3, grid nodes and EGI G5K. Other members of the consortium, in particular Peter Kacsuk and his team at SZTAKI, Tamas Kiss at Westminster University have helped us to achieve quality production, which is necessary for deployment on the EDGI infrastructure.

4.3 The Promethee Multi-criteria Scheduler for Hybrid DCIs

Since users now have a choice of infrastructure to run their applications, the question arises of how to allocate the use of resources between infrastructure based on user preferences: speed, low energy consumption, budget etc.

One of the lessons learnt from EDGI, is that Desktop Grid middleware is an excellent candidate for managing hybrid DCI. The *pull-based scheduler* offers several desirable properties, such as scalability, fault resilience, ease of deployment and ability to cope with elastic infrastructures. On the other hand, a pull-based scheduler has a limitation: when a computing resource asks for work, the scheduler has no or very limited choice to select the best resource to execute a task. This issue is particularly relevant in the context of hybrid DCIs, where the infrastructures have very heterogeneous capabilities in terms of computational power, reliability, security mechanisms, type and frequency of failures, cost, and power efficiency. Using efficiently these infrastructures requires taking into account different parameters when making resource management decisions. Thus, new schedulers are required, which can implement strategies allowing *multi-criteria decisions*, such as cost/energy/performance. For this reason, providing multi-criteria pull-based scheduling that is both scalable and smart remains a challenge.

In collaboration with Mircea Moca, during his stay as invited professor in our team, Cristian Litan and Gheorghe Cosmin Silaghi from the University of Babes-Bolyai (Romania), we addressed this issue in [204] and [205] by proposing a new pull-based scheduler which relies on the *Promethee* decision-making method for selecting tasks by taking into account user preferences and resource characteristics. Promethee [206] is a non-parametric decision model which employs pairwise comparisons to produce a *ranking* of potential alternatives [207].

Promethee considers the set of criteria $C = \{c_{i_c}, i_c \in N\}$ characterizing a set of alternatives A. Like other multi-criteria decision methods, Promethee starts from an evaluation table, then, by making pairwise comparisons of the evaluations within each criterion, computes a dominance relation.

Promethee requires a preference function: i) between criteria defined by giving criterion importance weights $\omega_{i_c} \equiv \omega(c_{i_c})$, and ii) within each criterion, by using a preference function P that takes as input the amplitude of the deviation between two criterion evaluations. The literature [207] proposes several models for the preference function P, such as Linear, Level, V-shape or Gaussian; otherwise one can define his own function. The design of the preference function influences the output of Promethee. It consists in adjusting the threshold for the amplitude of the deviation between the criteria to determine the alternative selection.

When applied to pull-based scheduling, the Promethe method computes for each task a set of criteria according to the characteristics of the node requesting the task. The criteria that we propose are: Expected Completion Time represents an estimation of the time interval needed by a pulling host to complete a particular task; Price represents the estimated cost eventually charged for the completion of a task; and Expected Error Impact indicates the estimated impact of scheduling a particular task to the pulling host taking into account its reputation (error proneness) and the size of the task.

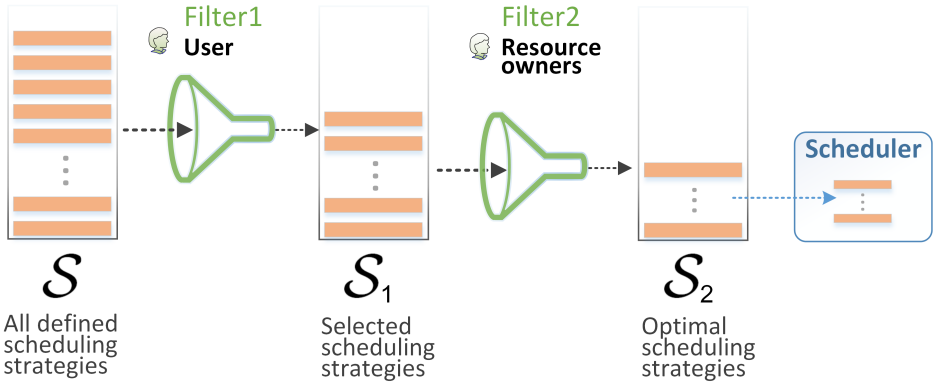


Figure 4.2: Satisfaction Oriented FilTering (SOFT) method

We simulated the Promethee scheduler managing a hybrid DCI composed of Internet Desktop Grid (IDG), Cloud and Grid infrastructures, where node failures, laggers and erroneous results were taken into account. We observed that the Promethee scheduler outperforms First Come, First Serve, the baseline Desktop Grid scheduler, at different degrees according to the type of DCI. While in Cloud environment the gain is 9-12%, it can reach up to 32% for IDG. For hybrid DCI, the Promethee scheduler shows a 38% improvement for the combined IDG/Grid infrastructure.

The Promethee scheduler allows application developers to empirically configure the scheduler to put more emphasis on criteria that are important from their own perspective. However, such configurable multi-criteria schedulers have two strong limitations: *i*) there is no guaranty that the user preferences expressed when configuring the scheduler actually translate in an execution that follows the favored criteria, and *ii*) the number of possible scheduling strategies explodes with the number of criteria and the number of application profiles, rapidly leading to an intractable situation by the user.

In [157], we proposed *Satisfaction Oriented FilTering* (SOFT), a new methodology that explores all the scheduling strategies provided by a Promethee multi-criteria scheduler to filter and select the most favorable ones according to the user execution profiles and the optimization of the infrastructure usage. The methodology, illustrated in Figure 4.2 begins with the definition of criteria to be integrated into the scheduler and a corresponding metric for each. On the constructed set of possible scheduling strategies a selection method is applied in order to find those that provide high and stable user satisfaction levels (Filter1). On this resulting subset a second selection method is applied in order to retain the strategies that are also the most efficient from the resource owners perspective (Filter2).

4.4 Virtualization and Security

Due to the anonymity and the lack of trust of participants, Desktop Grid Computing raises many security issues. In this Section, we review some of our most significant results concerning participants' machine protection, result certification and secure interoperability with Grid environment.

As mentioned in Section 2.2.5, sandboxing constitutes a cornerstone approach in the Desktop Grid security model. In collaboration with Attila Csaba Marosi and Peter Kacsuk, from SZTAKI and Oleg Lodygensky, we presented in [146] an method to provide a secure and transparent sandbox, common for XtremWeb-HEP and BOINC, for running untrusted DG applications. The sandbox environment is based on Virtual Machines (VM) techniques and can be easily integrated with DG worker with few extra overhead. It has the following features:

- simplify legacy application deployment by having a separate operating system available inside the VM, which contains all libraries and data needed to execute the application, regardless of the host OS configuration.
- System-level checkpointing. VMs have the ability to save their current state of execution to stable memory. The checkpoint image can be used to resume the execution on the same host, or can be migrated to a remote host.
- Enforce resource limits. The VMs can be configured in such a way that the DG application is limited in accessing the host resources; possibly denying some resources that could cause security issues, such as local disk or local network.

The sandbox environment consists in daemon software and APIs which allow to create and start VM instances, upload input files and executables into the instance, start tasks using the uploaded files, request status information and when finished retrieve the output files. We evaluated the environment against several virtualization approaches: Bochs and QEMU [208], which are open source processor emulators; and VMware Player and VirtualBox, which are two virtualization software.

A key security component of Desktop Grid systems is the result certification, which is needed to check that malicious volunteers do not tamper with the results of a computation. In collaboration with Mircea Moca, during his summer internship as a PhD student from University of Babes Bolyai and Gheorghe Cosmin Silaghi, we have proposed a new approach for result checking adapted to MapReduce computations. In classical Desktop Grid, the result certification is often centralized on the server. Unfortunately, this centralized approach in inefficient with MapReduce computation (See Section 5.2), because intermediate results might be too large to be sent back to the server. As a result, we have proposed in [174] a decentralized protocol for result certification based on majority voting. Although majority voting involves larger redundant computations, the heuristic is efficient at detecting erroneous results, which are likely to be significantly higher for disk bound jobs.

In [196], we propose the security issues when bridging Grid with Desktop Grid. The contribution is a new Desktop Grid security model to bridge this anonymous environment

to the strongly securized Grid one. XtremWeb-HEP introduces mechanisms aimed to secure and confine distributed resources usage; this new features permit to extend user actions over the platform as well as to secure resource usage and confine application deployment. The security model relies on *anonymous* and *trusted* resources, which can be users, data, application, and computing resources. The model defines access rights between each of these resources. When the resources are authenticated by the Grid, XtremWeb-HEP and the Grid $\leftrightarrow$ Desktop Grid bridge convey all the security credentials (usually X.509 proxies) to ensure authentication and activity logging.

4.5 CloudPower: a Cloud Service Providing HTC on-Demand

CloudPower starts from the observation that HTC is a key factor in knowledge and innovation in many fields of industry and service with high economic and social issues: aerospace, finance and business intelligence, energy and environment, chemicals and materials, medicine and biology, digital art and games, Web and social networks. However, acquiring HTC dedicated machines is very expensive, making HTC prohibitive to SMIs / SMEs for their research and development. The goal of CloudPower is to offer a low cost Cloud HTC service for small and medium-sized innovative companies. CloudPower leverages on the open-source software XtremWeb-HEP previously developed by the CNRS and INRIA. With CloudPower, companies and scientists will run their simulations to design and develop new products on a powerful, scalable, affordable, reliable and secure infrastructure.

CloudPower is a project of technology transfer supported by the ANR Emergence program, which I coordinate since 2013. Avalon/ENS-Lyon is the team leading the project, with the IN2P3/CNRS as the second technical and scientific partner, and the "Valorisation" office of ENS-Lyon as a partner for the business and legal development. Haiwu He and Sylvain Bernard have been recruited as research engineer and business developer respectively. Building on the network of SMIs from the competitiveness clusters System@tic and LyonBiopole, we implement scenarios and/or demonstrators which illustrate the ability of CloudPower to increase competitiveness, research and marketing of innovative SMEs. If the business model is conclusive, we envision the creation of a new and innovative company operating the platform.

4.6 Conclusion

In this Chapter, I presented algorithm and software contributions to Desktop Grid computing.

I introduced several scheduling algorithms aiming at improving Desktop Grid resource management when executing Bag-of-Tasks applications. We also addressed new classes of applications by considering the execution of BoT with soft-realtime constraints and BoT with large input data-sets distributed with a P2P protocol.

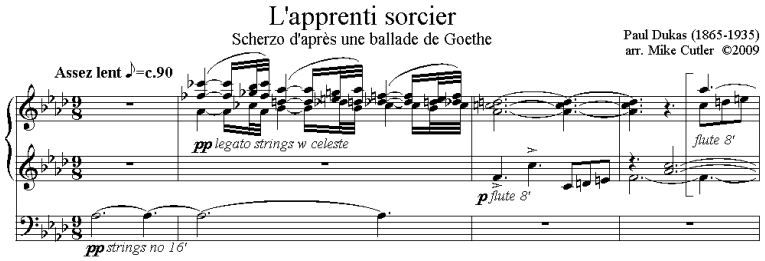
Security in Desktop Grids systems is a necessity for a broad acceptation and integration into actual cyber-science infrastructures. This issue was tackle from multiple

perspectives. We proposed a sandboxing environment, which is generic enough to be integrated to other DG systems, and which provides protection of the participant machine, while facilitating legacy application deployment. A new hybrid security model for Desktop Grid has been introduced, which cooperates with DCI by enforcing strong security requirements. Finally, we proposed a new approach for distributed result checking able to cope with data-intensive MapReduce computations.

Then I focused on two contributions that outline Hybrid DCIs problematics. The first contribution illustrates the benefit of having two classes of infrastructures so that one can minimize the drawbacks of the other. SpeQuloS, is a service which provides Quality of Service to execution that relies on volatile resources by provisioning stable resources from Cloud infrastructure. The second contribution explores the design of multi-criteria and user-oriented scheduling heuristics. The Promethee scheduler is able to use efficiently several DCIs, while allowing users to configure scheduling strategies according to their preferred execution profiles. However, such approach can lead to a combinatorial explosion of the possible scheduling strategies. This led us to propose Satisfaction Oriented FilTering, a method to filter and select the strategies that respond to application requirements and infrastructure usage. Our experience has shown that Desktop Grids were able to integrate the e-science cyber infrastructure, providing scientists large amounts of computing power at low-cost. The next step is to investigate how we can build business models based on Desktop Grid technology. This is the goal of the CloudPower project, which is proposes, a low-cost, on-demand and secure HTC solution for small and innovative business.

Chapter 5

Large Scale Data-Centric Processing and Management



(L'apprenti sorcier ("The Sorcerer's Apprentice") is a concert scherzo by Paul Dukas based on the poem Die Zauberlehrling by Johann Wolfgang von Goethe (1797))

Before jumping in the technical part of the Chapter, I would like the reader to remember Goethe's poem "The Sorcerer's Apprentice", popularized by the musical cartoon, produced by Walt Disney in 1940 and wonderfully accompanied by Dukas's symphonic piece. In this cartoon, Mickey is the Sorcerer's Apprentice and must fill a tank located in the wizard's underground laboratory. The apprentice has two pails of water in hands and he must walk from the water source to the tank, which appears to be a daunting task. By invoking maliciously a magic spell, Mickey animates a broom to get help carrying the water. Unfortunately the broom gets out of control and the situation quickly gets carried away. Mickey tries to stop the broom using an axe, but each of the pieces becomes a new broom, grabs a bucket and continues fetching water, now at twice the speed. At the height of the disaster, water waves engulf the laboratory and myriad of uncontrollable brushes are pursued by Mickey with an axe in hand.

This parable illustrates the challenge of the Data deluge that science and IT world is facing and what may happen if we consider the *magic* Cloud as the only recipient, and if no effort are pursued to control the data flow beyond the data center.

Increasingly, the next industrial innovative breakthroughs and the next scientific discoveries will depend on the capacity to extract knowledge and sense from the enourmous amount of *Big Data* information [209]. Examples vary from processing data provided by scientific instruments such as the CERN's LHC, the LSST Telescope in Chile, or the OOI large-scale underwater sensors network; grabbing, indexing and nearly instantaneously mining and searching the Web; building and traversing the billion-edge social network graphs; anticipating market and customer trends through multiple channels of information. Collecting information from various sources, recognizing patterns and returning human scale results from this "data deluge" is the new challenge the community is facing[210]. In this Chapter, we cover the research directions we pursued to enable large scale data processing and management in the context of Desktop Grid and Hybrid Distributed Infrastructures. Efficient data management, i.e ensuring data availability, multi-protocols file transfer, collective data distribution operation, scalable data indexing is considered a difficult issue by the Grid and Cloud communities. It is even more difficult, when we consider Hybrid DCIs, in particular because the characteristics of the nodes have a great impact on the design of the storage.

We believe that scientific applications require a fundamentally different paradigm for handling large scientific datasets when they are distributed on complex e-science cyber infrastructure. We present BitDew and BitDew-MapReduce, two environments for large scale data management and processing on Desktop Grid and Hybrid DCIs. The Chapter also introduces Active Data, our approach for data life cycle management, which has two distinguishing characteristics: data-centric (as opposed to task-centric) and event-driven (as opposed to task-completion triggered).

5.1 Environment for Large Scale Data Management

We identified Data management and distribution as being a major bottleneck in Desktop Grid computing. We conducted several studies with Bittorent, BOINC and XtremWeb [79, 163, 164, 165, 78] that showed that a P2P approach is a key advantage when executing Data-intense applications on Desktop Grids. In 2007, Haiwu He obtained an INRIA postdoc fellowship with the initial objective of building an environment for file distribution based on P2P protocols. Together, we initiated BitDew, which is a subsystem which can be easily integrated into Desktop Grid, Grid and Cloud 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. Later, when the software became more complex, I recruited José Saray as an engineer to develop new features and improve the quality of the code, thanks to the support of an ADT INRIA grant.

The key feature of BitDew is to leverage on special metadata, called here *Data Attributes*. In addition to traditional metadata that are used to index, categorize, and search data, as in other Data Grids System, data attributes control dynamically the repartition and distribution of data onto the storage nodes. Thus, complexity of Desktop Grids systems is hidden to the programmers who is freed from managing data location,

host failure and explicit host to host data movement. We have proposed different types of metadata, which correspond to data distribution abstractions : *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 data item 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.

While seeming simple at a first glance, the combination of these abstractions offers a powerful mechanism to implement complex data distribution pattern. For instance, we demonstrated in [211, 212] the ability to implement collective communication, à la all-to-all, along with replication and fault-tolerance on Internet volatile nodes. Such a scenario is still impossible using classical Desktop Grid middleware. In the next Section, we'll see how these abstractions enable us to implement a full MapReduce runtime environment.

An other remarkable design decision is the architecture of the BitDew services. Because BitDew is also aiming at being deployed on Grid infrastructures, where the storage resources and services span over several administrative domains, we wanted to break the monolithic architecture into a set of core services : data catalog, data scheduler, data repository and data transfer. Thus, the BitDew runtime environment is itself distributed, so that several service nodes can be instantiated in order to enhance reliability and scalability or to adjust with an existing infrastructure where data are distributed over multiple data servers. Thanks to this flexibility, we demonstrated for the first time that a Desktop Grid was able to execute efficiently a data-intense application, namely BLAST, by using a combination of data staging approach and P2P protocol [213].

The last point which makes BitDew suitable not only for Desktop Grids, but also for hybrid DCIs is the support for a wide variety of file transfer and storage protocols [214]; Traditional client/server protocols FTP, HTTP, SCP; P2P protocols such as BitTorrent; Grid protocol through SAGA interface [131], and more recently Cloud storage such as Amazon S3 and Dropbox. These unique features form a complete framework and toolbox that allows researcher to rapidly prototype their data-oriented application and experiment their ideas. To give few examples of such usages: When Adriana Iamnitchi and her PhD student Nikolas Kourtelis from the University of Florida studied distributed P2P social network dedicated to data sharing, they rapidly design a mock-up of the architecture to evaluate the relevance of DHT protocols in this context [215]; For the European FP7 EDGeS project, Ian Taylor and Ian Kelley, from University of Cardiff sketched-up the first design principles of P2P-Attics [85]; Mohamed Labidi, built an environment for data-driven master/worker computing [216]; To validate her simulation of a distributed hierarchical checkpointing system, Fatiha Bouabache and Franck Cappello implemented the protocol and conducted experiments on Grid'5000 [65, 217, 218, 219]. Obviously, BitDew has been the substratum to our own research around data management for hybrid infrastructure, some of them are described in the next sections.

Another BitDew follow-up work, which is still on-going, is the WukaStore hybrid stor-

age system [220], which is a joint work with Bing Tang. The objective of WukaStore is to propose configurable, reliable and scalable storage with file availability guaranty that can take advantage of the huge storage capacity provided by Desktop PC and stable Cloud storage. In this set-up, we imagine that participant would offer unused part of their online storage that online applications and services provide: for instance Flickr provides each users 1TB online capacity. We conducted a first opportunity study that was aiming at evaluating different storage strategies in term of data availability and durability, as well as storage overhead [180] using trace-based simulations. To use WukaStore, the user determines the requirements for their data storage in term of overall storage space, storage cost, probabilistic availability and durability. For instance, some file, such as the input or the output of a simulation may require strong availability and durability, even at the cost of a high storage overhead while very large intermediate results can accomodate a lower availability, because they can be computed again, as long as the storage overhead is minimized. Once the data requirements are understood, the user selects an adequate combination of several storage strategy leverage: chunk replication, recovery on failure, reconstruction based on erasure codes and repartition between stable and volatile nodes. Thus Wukastore is an example where taking advantage of infrastructure heterogeneity allows to overcome pure Desktop Grid limitations: stable but sparse or expensive storage (such as Cloud storage or reliable file servers) and large, inexpensive but volatile storage (such as idle storage harnessed from desktop PCs over the Internet). We'll see in the next section that this principle can be applied to data-intensive computing as well.

5.2 Implementing the MapReduce Programming Model on Desktop Grids

Since its introduction in 2004 by Google, MapReduce has become the programming model of choice for processing large data sets. MapReduce borrows from functional programming, where a programmer can define both a Map task that maps a data set into another data set, and a Reduce task that combines intermediate outputs into a final result. Although MapReduce was originally developed for use by web enterprises in large data-centers, this technique has gained a lot of attention from the scientific community for its applicability in large parallel data analysis (including geographic, high energy physics, genomics, etc.).

We started this research direction because we thought that applications requiring an important volume of data input storage with frequent data reuse and limited volume of data output could take advantage not only of the vast processing power but also of the huge storage potential offered by Desktop Grid systems. There exists a broad range of scientific applications, such as bioinformatics and simulation in general as well as non scientific applications such as web ranking or data mining which meet these criteria [221]. After co-organizing the MapReduce workshop with Geoffrey Fox from the Indiana University, I became convinced that supporting the MapReduce programming model on Desktop Grid was a necessary step to support the execution of data-intensive applications.

However, enabling MapReduce on Desktop Grids raises many research issues with respect to the state of the art in existing Desktop Grid middleware. In contrast with traditional Desktop Grids which have been built around Bag-of-Tasks applications with few I/O, MapReduce computations are characterized by the handling of large volume of input and intermediate data. The challenges includes: *i*) support for collective file operations which exist in MapReduce, in particular the Shuffle phase, which is the redistribution of intermediate results between the execution of Map and Reduce tasks; *ii*) the result certification, which is a key security component, has to be decentralized because the volume of intermediate data would be too large to be sent back to the server for certification; *iii*) dependencies between the Reduce tasks and the Map tasks, combined with hosts volatility and laggers can slowdown dramatically the execution of MapReduce applications. Thus, there is a need to develop aggressive performance optimization solution, which combines latency hiding, data and tasks replication and barriers-free reduction.

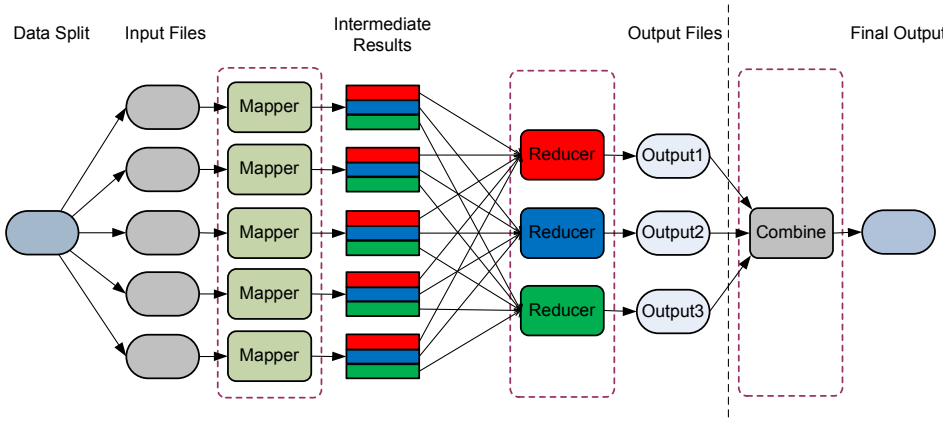


Figure 5.1: MapReduce flow of execution

In [175, 172], we have proposed an implementation of MapReduce runtime environment for Desktop Grid systems based on BitDew. Mircea Moca, during a 6 month PhD internship in our team and Stéphane Chevalier, during his ENS Master's internship designed the first prototype of the environment. To our knowledge, this was the first implementation specifically designed to execute MapReduce applications on an Internet environment, which fully addresses the problematics of this platform: nodes connectivity, malicious computing resources, and extreme nodes heterogeneity and volatility. We list some of the high level features that have been specifically developed:

- *Latency hiding and collective file operations* Implementing efficient communication in Desktop Grid present a considerable challenge. The latency can be orders of magnitude higher than latency provided by interconnection networks found in clusters. We overlap communication with computation thanks to the multi-threaded worker design, which can concurrently transfer several files, and execute several

Map and Reduce tasks with a high degree of control. A MapReduce execution comprises several collective file operations which in some aspects, look similar with collective communications in parallel programming such as MPI (see Figure 5.1).

- *Fault tolerance and scheduling* In Desktop Grid, computing resources have high failure rates, therefore the execution runtime must be resilient to a massive number of crash failures. Our implementation tolerates failures that can happen during the computation, either execution of Map or Reduce tasks, or during the communication, that is file upload and download. The protocol takes advantage of data and file transfer resilience provided by BitDew to redistribute file chunks when crashes are detected: the input chunks in case of mapper failure and intermediate results in case of reducer failure. Traditional cluster implementations of MapReduce such as Hadoop introduce several barriers in the computation and in particular between the execution of Map and Reduce tasks. In Desktop Grid systems, because there can be a long period of time before nodes reconnect, it is necessary to remove any barriers so that the Reduce task can start as soon as intermediate files are produced. To do so, we have slightly changed the API of the Reduce task to allow the programmer to write the reduce function on segment of keys interval. The early reduction combined with the replication of intermediate results allowed us to remove the barrier between Map and Reduce tasks. Traditional scheduling associates computation to processing nodes. In contrast, our implementation relies on a two-level scheduler. First, the placement of data on hosts is ensured by the BitDew scheduler, which is mainly guided by the attribute properties given to data. However this would not be enough to efficiently steer the complex execution of MapReduce application. The second scheduler is the MapReduce master, which detect laggers, that is, nodes which spend an unusually long time to process the data and slow down the whole computation. The master determines if there are more nodes available than tasks to execute. In this case, increasing the replication factor of the remaining tasks to compute can avoid the lagger effect.

- *Distributed result checking* As mentioned in Section 2.2.5, we have adapted the majority voting heuristics, as it exists in BOINC to the decentralized network of storage Reducers [174]. Once a reducer has obtained n out of p intermediate results, the result that appears most often is assumed to be correct.

Although this architecture has been specifically planned with Internet Desktop Grid as a target, it shows several similarities with Hadoop design, i.e master/worker, fault-tolerance, speculative execution, etc. Xuanhua Shi, from HUST University in Wuhan, China has been a XtremWeb user and after organizing a France-China workshop, with the help of the French consulate of Wuhan about Big Data and Cloud Computing, we decided to work together, and to organize a 6 months internship for his PhD student Lu Lu. Lu Lu extensively compared Hadoop and BitDew-MR to assess the contribution of each of these features to the feasibility of running MapReduce as an Internet Desktop Grid [173]. The evaluation was conducted on Grid'5000 using a set of independent

experiments, where each one were evaluating one aspect of the Desktop Grid: emulation of firewall, host churns, sabotage, massive node crashes and more. This new methodology approach allowed to assess with a great confidence, that our prototype was able to run in conditions close to an actual Internet deployment.

There are several scenarios that motivate the use of MapReduce on multiple infrastructures (for instance to increase storage and computing capacity or to reduce data transfer during the computation), and more specifically on Desktop Grid and Cloud systems. Enabling MapReduce on hybrid infrastructures is the purpose of the ANR MapReduce project [171, 222], funded by the French National Research Agency, on which collaborate INRIA, IBM France, the University of Rennes, Argonne National Lab and the Institut de Biologie et Chimie des Protéines. In this project, we investigate a runtime system that would allow to split a MapReduce computation over two different runtime environments; Namely BitDew-MR for the execution on Desktop Grid resources, while the execution on Cloud infrastructure is being handled by a combination of Hadoop with BlobSeer, a file system optimized for high concurrent write [223]. Julio Anjos, is a PhD student from UGFS, Brazil who is spendind one year in our team working on these issues. Julio's first result is a simulator of the whole environment (See Section 3.2), and he is now looking at strategies to split and distribute data between the two kind of infrastructures according to their computational capabilities.

A particular use-case for hybrid MapReduce concerns data privacy. During her Master's internship, Asma Ben Cheick proposed an approach [179] to protect data privacy by spreading data-sets on a combination of public and private clouds so that the compromise of an infrastructure would not allow the attacker to reconstruct the whole data-set. To do so, we rely on Information Dispersion Algorithms (IDA), which allow to split a file into pieces so that, by carefully dispersing the pieces, guarantees there is no method for a single node to reconstruct the data if it cannot collaborate with other nodes. An interesting and somewhat unexpected development of this work would be to take benefit of Desktop Grids, where collaborative attacks are much harder to perform to increase the overall security of the computation [99].

5.3 Handling Data Life Cycles on Heteregeneous Distributed Infrastructures

E-Science infrastructures form complex assemblages of data management services and computational software, which often span over multiple heterogeneous infrastructures.

As the volume of data grows exponentially, the management of this data becomes more complex in proportion. A key challenge is to handle the complexity of *data life cycle (DLC)*, i.e. the course of operational stages through which data pass from the time when they enter a system to the time when they leave it. The DLC starts when data enter the system either acquired by an instrument, or as results of a computation; the DLC terminates when data are physically erased, or when moved to storage outside of the system. Between these two points in time, data progress through a series of different stages (e.g., acquisition, cleanup, duplication, archival, transfer) that are either appli-

cation initiated (e.g., transformation, aggregation, metadata extraction) or triggered by external events (e.g., failures that lead to data unavailability).

The result is that life-cycles of scientific data set are becoming very complex and controlling the whole life-cycle almost intractable without understanding and monitoring the interaction between data-sets and e-infrastructures.

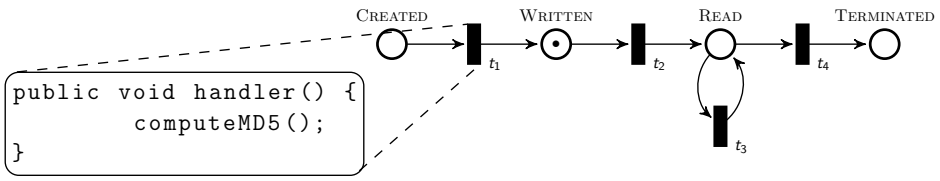


Figure 5.2: Representation of the *"Write-Once, Read-Many"* data life cycle: *Places*, represented by circles are the states of the life cycle; *Transitions*, represented by rectangles are the operations that happen on data items; *Tokens*, represented by • on Places, are data items in a particular state of the life cycle.

Active Data [224, 225] is the contribution of Anthony Simonet's PhD thesis, whom I advise since 2012. Active Data proposes a different and innovative paradigm for data life cycle management. We started this joint work with Matei Ripeanu (UCB, Canada), when he was invited professor at ENS-Lyon.

Active Data allows to reason about data sets handled by heterogeneous software and infrastructures. It's composed of:

- a *formal model* that captures the essential life cycle stages and properties: creation, deletion, faults, replication, error checking. The model is based on Petri Networks [226], which is a formalism and a graphical tool widely used for the analysis of systems with concurrency and resource sharing. An example of a DLC is presented in Figure 5.2.

- a *programming model* which allows code execution at each stage of the data life cycle. In Active Data, the programmer, specifies the set of data-related events (e.g., data item creation, replication, transfer completion, data loss, deletion) to be monitored per data item and programs the operations to be executed when these events happen. In the example presented in Figure 5.2, the transition t_2 is associated with a code that automatically compute the MD5 signature for each file copied in the system.

This programming model allows developing a broad range of data life cycle management (DLM) applications such as automated tiered storage, processing attached to any stage of the life cycle, coordination between data acquisition mechanisms and remote storage, content delivery networks, deep storage archival, incremental data management, and so forth.

The Active Data framework developed by Anthony Simonet has several high level features: *i*) it allows legacy data management systems to expose their intrinsic DLC and

to report about DLC events; *ii*) *compose* DLC to represent data movements from one system to the other, reconciling identifiers and offering a high-level and flat view of large data life cycles, abstracting hardware and software complexity; *iii*) powerful filters based on automatic Data Tagging, and guarded transitions, which only executes on data item having specific tags; *iv*) a scalable runtime system based on publish/subscribe paradigm.

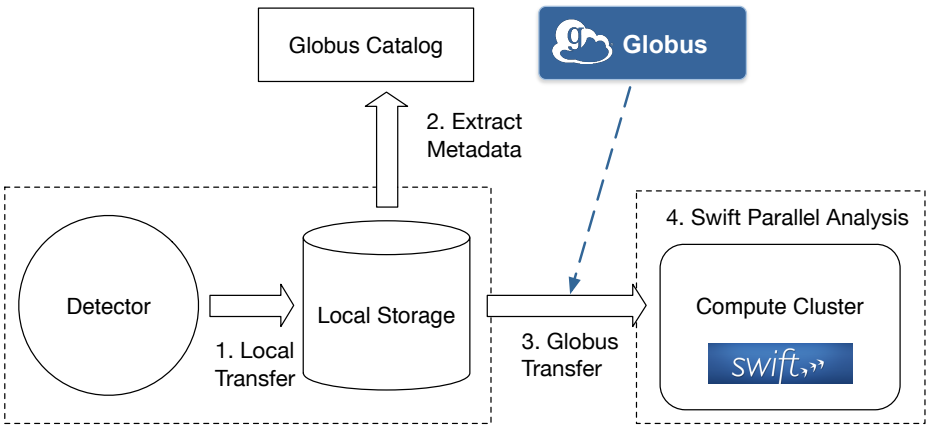


Figure 5.3: The Advanced Photon Source

As a use-case for this work we use a real-world application from the Advanced Photon Source (APS) at Argonne National Laboratory. This is a joint work with Ian Foster and Kyle Chard [227] from ANL and the University of Chicago, which took place in the context of the INRA/ANL Joint Laboratory on exascale computing. In this example, the researchers use a near-real time workflow in which data is automatically analyzed as it is acquired. The workflow is both compute and data intensive, requiring 1000 nodes for near-real time analysis and 3-5TB of data is generated per week. The workfow can be split in five stages : 1) data acquisition from a detector, 2) initial transfer to a larger shared cluster using Globus Online transfers, 3) data reduction and aggregation ; e.g., refinement operations and calculation of stresses and strains for individual grains, grouping of files into "datasets" per experiments or sample, 4) meta-data extraction; data is then cataloged in the Globus Catalog system, 5) data is moved to large scale compute resources where Swift-based analysis pipeline is run to fit a crystal structure to the observed image.

While the APS use case described above achieves the goals of its users, there is potential for significant inefficiency, unreported failures and even errors due to the complexity of dealing with several terabytes of data and a number of different operations. Modeling the whole life cycle of data in an APS experiment, from end to end, gives the ability to expose what happens inside the three systems that compose the infrastructure: Swift, Globus Online and Globus Dataset Catalogs. Based on the APS life cycle model, we design a *Data Surveillance Framework*, which allows Scientist to fully monitor their data-sets processing with the following features:

- *Progress Monitoring*, which consists in: *i*) generate reports on progress, get relevant notification when a workflow fully completes, *ii*) get a single relevant notifications when several related events occurred in different systems; *iii*) identify bottlenecks in executions, allowing backtracking the chain of causality, helping to fix the problem at runtime and optimizing the workflow for future executions;

- *Automation*: The workflow used by APS scientists requires human interventions to progress between stages and to recover from unexpected events. Such interventions cannot be integrated in a traditional workflow system, because they reside a level of abstraction above workflow systems. The data-surveillance framework allows to coordinate between systems, and eventually automate several tasks that were performed manually: e.g start the meta-data extraction script when files have been correctly transferred.

- *Sharing*: accelerating data sharing with the community by pushing notifications to collaborators and colleagues. We believe the best way for scientists to automatically integrate new datasets in their workflows is to rely on widely used tools—such as Twitter that can be furthermore easily integrated with other systems as well.

- *Recovery*: Even if systems may individually have failure recovery features, they cannot detect all errors because they lack a global vision of the entire process. Thus, when unexpected events occur, systems often fail ungracefully, leaving the scientists as the only one able to resolve the problem through costly manipulations.

5.4 Conclusion

Handling data-intensive science is pushing the Desktop Grid paradigm to its limits. This has forced us to revisit the Desktop Grid architecture by breaking the monolithic server in a set of independent services to catalog, store, transfer and schedule data and by integrating P2P protocols when a centralized approach might induce a performance bottleneck.

We proposed new abstractions, which help developers to perform complex tasks associated with large scale data management, such as life cycle, transfer, placement, replication, fault tolerance, storage and processing. Particular care was taken to obtain well principled foundation to define the data life cycle. After several years of development, we now have a software portfolio which implements these abstractions: BitDew, MapReduce/BitDew, and Active Data.

These frameworks have been designed to cope with a large number of volatile resources, and they have the following high-level features: multi-protocol file transfers, data scheduling, automatic replication, data privacy, parallel processing, data affinity and transparent data placement. Moreover, they can take advantage of Cloud storage to mitigate Desktop resources volatility.

These frameworks make possible many usage scenarios that were not feasible or difficult to perform before on Desktop Grid or on hybrid DCIs: provide reliable storage

from hybrid DCIs with configurable storage strategies, organize complex communication pattern transfer with replicated data, realtime data surveillance framework, incremental MapReduce processing, and more. We showed that although MapReduce is significantly more complex than traditional Bag-of-Tasks application, it is possible to build an efficient and secure runtime to enable data-intensive application on Desktop Grid.

Active Data is a new paradigm for data life cycle management and we are just starting to explore the possibilities of the model. Here are some open perspectives:

- Big-Data deployment on the Cloud. Asma Ben Cheick, PhD student advised by Heithem Abbes from the University of Tunisia, is proposing a system that takes the data life cycle as a specification for deploying the required data management systems on IaaS infrastructures.

- Big Data inter-systems optimization. With Haiwu He, who is now Professor at the Chinese Academy of Sciences, in Beijing, we are investigating the use of Active Data to instrument some elements of the Apache Big Data stack (HDFS, Hadoop, PIG, HBase etc.) and provide generic inter-system optimizations.

- Data traceability and provenance. Once a data life cycle is known and the software involved have been made "Active Data-aware", it is trivial to record information about all the events that happened to the data set. Some possible applications are: *i*) data resource consumption traceability, that can allow for global energy optimization (with Laurent Lefevre), and *ii*) data derivation history, that can allow for workflow optimizations (with Frédéric Suter).

Chapter 6

Conclusion and Perspectives

> In summary, who say Liberty, say Federation or say nothing.
>
> *(Joseph Proudhon (1809-1865))*

6.1 Conclusion

The Desktop Grid paradigm has evolved considerably over a decade. It started as a little silly idea to become full-scale and sustainable systems used on a daily basis by large scientific communities. An evidence of this transformation is the International Desktop Grid Federation (IDGF)[1], established on the base of FP7 EDGI and FP7 EDGeS European projects. IDGF now brings together more than 40 institutions worldwide. I believe that students, postdocs, and engineers who worked under my supervision, and myself have contributed significantly to close the question of the feasibility of Desktop Grid Computing.

Collectively, researchers and practitioners of Desktop Grid have been able to come together in a community, albeit small, but active, passionate, and which has been able to provide the means of its ambitions by organizing regular workshops and user group meetings and by taking advantage of international opportunities for collaboration. On my side, I had a lot of fun to animate this community, for instance by co-organizing the GP2PC, PCGRID and MAPREDUCE workshops and by co-editing the "Desktop Grid Computing" book.

As a scientist, I have been fortunate to have an awesome playground at my disposal to develop my research. What is really surprising is the variety of scientific methods we elaborated to develop, evaluate, and validate our hypothesis and solutions. Methods range from system observation and resource characterization to platform simulation and software emulation on Grid5000. To obtain realistic and reproducible experimental conditions, it has been necessary to build a special purpose experimental testbed. DSL-Lab has been Paul Malécot's core PhD thesis contribution. I happened sometimes to

[1] International Desktop Grid Federation: `http://desktopgridfederation.org`

be terribly envious of my colleagues doing Cloud and Cluster computing for the ease of access to experimental testbeds. On the other hand, it gave me the opportunity to explore this vast and still mysterious territory, which is the Internet. Thanks to Derrick Kondo's postdoc work, we now understand better the impact of node volatility on Desktop Grid performance.

From a scientific point of view, it is extremely gratifying to see that simple ideas proposed years ago, are still valid. This is the case with XtremWeb architectural principles that were established during my PhD thesis: computing on volatile resources, fault-tolerance through replication and host failure detection, and push/pull scheduling protocol. Many algorithms are covered in the manuscript that improve this architecture on several aspects : scheduling, QoS, security, programming model, and more. We also proposed new software for Desktop Grid Computing : BitDew, an environment for large scale Data management on Desktop Grid and Cloud (Haiwu He's postdoc); SpeQulos, a QoS service for Best-effort infrastructures (Simon Delamare's postdoc); and the first environment for MapReduce computing on Desktop Grid (Bing Tang's postdoc).

Thanks to these breakthrough, Desktop Grid systems can now offer a user experience close to the regular Distributed Computing Infrastructures (DCI), such as Cloud and Grid systems. In addition, we addressed many interoperability issues between Desktop Grids and other DCIs, such as support for virtualization technologies to improve scientific application portability, Grid $\leftrightarrow$ Desktop Grid bridge, support for Grid standards with respect to user authentications, logging and bookkeeping, job submissions, file transfer protocols, and so forth.

The consequence is that scientists now have at their disposal several kinds of DCIs, that can be used simultaneously to run their Grand Challenge applications. We call this assemblage of Grids, Desktop Grids and Clouds, an *Hybrid Distributed Computing Infrastructure (Hybrid DCI)*. Computing on Hybrid DCI introduces many challenges as the infrastructure and the computing resources may be very heterogeneous in term of power efficiency, cost of usage, reliability, trust, usage paradigm, resource management, geographical location, and more. The manuscript presents several results which addressed the issues of using this emerging class of infrastructure efficiently. In particular, a promising research direction is the Promethee Scheduler (Mircea Moca's contribution), which combines a pull-based scheduler with multi-criteria decision and user satisfaction oriented methods.

The manuscript reports on several significant progresses towards Data-intensive applications on Desktop Grids and Hybrid DCIs. We proposed new abstractions for large scale data management and implemented these abstractions in several middleware. These abstractions allow to develop complex data-oriented scenarios, such as: configurable storage for Hybrid DCIs, data surveillance framework for a complex physics workflow, MapReduce runtime environment for Hybrid DCI, and more. To undertake this riskier research, I led and participated to several national and international projects: ANR DSLLAB, ANR Clouds@Home, ANR MapReduce, ANR CloudPower, France-Japan Sakura P2PLab.

The latest evolution of this research leads us to have a more comprehensive view of the interactions between large scientific datasets and the complex infrastructures which

handle them. The proposition of Anthony Simonet's thesis is to use *data life cycle* as a new abstraction for data management. His contributions are a well principled model that allows for a unified view of data life cycle across heterogeneous systems and distributed infrastructures, and a programming model that facilitates the development of complex applications to manage large, dynamic and distributed data sets. The first results obtained on a variety of use cases are promising, and we think that Active Data is a good starting point to tackle more complex problems involving the coordination of heterogeneous Big Data software stacks, hybrid DCIs, and more dynamic data sets, such as data streams or large graphs.

6.2 Perspectives

One of the specific features of Desktop Grid Computing that we do not have addressed so far is the lack of certainty that one can have on its future. If we keep on thinking of Desktop Grid Computing as a Desktop PC tower running the SETI@Home screensaver while happily participating in global warming, then in this case, we should no longer call this Volunteer Computing but instead "Vintage Computing".

Let's now look at some possible radical evolutions of the context, i.e infrastructures, technologies and applications, in order to draw some perspectives for the discipline and propose some future research directions.

The Death of Desktop PCs. So, this big, ugly, and energy-hungry thing dies – although we must be cautious here as it seems that the tablet market decreases more than the Desktop PCs market – ; then, the question is: is it going to be reincarnated? and how? According to Wikipedia, a Personnal Computer is : *"a general-purpose computer whose size, capabilities and original sale price make it useful for individuals, and is intended to be operated directly by an end-user"*. Following this definition, we can already see many reincarnation of this machine around us: smartphones, tablets, TV set-top boxes and in a very next future, camera, car entertainment systems, and a lot more. Actually, this evolution, has been anticipated almost from the beginning of XtremWeb. In the early 2000s, we acquired ARM-based PDAs on which we executed a ray tracing application distributed through XtremWeb. And new opportunities appear constantly. At the CES'2015 keynote address, Intel CEO Brian Krzanich, announced the "Curie" SoC, a low-power 32-bits Quark (2 GHz) processor with embedded sensors, bluetooth, Flash memory and RAM, that is the size of a jacket button. This platform is likely to prefigure what will dominate the wearable and IoT market.

Thus, we are entering a new era where we have to take into account new devices, not only because they are likely to perform computationally intensive tasks, but more certainly because they will be able to produce or acquire data, while being powerful enough to handle part the data processing. Of course, there is a practical limit to this concept: these devices have a short battery life, and using the CPU or network for data analysis or transfer may shorten it even further.

Thus, the challenge is how to model the energy footprint of running large and distributed applications, considering both compute-intensive and data-intensive ones and

comparing diverse scenarios, which involves traditional Desktop resources, mobile devices and sensors, and Cloud infrastructures, in order to understand what are the infrastructure mixes that lead to a global power consumption efficiency; while enforcing end-user device usability.

Do we still need XtremWeb ? Desktop Grid Computing consists in a number of technologies that have been developed to allow to run HTC workload on a non-dedicated infrastructure. There is a trend in pushing the machines out of the data-center walls to lower the power and the cooling cost [228]: some are designing a micro-data center on the building roof powered by solar panels [229]; some others are transforming a data-center in the building basement in a furnace [230]. For example, in the scope of the ANR CloudPower project, we have been working with the French enterprise Qarnot Computing. Qarnot Computing is designing a product called Q.Rad, which is a heater embedding high performance processors as a heat source. Q.Rad are installed in each room of an individual home and the heat is produced when the Q.Rad is processing a workload distributed by the Qarnot scheduler.

Because Desktop Grid systems have been designed to opportunistically take advantage of unused or underused resources, they are good candidates for managing such kind of infrastructures, in particular when augmented with multi-criteria scheduler (see Section 4.3). However there are several challenges to address to make Desktop Grid fully *energy-aware*. First a predictive model of host machine energy consumption is needed, to avoid drying out the device battery. Desktop Grid schedulers require a finer control of the workload that can be adjusted according to the energy available, in the case of energy-opportunistic computing or to the energy produced, in the case of a furnace or heater. Finally, for Hybrid DCI, new scheduling algorithms are expected that can distribute data and computations according to performance vs. power efficiency criteria, while taking into account the specific characteristic of each infrastructures: reliability, usage costs, etc.

Towards Data Infrastructure It is a commonplace, but we are entering the era of Big Data, a considerable phenomena that is impacting the whole process of scientific discovery [209]. However, the phenomena is not limited to some fields of Big Science. The methods and know-how to collect information from various sources, to recognize patterns and extract meaning out of vast data quantities, to access efficient computing resources, to share and collaborate on data-sets are critical in this new way of distilling scientific insights. The challenge of Big Data cannot be restricted to the question of providing larger storage capacity and scalable computing facilities. One radical way of entering this area is to consider the *dataset as the infrastructure*, i.e the data should stands as a layer between the user and the infrastructure.

However, challenges exist, which prevent to fully exploit the value of the scientific data-sets, in three key facets of Data-intensive Sciences: Data infrastructures, Data management and analysis workflows, and collaborative e-Sciences.

Scaling-up Data infrastructure is a great challenge that goes beyond optimizing each component of the whole system. The reason is that we have a limited understanding of the interactions between data-sets and e-infrastructure, which precludes tighter coordination between the various systems involved in data management. This is even more

complex when we consider dynamic dataset widely distributed on sensor networks or mobile devices.

Data scientists are facing data analysis workflows that are becoming increasingly more complex and now requires highly qualified engineers able to mix parallel computing, statistical programming as well as scripting languages to glue the various tools in an ad-hoc way. The second challenge is to provide solution to mitigate the overwhelming sum of human tasks that are not sufficiently automated, and the lack of high-level languages or programming models to express how systems should cooperate.

A crucial role of data-sets is that they are a vehicle for collaborative work, and increasingly, become communication tool between the disciplines. A popular example is the Kaggle web site, which by designing "contest" around datasets has been highly successful for popularizing interesting data analysis problems to the data mining and machine learning community. The third challenge is to further develop collaborative data-intensive science, i.e scientist who share common interest in data-sets would be able to exchange information, share analysis and programs, track similar data-set usage and receive recognition for publicizing data.

I think that Active Data offers us the good level of abstraction to tackle all the aforementioned challenges. However, we'll have to adapt our legacy Desktop Grid approach that was well adapted to data processing with a large to medium granularity to much finer grain processing, required by data stream or graph computing.

Desktop Grid Computing might not be the most adequate term to describe this new computing paradigm, and we should probably start to look for a new name. Any suggestions ?

Publications

Thesis

[T1] G. Fedak. *XtremWeb: une plate-forme pour l'étude expérimentale du calcul global pair-à-pair*. PhD thesis, Université Paris XI, Orsay 2003.

Edition: Book, Journal and Proceedings

[E2] G. Fedak, K. Li, and K. Lin, editors. *Journal on Concurrency and Computation. Special Issue: Combined Special Issue of MapReduce and its Applications & Advanced Topics on Wireless Sensor Networks*, volume 25. John Wiley & Sons, Ltd, January 2013.

[E3] C. Cérin and G. Fedak, editors. *Desktop Grid Computing*. Chapman & All/CRC Press, May 2012.

[E4] S. Jha, N. G. Felde, R. Buyya, and G. Fedak, editors. *Proceedings of 12th IEEE/ACM International Conference on Grid Computing (Grid 2011)*, Lyon, France, 2011.

Book Chapters

[B5] A. Lebre, J. Pastor, M. Bertier, F. Desprez, J. Rouzaud-Cornabas, C. Tedeschi, A.-C. Orgerie, F. Quesnel, and G. Fedak. Beyond The Clouds, How Should Next Generation Utility Computing Infrastructures Be Designed ? In Z. Mahmood, editor, *Cloud Computing: Challenges, Limitations and R&D Solutions*. Springer, November 2014.

[B6] H. Lin, W.-C. Feng, and G. Fedak. Data-Intensive Computing on Desktop Grids. In C. Cérin and G. Fedak, editors, *Desktop Grid Computing*, pages 237–259. Chapman & All/CRC Press, 2012.

[B7] S. Delamare and G. Fedak. Towards Hybridized Clouds and Desktop Grid Infrastructures. In C. Cérin and G. Fedak, editors, *Desktop Grid Computing*, pages 261–285. Chapman & All/CRC Press, 2012.

[B8] L. Rodero-Merino, G. Fedak, and A. Muresan. MapReduce and Hadoop. In L. M. Vaquero, J. Hierro, and J. Cáceres, editors, *Open Source Cloud Computing Systems: Practices and Paradigms*. IGI Global, 2011.

[B9] F. Cappello, G. Fedak, D. Kondo, P. Malécot, and A. Rezmerita. Chapter 3: Desktop Grids: From Volunteer Distributed Computing to High Throughput Computing Production Platforms. In K.-C. Li, C.-H. Hsu, L. T. Yang, J. Dongarra, and H. Zima, editors, *Handbook of Research on Scalable Computing Technologies*, pages 31–61. IGI Global, July 2009.

[B10] F. Bouabache, T. Herault, G. Fedak, and F. Cappello. *A Distributed and Replicated Service for Checkpoint Storage*, volume 7 of *CoreGRID Books: Making Grids Work*, chapter Checkpointing and Monitoring, pages 293–306. M. Danelutto, P. Fragopoulou and V. Getov, eds. Springer, 2008. [1].

[B11] F. Cappello, G. Fedak, T. Morlier, and O. Lodygensky. Des systèmes client-serveur aux systèmes pair à pair. In I. Comyn and W. Jacky-Akoka, editors, *Encyclopédie de l'informatique et des systèmes d'information*, pages 195–210. Vuibert, 2007.

[B12] F. Cappello, A. Djilali, G. Fedak, C. Germain, O. Lodygensky, and V. Néri. Xtremweb: une plate-forme de recherche sur le calcul global et pair à pair. In F. Baud, editor, *Calcul réparti à grande échelle Metacomputing*. Hermes Science, Lavoisier, 2002.

[1] extended version of [C60]

Journal Articles

— International Journal Articles —

[J13] A. Simonet, G. Fedak, and M. Ripeanu. Active Data: A Programming Model to Manage Data Life Cycle Across Heterogeneous Systems and Infrastructures. *Future Generation in Computer Systems*, 2015. Accepted, to appear.

[J14] M. Moca, C. Litan, G. C. Silaghi, and G. Fedak. Multi-Criteria and Satisfaction Oriented Scheduling for Hybrid Distributed Computing Infrastructures. *Future Generation in Computer Systems*, 2015. Accepted, to appear.

[J15] B. Tang, H. He, and G. Fedak. HybridMR: A New Approach for Hybrid MapReduce Combining Desktop Grid and Cloud Infrastructures. *Concurrency Practice and Experience*, 2015. Accepted, to appear.

[J16] L. Costa, H. Yang, E. Vairavanathan, A. Barros, K. Maheshwari, G. Fedak, D. Katz, M. Wilde, M. Ripeanu, and S. Al-Kiswany. The Case for Workflow-Aware Storage: An Opportunity Study using MosaStore. *Journal of Grid Computing*, pages 1–19, 2014. appeared online.

[J17] S. Delamare, G. Fedak, D. Kondo, and O. Lodygensky. SpeQuloS : A QoS Service for Hybrid and Elastic Computing Infrastructures. *Journal of Cluster Computing*, 17(1):79–100, 2014. [2].

[J18] G. Antoniu, J. Bigot, C. Blanchet, L. Bougé, F. Briant, F. Cappello, A. Costan, F. Desprez, G. Fedak, S. Gault, K. Keahey, B. Nicolae, C. Pérez, A. Simonet, F. Suter, B. Tang, and R. Terreux. Scalable Data Management for MapReduce-Based Data-Intensive Applications: a View for Cloud and Hybrid Infrastructures. *International Journal on Cloud Computing*, 2(2-3), January 2013. [3].

[J19] O. Gatsenko, O. Baskova, O. Lodygensky, G. Fedak, and Y. Gordienko. Statistical Properties of Deformed Single-Crystal Surface under Real-Time Video Monitoring and Processing in the Desktop Grid Distributed Computing Environment. *Journal of Key Engineering Materials*, Materials Structure & Micromechanics of Fracture VI(465):306–309, January 2011.

[J20] F. Bouabache, T. Herault, G. Fedak, and F. Cappello. Hierarchical Replication Techniques to Ensure Checkpoint Storage Reliability in Grid Environment. *Journal of Interconnection Networks*, 10(4):345–364, 2009.

[J21] E. Urbah, P. Kacsuk, Z. Farkas, G. Fedak, G. Kecskemeti, O. Lodygensky, A. Marosi, Z. Balaton, G. Caillat, G. Gombas, A. Kornafeld, J. Kovacs, H. He, and R. Lovas. EDGeS: Bridging EGEE to BOINC and XtremWeb. *Journal of Grid Computing*, 7(3):335–354, Sept. 2009.

[J22] G. Fedak, H. He, and F. Cappello. BitDew: A Data Management and Distribution Service with Multi-Protocol and Reliable File Transfer. *Journal of Network and Computer Applications*, 32(5):961–975, Sept. 2009. [4].

[J23] F. Costa, L. Silva, G. Fedak, and I. Kelley. Optimizing Data Distribution in Desktop Grid Platforms. *Parallel Processing Letters*, 18(3):391–410, September 2008. [5].

[J24] Z. Balaton, Z. Farkas, G. Gombas, P. Kacsuk, R. Lovas, A. C. Marosi, A. Emmen, G. Terstyanszky, T. Kiss, I. Kelley, I. Taylor, O. Lodygensky, M. Cardenas-Montes, G. Fedak, and F. Araujo. EDGeS: the Common Boundary Between Service and Desktop Grids. *Parallel Processing Letters*, 18(3):433–453, September 2008. [6]

[2] extended version of [C40], Special Issue: Selected Papers from the HPDC'12 Conference

[3] extended version of [C42] Special Issue: Selected Papers from the ICACON'12 Conference

[4] extended version of [C56], Invited Paper from the UPGRADECN-08 Workshop Keynote

[5] extended version of [C63]

[6] Similar versions of the paper presenting the EDGeS project have been published in several conferences ([C54, C59, C61, C62]).

[J25] B. Wei, G. Fedak, and F. Cappello. Towards Efficient Data Distribution on Computational Desktop Grids with BitTorrent. *Future Generation Computer Systems*, 23(7):983–989, Nov. 2007. [7].

[J26] D. Kondo, G. Fedak, F. Cappello, A. A. Chien, and H. Casanova. Resource Availability in Enterprise Desktop Grids. *Future Generation Computer Systems*, 23(7):888–903, Aug. 2007. [8].

[J27] F. Cappello, S. Djilali, G. Fedak, T. Herault, F. Magniette, V. Néri, and O. Lodygensky. Computing on Large Scale Distributed Systems: XtremWeb Architecture, Programming Models, Security, Tests and Convergence with Grid. *Future Generation Computer Systems*, 21(3):417–437, mar 2005.

[J28] G. Bosilca, G. Fedak, and F. Cappello. OVM: Out-of-Order Execution Parallel Virtual Machine. *Future Generation Computer Systems*, 18(4):525–537, mar 2002. [9].

— French Journal Articles —

[J29] O. Richard, G. Fedak, and T. Devanneaux. Approche multiflot pour le partage des réseaux rapides dans les clumps. *Technique et Science Informatiques*, 2000/6, 1999. [10].

Conferences and Workshops with Proceedings

— International Conference Articles —

[C30] N. Gordienko, O. Lodygensky, G. Fedak, and Y. Gordienko. Synergy of volunteer measurements and volunteer computing for effective data collecting, processing, simulating and analyzing on a worldwide scale. In IEEE, editor, *38th International Convention on Information and Communication, Technology, Electronics and Microelectronics*, Opatija, Croatia, May 2015.

[C31] A. Simonet, K. Chard, G. Fedak, and I. Foster. Active Data to Provide Smart Data Surveillance to E-Science Users. In *Proceedings of IEEE Euromicro-PDP'15*, Turku, Finland, March 2015. to appear.

[C32] J. C. S. dos Anjos, G. Fedak, and C. F. R. Geyer. BIGhybrid - A Toolkit for Simulating MapReduce in Hybrid Infrastructures. In *Workshop on Parallel and Distributed Computing for Big Data Applications (WPBA'14)*, Paris, October 2014.

[C33] A. B. Cheikh, H. Abbes, and G. Fedak. Ensuring Privacy for MapReduce on Hybrid Clouds Using Information Dispersal Algorithm. In *7th International Conference on Data Management in Cloud, Grid and P2P Systems (Globe '14)*, volume 8648 of *Lecture Notes on Computer Science*, pages 37–48, Munich, Germany, September 2014. Springer Verlags.

[C34] B. Tang, H. He, and G. Fedak. Parallel Data Processing in Dynamic Hybrid Computing Environment Using MapReduce. In *14th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP'14)*, volume 8631 of *Lecture Notes on Computer Science*, pages 1–14, Dalian, China, August 2014. Springer Verlags.

[C35] A. Simonet, G. Fedak, M. Ripeanu, and S. Al-Kiswany. Active Data: A Data-Centric Approach to Data Life-Cycle Management. In *8th Parallel Data Storage Workshop (PDSW), Proceedings of SC13 workshops*, pages 39–44, Denver, USA, November 2013. ACM.

[C36] M. Moca, C. Litan, G. C. Silaghi, and G. Fedak. Advanced Promethee-based Scheduler Enriched with User-Oriented Methods. In *In Proceedigs of the 10th IEEE Conference on Economics of Grids, Clouds, Systems, and Services (GECON 2013)*, volume 8193 of *Lecture Notes in Computer Science*, pages 161–172, Zaragoza, Spain, September 2013. Springer International Publishing.

[7] extended version of [C71], Special Issue: Selected Papers from the ISPDC'07 Conference
[8] extended version of [C66]
[9] extended version of [C79] Special Issue: Selected Papers from the CCGRID'01 Conference
[10] Special Issue: Selected Papers from the RENPAR Conference

[C37] M. Moca and G. Fedak. Using Promethee Methods for Multi-Criteria Pull-based Scheduling on DCIs. In *Proceedings of the 8th IEEE International Conference on eScience (eScience'12)*, Chicago, USA, October 2012.

[C38] M. Labidi, B. Tang, G. Fedak, M. Khemakem, and M. Jemni. Scheduling Data and Task on Data-Driven Master/Worker Platform. In *The 13th IEEE International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT-12)*, Beijing, China, December 2012.

[C39] L. Lu, H. Jin, X. Shi, and G. Fedak. Assessing MapReduce for Internet Computing: a Comparison of Hadoop and BitDew-MapReduce. In *Proceedings of the 13th ACM/IEEE International Conference on Grid Computing (Grid 2012)*, Beijing, China, September 2012.

[C40] S. Delamare, G. Fedak, D. Kondo, and O. Lodygensky. SpeQuloS: A QoS Service for BoT Applications Using Best Effort Distributed Computing Infrastructures. In *Proceedings of the 21st ACM International Symposium on High Performance Distributed Computing (HPDC'12)*, pages 173–186, Delft, The Netherlands, June 2012.

[C41] B. Tang and G. Fedak. Analysis of Data Reliability Tradeoffs in Hybrid Distributed Storage Systems. In *Proceedings of Parallel and Distributed Symposium Workshops and PhD Forum (IPDPSW'12), 17th IEEE International Workshop on Dependable Parallel, Distributed and Network-Centric Systems (DPDNS'12)*, Shanghai, China, May 2012.

[C42] G. Antoniu, J. Bigot, C. Blanchet, L. Bougé, F. Briant, F. Cappello, A. Costan, F. Desprez, G. Fedak, S. Gault, K. Keahey, B. Nicolae, C. Pérez, A. Simonet, F. Suter, B. Tang, and R. Terreux. Towards Scalable Data Management for Map-Reduce-based Data-Intensive Applications on Cloud and Hybrid Infrastructures. In *The 1st International IBM Cloud Academy Conference (ICA CON 2012)*, North Carolina, USA, April 2012.

[C43] O. Lodygensky, E. Urbah, S. Dadoun, A. Simonet, G. Fedak, S. Delamare, D. Kondo, L. Duflot, and X. Garrido. FlyingGrid : from Volunteer Computing to Volunteer Cloud. In *poster in Computing in Hign Energy and Nuclear Physics (CHEP'12)*, New York, USA, 2012.

[C44] M. Moca, G. C. Silaghi, and G. Fedak. Distributed Results Checking for MapReduce on Volunteer Computing. In *Proceedings of IPDPS'2011, 4th Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2011)*, Anchorage, Alaska, May 2011.

[C45] B. Tang, M. Moca, S. Chevalier, H. He, and G. Fedak. Towards MapReduce for Desktop Grid Computing. In *Fifth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC'10)*, pages 193–200, Fukuoka, Japan, November 2010. IEEE.

[C46] G. Fedak, J.-P. Gelas, T. Hérault, V. Iniesta, D. Kondo, L. Lefèvre, P. Malécot, L. Nussbaum, A. Rezmerita, and O. Richard. DSL-Lab: a Platform to Experiment on Domestic Broadband Internet. In *Proceedings of the The 9th IEEE International Symposium on Parallel and Distributed Computing (ISPDC'10)*, pages 141–148, Istanbul, Turquey, July 2010.

[C47] O. Gatsenko, O. Baskova, O. Lodygensky, G. Fedak, and Y. Gordienko. Statistical Properties of Deformed Single-Crystal Surface under Real-Time Video Monitoring and Processing in the Desktop Grid Distributed Computing Environment. In *Proceedings of the Sixth International Conference on Materials Struncture and Micromechanics of Fracture (MSMF6)*, Brno, Czech Republic, June 2010.

[C48] H. He, G. Fedak, P. Kacsuk, Z. Farkas, Z. Balaton, O. Lodygensky, E. Urbah, G. Caillat, and F. Araujo. Extending the EGEE Grid with XtremWeb-HEP Desktop Grids. In *Proceedings of CCGRID'10, 4th Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2010)*, pages 685–690, Melbourne, Australia, May 2010.

[C49] O. Gatsenko, O. Baskova, G. Fedak, O. Lodygensky, and Y. Gordienko. Porting Multiparametric MATLAB Application for Image and Video Processing to Desktop Grid for High-Performance Distributed Computing. In *Proceedings of 3rd Grid Experience workshop - Desktop Grid Applications for eScience and eBusiness*, Alemere, Nederlands, March 2010. EnterTheGrid.

[C50] O. Gatsenko, O. Baskova, G. Fedak, O. Lodygensky, and Y. Gordienko. Kinetics of Defect Aggregation in Materials Science Simulated in Desktop Grid Computing Environment Installed in Ordinary Material Science Lab. In *Proceedings of 3rd Grid Experience workshop - Desktop Grid Applications for eScience and eBusiness*, Alemere, Nederlands, March 2010. EnterTheGrid.

[C51] A. C. Marosi, P. Kacsuk, G. Fedak, and O. Lodygensky. Sandboxing for Desktop Grids Using Virtualization. In *Proceedings of the 18th Euromicro International Conference on Parallel, Distributed and Network-Based Computing PDP 2010*, pages 559–566, Pisa, Italy, February 2010.

[C52] G. Fedak. Recent advances and research challenges in desktop grid and volunteer computing. In *Proceedings of the EuroPAR 2009 Workshops, CoreGrid ERCIM Working Group Workshop on Grids, P2P and Service Computing*, pages 171–185, Delft, Netherlands, Aug 2009. LNCS.

[C53] H. He, G. Fedak, B. Tran, and F. Cappello. BLAST Application with Data-aware Desktop Grid Middleware. In *Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid CCGRID'09*, pages 284–291, Shanghai, China, May 2009.

[C54] G. Caillat, O. Lodygensky, G. Fedak, H. He, Z. Balaton, Z. Farkas, G. Gombas, P. Kacsuk, R. Lovas, A. C. Maros, I. Kelley, I. Taylor, G. Terstyanszky, T. Kiss, M. Cardenas-Montes, A. Emmen, and F. Araujo. EDGeS: The art of bridging EGEE to BOINC and XtremWeb. In *Proceedings of Computing in High Energy and Nuclear Physics (CHEP'09) (Abstract)*, Prague, Czech Republic, March 2009. [6].

[C55] P. Kacsuk, Z. Farkas, and G. Fedak. Towards Making BOINC and EGEE Interoperable. In *Proceedings of 4th IEEE International Conference on e-Science (e-Science 2008), International Grid Interoperability and Interoperation Workshop 2008 (IGIIW 2008)*, pages 478–484, Indianapolis, USA, December 2008.

[C56] G. Fedak, H. He, and F. Cappello. BitDew: A Programmable Environment for Large-Scale Data Management and Distribution. In *Proceedings of the ACM/IEEE SuperComputing Conference (SC'08)*, pages 1–12, Austin, USA, November 2008.

[C57] G. Fedak, H. He, and F. Cappello. A File Transfer Service with Client/Server, P2P and Wide Area Storage Protocols. In *Proceedings of the First International Conference on Data Management in Grid and P2P Systems (Globe'2008)*, LNCS, pages 1–11, Turin, Italy, September 2008. Springer Verlag.

[C58] G. Caillat, G. Fedak, H. He, O. Lodygensky, and E. Urbah. Towards a Security Model to Bridge Internet Desktop Grids and Service Grids. In *Proceedings of the Euro-Par 2008 Workshops (LNCS), Workshop on Secure, Trusted, Manageable and Controllable Grid Services (SGS'08)*, Las Palmas de Gran Canaria, Spain, August 2008. 247–259.

[C59] G. Fedak, H. He, O. Lodygensky, Z. Balaton, Z. Farkas, G. Gombas, P. Kacsuk, R. Lovas, A. C. Maros, I. Kelley, I. Taylor, G. Terstyanszky, T. Kiss, M. Cardenas-Montes, A. Emmen, and F. Araujo. EDGeS:A Bridge Between Desktop Grids and Service Grids. In *IEEE computing society Proceeding of the 3rd ChinaGrid Annual Conference*, pages 1–9, Dunhuang, Gansu, China, August 2008. [6].

[C60] F. Bouabache, T. Herault, G. Fedak, and F. Cappello. Hierarchical Replication Techniques to Ensure Checkpoint Storage Reliability in Grid Environments. In *Proceedings of 8th IEEE International Symposium on Cluster Computing and the Grid CCGRID'08*, pages 475–483, Lyon, France, may 2008.

[C61] M. Cárdenas-Montes, A. Emmen, A. C. Marosi, F. Araujo, G. Gombás, G. Fedak, I. Kelley, I. Taylor, O. Lodygensky, P. Kacsuk, R. Lovas, T. Kiss, Z. Balaton, Z. Farkas, and G. Terstyanszky. EDGeS: Bridging Desktop and Service Grids. In *Proceedings of IBERGRID, 2nd Iberian Grid Infrastructure Conference*, pages 212–226, Porto, Portugal, May 2008. [6].

[C62] Z. Balaton, Z. Farkas, G. Gombas, P. Kacsuk, R. Lovas, A. C. Marosi, A. Emmen, G. Terstyanszky, T. Kiss, I. Kelley, I. Taylor, O. Lodygensky, M. Cardenas-Montes, G. Fedak, and F. Araujo. EDGeS: the Common Boundary Between Service and Desktop Grids. In *Proceedings of the CoreGrid Integration Workshop (CGIW08)*, Hersonissos-Crete, Greece, April 2008. [6].

[C63] F. Costa, L. Silva, G. Fedak, and I. Kelley. Optimizing the Data Distribution Layer of BOINC with BitTorrent. In *Proceedings of IPDPS'08, 2nd Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2008)*, pages 1–8, Miami, Florida, apr 2008.

[C64] D. Kondo, F. Araujo, P. Malecot, P. Domingues, L. M. Silva, G. Fedak, and F. Cappello. Characterizing Result Errors in Internet Desktop Grids. In *European Conference on Parallel and Distributed Computing EuroPar'07*, Rennes, France, August 2007. **Best paper award**.

[C65] F. Bouabache, T. Herault, G. Fedak, and F. Cappello. A Distributed and Replicated Service for Checkpoint Storage. In *CoreGRID Workshop on Grid programming model, Grid and P2P systems architecture and Grid systems, tools and environments*, Heraklion, Greece, June 2007.

[C66] D. Kondo, G. Fedak, F. Cappello, A. A. Chien, and H. Casanova. On Resource Volatility in Enterprise Desktop Grids. In *Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing (eScience'06)*, pages 78–86, Amsterdam, Netherlands, December 2006.

[C67] D. Anderson and G. Fedak. The Computational and Storage Potential of Volunteer Computing. In *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, pages 73–80, Singapore, May 2006.

[C68] D. Kondo, B. Kindarji, G. Fedak, and F. Cappello. Towards Soft Real-Time Applications on Enterprise Desktop Grids. In *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID '06)*, pages 65–72, Singapore, May 2006.

[C69] P. Malécot, D. Kondo, and G. Fedak. XtremLab: A System for Characterizing Internet Desktop Grids. In *Poster in The 15th IEEE International Symposium on High Performance Distributed Computing HPDC'06*, Paris, France, June 2006.

[C70] B. Wei, G. Fedak, and F. Cappello. Scheduling Independent Tasks Sharing Large Data Distributed with BitTorrent. In *6th IEEE/ACM International Workshop on Grid Computing*, pages 219–226, Seattle, USA, Nov. 2005.

[C71] B. Wei, G. Fedak, and F. Cappello. Collaborative Data Distribution with BitTorrent for Computational Desktop Grids. In *Proceedings of the The 4th IEEE International Symposium on Parallel and Distributed Computing (ISPDC'05)*, pages 250–257, Lille, France, June 2005.

[C72] S. Djilali, T. Herault, O. Lodygensky, T. Morlier, G. Fedak, and F. Cappello. RPC-V: Toward Fault-Tolerant RPC for Internet Connected Desktop Grids with Volatile Nodes. In *Proceedings of the ACM/IEEE SuperComputing Conference (SC'04)*, pages 39–, Pittsburgh, USA, Nov. 2004.

[C73] O. Lodygensky, G. Fedak, F. Cappello, V. Neri, M. Livny, and D. Thain. XtremWeb & Condor : Sharing Resources Between Internet Connected Condor Pools. In *Proceedings of CCGRID'2003, Third International Workshop on Global and Peer-to-Peer Computing (GP2PC'03)*, pages 382–389, Tokyo, Japan, 2003. IEEE/ACM.

[C74] O. Lodygensky, G. Fedak, V. Neri, C. Germain, F. Cappello, and A. Cordier. Augernome & XtremWeb : Monte Carlo Computation on a Global Computing Platform. In *Proceedings of CHEP03 Conference for Computing in High Energy and Nuclear Physics*, San Diego, USA, 2003.

[C75] G. Bosilca, A. Bouteillier, F. Cappello, S. Djilali, G. Fedak, C. Germain, T. Herault, P. Lemarinier, O. Lodygensky, F. Magniette, V. Neri, and A. Selhikov. Mpich-v: Toward a scalable fault tolerant mpi for volatile nodes. In *Proceedings of ACM/IEEE International Conference on Supercomputing SC02*, Baltimore, USA, Nov. 2002. IEEE/ACM, IEEE Press.

[C76] A. Selhikov, C. Germain, G. Bosilca, F. Cappello, and G. Fedak. MPICH-CM: a P2P MPI implementation. In *9th EuroPVM/MPI*, Johannes Kepler University, Linz, Austria, September 2002.

[C77] G. Fedak, C. Germain, V. Néri, and F. Cappello. XtremWeb: A Generic Global Computing Platform. In *Proceedings of 1st IEEE International Symposium on Cluster Computing and the Grid CCGRID'2001, Special Session Global Computing on Personal Devices*, pages 582–587, Brisbane, Australia, May 2001. IEEE/ACM, IEEE Press.

[C78] V. N. Cécile Germain, Gilles Fedak and F. Cappello. Global Computing Systems. In *Proceedings of the Third International Conference on Large-Scale Scientific Computing (LSCC'01)*, volume 2179 of *Lecture Notes in Computer Science*, pages 218–227, London, UK, 2001. Springer-Verlag.

[C79] G. Bosilca, G. Fedak, and F. Cappello. OVM: Out-of-order Execution Parallel Virtual Machine. In *Proceedings of 1st IEEE International Symposium on Cluster Computing and the Grid CCGRID'2001*, pages 212–220, Brisbane, Australia, May 2001. IEEE/ACM, IEEE Press. **Best Paper Award**.

[C80] C. Germain, V. Néri, G. Fedak, and F. Cappello. XtremWeb: Building an Experimental Platform for Global Computing. In *Grid'2000 First Internationnal Workshop on Grid Computing*, volume 1971, Bangalore, India, December 2000. IEEE/ACM, Springler-Verlags LNCS.

[C81] G. Bosilca, G. Fedak, O. Richard, and F. Cappello. High Performance Computing with RPC Programming Style. In *Proceedings of the First Myrinet User Group Conference MUG2000*, September 2000.

— French Conference Articles —

[C82] P. Malécot, D. Kondo, and G. Fedak. Xtremlab: Une plateforme pour l'observation et la caractérisation des grilles de pc sur internet. In *Rencontres francophones du parallélisme (Renpar'06)*, Perpignan, France, October 2006.

[C83] G. Bosilca, G. Fedak, T. Herault, and F. Magniette. Evaluation de performance de différentes techniques de confinement d'exécution pour le calcul pair-à-pair. In *Rencontres francophones du parallélisme (Renpar'13)*, Hammamet, Tunisie, Avril 2002.

[C84] G. Fedak. Exécution délocalisée et répartition de charge : une étude expérimentale. In *Rencontres francophones du parallélisme (Renpar'12)*, 2000.

[C85] O. Richard, G. Fedak, and T. Devanneaux. Passage de messages pour grappe de multiprocesseurs et réseau rapide dans un contexte multiflot. In *Rencontres francophones du parallélisme (Renpar'12)*, 1999.

Invited Talks, Tutorials, Challenges and Conferences without Proceedings

---

[T86] G. Fedak. Active data: Data life cycle management across heterogeneous systems and infrastructures. In *Hot Topics in High-Performance Distributed Computing Workshop*, San Jose, California, March 2015. IBM Almaden Research Center.

[T87] G. Fedak. Big data, beyond the data center. In *Cluj Economics and Business Seminar Series (CEBSS)*, University Babes-Bolyai Faculty of Economics and Business Administration, Romania, September 2014. **University Seminary**.

[T88] G. Fedak. Big data, beyond the data center. In *Seminary of Computer Network Information Center, Chinese Academy of Sciences*, Beijing, China, September 2014. Seminary speaker.

[T89] G. Fedak. Unleashing the power of big data and hpc for innovative small business. In *Annual Conference of Zhongguancun Forum*, Beijing China, September 2014. Invited Speaker.

[T90] A. Simonet, G. Fedak, and M. Ripeanu. activedata, un modèle de programmation pour la gestion des cycles de vie des données. In *Calcul intensif et Sciences des données*, Vichy, France, June 2014. Invited Speaker.

[T91] A. Simonet, G. Fedak, K. Chard, and I. Foster. Using active data to provide smart data surveillance to e-science users. In *The Tenth Workshop of the INRIA-Illinois Joint Laboratory on Petascale Computing*, Nice, France, June 2014.

[T92] A. Simonet, G. Fedak, and M. Ripeanu. Active data: A programming model to manage data life cycle across heterogeneous systems and infrastructures. In *Department of Computer Science Seminary*, University of Chicago, USA, November 2013.

[T93] A. Simonet, G. Fedak, and M. Ripeanu. Active data: A programming model to manage data life cycle across heterogeneous systems and infrastructures. In *The Ninth Workshop of the INRIA-Illinois Joint Laboratory on Petascale Computing*, Lyon, France, June 2013. Invited speaker.

[T94] A. Simonet, L. Lu, X. Shi, B. Tang, J.-F. Saray, and G. Fedak. MapReduce on Desktop Grids with BitDew and Active Data. In *Grid5000 Spring School*, Nantes, France, February 2013.

[T95] G. Fedak. MapReduce Runtime Environments: Design, Performance, Optimization. In *Seminar Datenverarbeitung mit Map-Reduce*, University of Heidelberg, Germany, May 2012. Invited speaker.

[T96] G. Fedak. Recent Advances Towards Data Desktop Grids. In *Google Tech Talks*, Mountain View, CA, USA, June 2011. **Google Tech Talk**.

[T97] S. Delamare and G. Fedak. SpeQulos: A Framework for QoS in Unreliable Distributed Computing Infrastructures using Cloud Resources. In *Grid5000 Spring School*, Reims, France, February 2011. **Best Presentation Award**.

[T98] G. Fedak and S. Delamare. SpeQulos: A Framework for QoS in Unreliable Distributed Computing Infrastructures using Cloud Resources. In *Argonne National Laboratory*, Argonne, USA, February 2011. Seminary.

[T99] G. Fedak. Cloud Resource Management and Programming Model. In *in Anniversary Workshop of the INRIA - Alcatel Lucent Bell-Labs joint research lab*, Rocquencourt, France, January 2011. Invited talk.

[T100] G. Fedak. Introduction to MapReduce. In *in Workshop Langages et paradigmes de programmation émergents*, Lyon, France, December 2010. Cluster ISLE. Invited talk.

[T101] G. Fedak. Towards Large Scale Data Processing on Desktop Grid with BitDew and MapReduce. In *Workshop Grilles de calcul : recherches et applications lors de la Conférence Internationale NOTERE'10*, Tozeur, Tunisie, May 2010. Invited Talk.

[T102] B. Tang, G. Fedak, and H. He. The BitDew project : Towards Large Scale Data Processing. In *First France-China Workshop on Virtualization Technologies and Cloud Computing*, Wuhan, China, March 2010. Huazong University of Technology. Invited talk.

[T103] G. Fedak. Recent Advances Towards Data Desktop Grids . In *NetSysLab Seminary*, University of British Columbia, Canada, October 2009. Seminary.

[T104] G. Fedak. Hot Topics in Desktop Grids Research. In *GRAAL GdT, LIP/ENS*, ENS Lyon, France, January 2009. Seminary.

[T105] G. Fedak. BitDew: A Programmable Environment for Large-Scale Data Management and Distribution. In *Innovative Computing Laboratory, Friday Talk*, UTK, Knoxwille, USA, November 2008. Seminary.

[T106] G. Fedak, H. He, and F. Cappello. Keynote: Distributing and Managing Data on Desktop Grids with BitDew. In *Proceedings of High Performance Distributed Computing (HPDC'08), 3rd Workshop on the Use of P2P, GRID and Agents for the Development of Content Networks (UPGRADE-CN'08)*, pages 63–64, Boston, USA, June 2008. **Keynote Speaker**.

[T107] G. Fedak. Bridging XtremWeb with the EGEE Grid. In *1rst EDGeS User Forum and Industry Forum*, Orsay, France, May 2008. Invited talk.

[T108] G. Fedak. Implementing New File Transfer Protocols in BitDew: Amazon S3 Case Study. In *XW'08 : 2nd XtremWeb Users Group Workshop*, Orsay, France, May 2008. Invited talk.

[T109] H. He, G. Fedak, and F. Cappello. Large-Scale Bioinformatic Computing on Data Desktop Grid. In *First IEEE International Scalable Computing Challenge (SCALE 2008) along with CCGRID'08*, Lyon, France, May 2008. **Challenge**.

[T110] G. Fedak. DSLLab : Plate-forme d'expérimentation pour les systèmes distribués à large échelle sur Internet haut-débit. In *Colloque ANR JCJC*, Montpellier, France, May 2007. Invited talk.

[T111] G. Fedak. Towards Data-Intensive Applications on XtremWeb. In *XtremWeb Users Group Workshop*, Hammamet, Tunisia, February 2007. Invited talk.

[T112] G. Fedak, B. Wei, and F. Cappello. Scheduling Independent Tasks Sharing Large Data Distributed with BitTorrent. In *NSF/INRIA Workshop Scheduling for Large-Scale Distributed Platforms*, La Jolla, California, USA, November 2005. Invited talk.

[T113] G. Fedak. XtremWeb: Calcul à Large Echelle. In *Rencontres de la Société de Mathématiques Appliquées*, Evian, France, May 2005. Invited talk.

[T114] G. Fedak. Grand Large Desktop Grid. In *France-Korea Joint Workshop on Grid Computing*, Rennes, France, July 2004. Invited talk.

[T115] G. Fedak. XtremWeb: A Peer-to-Peer Global Computing Experimental Platform. In *Free Software and Open Source Developers Meeting FOSDEM*, Brussel, Belgium, February 2002. Invited talk.

[T116] G. Fedak. XtremWeb: an experimental platform for Global and Peer-to-Peer Computing. In *HEPiX 2001, UNIX users in the High Energy Physics*, Orsay, France, April 2001. Invited talk.

[T117] F. Cappello, G. Fedak, and O. Richard. Systèmes distribués de calcul global et pair à pair. In *Tutoriel à Renpar'2001*, Paris, France, April 2001. **Tutorial**.

Research Reports

[R118] A. Simonet, G. Fedak, and M. Ripeanu. Active Data: A Programming Model for Managing Big Data Life Cycle. Technical Report RR-8062, INRIA, 2012.

[R119] G. Fedak, O. Lodygensky, Z. Farkas, and P. Kacsuk. Prototype of the generic bi-directional service grids to desktop grids bridge. Technical report, Deliverable JRA1.3, EDGeS project European Union, 2009.

[R120] G. Fedak, J.-P. Gelas, T. Hérault, V. Iniesta, D. Kondo, L. Lefèvre, P. Malécot, L. Nussbaum, A. Rezmerita, and O. Richard. DSL-Lab: a Platform to Experiment on Domestic Broadband Internet. Technical Report 7024, INRIA, 2009.

[R121] G. Fedak, O. Lodygensky, and Z. Farkas. Prototypes of bridge from service grids to desktop grids. Technical report, Deliverable JRA1.2, EDGeS project European Union, 2008.

[R122] A. C. Marosi, P. Kacsuk, G. Fedak, and O. Lodygensky. Using Virtual Machines in Desktop Grid Clients for Application Sandboxing. Technical Report TR-0140, Institute on Architectural Issues: Scalability, Dependability, Adaptability, CoreGRID - Network of Excellence, August 2008.

[R123] F. Costa, L. Silva, G. Fedak, and I. Kelley. Optimizing the data distribution layer of boinc with bittorrent. Technical Report TR-0139, Institute on Architectural issues: scalability, dependability, adaptability, CoreGRID Technical Report, June 2008.

[R124] G. Fedak, O. Lodygensky, and Z. Farkas. Prototypes of bridge from desktop grids to service grids. Technical report, Deliverable JRA1.1, EDGeS project European Union, 2008.

[R125] G. Fedak, H. He, and F. Cappello. BitDew: A Programmable Environment for Large-Scale Data Management and Distribution. Technical Report 6427, INRIA, jan 2008.

[R126] F. Boyer, J. Kornas, J.-B. Stefani, N. Parlavanzas, N. de Palma, A. Ouorou, E. Gourdin, N. Amara, R. Krishnaswamy, L. Navarro, R. Brunner, X. Leon, X. Vilajosana, D. Kondo, G. Fedak, P. Malecot, A. Valarakos, A. Papasalouros, G. Vouros, K. Kotis, S. Retalis, J. Quiane-Ruiz, P. Valduriez, and P. Lamarre. D2.1 requirements for grid4all virtual organisations and resource management and state of the art analysis. Technical report, European Union, Grid4All project, June 2007.

[R127] D. Kondo, P. Malecot, G. Fedak, F. Cappello, F. Araujo, L. Silva, and P. Domingues. Characterizing Result Errors in Internet Desktop Grids. Technical Report TR-0040, Institute on System Architecture, CoreGRID - Network of Excellence, October 2006.

Software and Repositories

[W128] S. Delamare and G. Fedak. SpeQulos: A Framework for QoS in Hybrid Distributed Computing Infrastructures. http://spequlos.gforge.inria.fr.

[W129] G. Fedak, H. He, and F. Cappello. BitDew: an Open Source Middleware for Large Scale Data Management. http://www.bitdew.net.

[W130] P. Malecot, D. Kondo, and G. Fedak. XtremLab: Characterizing Internet Volunteer Computing System. http://xtremlab.lri.fr.

[W131] D. Kondo, G. Fedak, P. Malecot, F. Cappello, H. Casanova, and A. Chien. Desktop Grid Traces Archive. http://dgtrace.lri.fr.

[W132] P. Malecot, A. Rezmerita, G. Fedak, T. Herault, L. Lefevre, and O. Richard. DSL-Lab: an Experimental Platform About Distributed Systems Running on DSL Internet. http://www.dsllab.org.

[W133] O. Lodygensky, G. Fedak, and F. Cappello. XtremWeb: an Open Source Middleware for Desktop Grid Computing. http://www.xtremweb.net.

Bibliography

[1] Ian Foster and Carl Kesselman, editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, Inc., San Francisco, USA, 1999.

[2] Christophe Cérin and Gilles Fedak, editors. *Desktop Grid Computing*. Chapman & All/CRC Press, May 2012.

[3] Franck Cappello, Samir Djilali, Gilles Fedak, Thomas Herault, Frédéric Magniette, Vincent Néri, and Oleg Lodygensky. Computing on Large Scale Distributed Systems: XtremWeb Architecture, Programming Models, Security, Tests and Convergence with Grid. *Future Generation Computer Systems*, 21(3):417–437, mar 2005.

[4] Derrick Kondo, Bahman Javadi, Alexandru Iosup, and Dick Epema. The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems. In *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, pages 398–407. IEEE, 2010.

[5] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51:107–113, January 2008.

[6] Raphaël Bolze, Franck Cappello, Eddy Caron, Michel Daydé, Frédéric Desprez, Emmanuel Jeannot, Yvon Jégou, Stephane Lanteri, Julien Leduc, Noredine Melab, et al. Grid'5000: a large scale and highly reconfigurable experimental grid testbed. *International Journal of High Performance Computing Applications*, 20(4):481–494, 2006.

[7] Franck Cappello, Gilles Fedak, Derrick Kondo, Paul Malécot, and Ala Rezmerita. Chapter 3: Desktop Grids: From Volunteer Distributed Computing to High Throughput Computing Production Platforms. In Kuan-Ching Li, Ching-Hsien Hsu, Laurence Tianruo Yang, Jack Dongarra, and Hans Zima, editors, *Handbook of Research on Scalable Computing Technologies*, pages 31–61. IGI Global, July 2009.

[8] Franck Cappello, Abderrahmane Djilali, Gilles Fedak, Cécile Germain, Oleg Lodygensky, and Vincent Néri. Xtremweb: une plate-forme de recherche sur le calcul global et pair à pair. In Françoise Baud, editor, *Calcul réparti à grande échelle Metacomputing*. Hermes Science, Lavoisier, 2002.

[9] Franck Cappello, Gilles Fedak, Tangui Morlier, and Oleg Lodygensky. Des systèmes client-serveur aux systèmes pair à pair. In Isabelle Comyn and

Wattiau Jacky-Akoka, editors, *Encyclopédie de l'informatique et des systèmes d'information*, pages 195–210. Vuibert, 2007.

[10] John F. Shoch and Jon A. Hupp. The "worm" programs - early experience with a distributed computation. *Communications of the ACM*, 3(25), 03 1982.

[11] Matt W. Mutka and Miron Livny. Profiling workstations' available capacity for remote execution. In *Proceedings of Performance-87, The 12th IFIP W.G. 7.3 International Symposium on Computer Performance Modeling, Measurement and Evaluation*, Brussels, Belgium, 1987.

[12] M. Litzkow, M. Livny, and M. Mutka. Condor - a hunter of idle workstations. In *Proceedings of the 8th International Conference of Distributed Computing Systems (ICDCS)*, pages 104–111, 1988.

[13] D. Ghormley, D. Petrou, S. Rodrigues, A. Vahdat, and T. Anderson. GLUnix: a Global Layer Unix for a Network of Workstations. *Software-Practice and Experience*, 28(9), July 1998.

[14] A. Barak, S. Guday, and Wheeler R. *The MOSIX Distributed Operating System, Load Balancing for UNIX*, volume 672 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[15] Sandeep N. Bhatt, Fan R. K. Chung, Frank Thomson Leighton, and Arnold L. Rosenberg. An optimal strategies for cycle-stealing in networks of workstations. *IEEE Trans. Computers*, 46(5):545–557, 1997.

[16] P. Cappello, B. Christiansen, M. Ionescu, M. Neary, K. Schauser, and D. Wu. Javelin: Internet-Based Parallel Computing Using Java. In *Proceedings of the Sixth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 1997.

[17] Timothy Mattson, Beverly Sanders, and Berna Massingill. *Patterns for Parallel Programming*. Addison-Wesley, 2004.

[18] L. Sarmenta and S. Hirano. Bayanihan: Building and Studying Web-Based Volunteer Computing Systems Using Java. *Future Generation Computer Systems*, 15(5-6):675–686, 1999.

[19] The Great Internet Mersene Prime Search (GIMPS). `http://www.mersenne.org/`.

[20] RSA Labs' 64bit RC5 Encryption Challenge. `http://www.distributed.net`.

[21] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. Seti@home: An experiment in public-resource computing. *Communications of the ACM*, 45(11):56–61, November 2002.

[22] Domenico Talia and Paolo Trunfio. Toward a synergy between p2p and grids. *Internet Computing, IEEE*, 7(4):96–95, 2003.

[23] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, San Diego, California, August 2001.

[24] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, 2001.

[25] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Technical Report UCB/CSD-01-1141, UC Berkeley, April 2001.

[26] B. Cohen. Incentives Build Robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer Systems*, Berkeley, 2003.

[27] D. Anderson. Boinc: A system for public-resource computing and storage. In *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, Pittsburgh, USA, 2004.

[28] Gilles Fedak, Cécile Germain, Vincent Néri, and Franck Cappello. XtremWeb: A Generic Global Computing Platform. In *Proceedings of 1st IEEE International Symposium on Cluster Computing and the Grid CCGRID'2001, Special Session Global Computing on Personal Devices*, pages 582–587, Brisbane, Australia, May 2001. IEEE/ACM, IEEE Press.

[29] J. Pedroso, L.M. Silva, and J.G. Silva. Web-based metacomputing with JET. In *Proc. of the ACM PPoPP Workshop on Java for Science and Engineering Computation*, June 1997.

[30] A. Baratloo, M. Karaul, Z. Kedem, and P. Wyckoff. Charlotte: Metacomputing on the Web. In *Proc. of the 9th International Conference on Parallel and Distributed Computing Systems (PDCS-96)*, 1996.

[31] A. D. Alexandrov, M. Ibel, K. E. Schauser, and C.J. Scheiman. SuperWeb: Towards a Global Web-Based Parallel Computing Infrastructure. In *Proc. of the 11th IEEE International Parallel Processing Symposium (IPPS)*, April 1997.

[32] Tim Brecht, Harjinder Sandhu, Meijuan Shan, and Jimmy Talbot. Paraweb: towards world-wide supercomputing. In *EW 7: Proceedings of the 7th workshop on ACM SIGOPS European workshop*, pages 181–188, New York, NY, USA, 1996. ACM.

[33] N. Camiel, S. London, N. Nisan, and O. Regev. The PopCorn Project: Distributed Computation over the Internet in Java. In *Proc. of the 6th International World Wide Web Conference*, April 1997.

[34] A. Chien, B. Calder, S. Elbert, and K. Bhatia. Entropia: Architecture and Performance of an Enterprise Desktop Grid System. *Journal of Parallel and Distributed Computing*, 63:597–610, 2003.

[35] Cécile Germain, Vincent Néri, Gilles Fedak, and Franck Cappello. XtremWeb: Building an Experimental Platform for Global Computing. In *Grid'2000 First Internationnal Workshop on Grid Computing*, volume 1971, Bangalore, India, December 2000. IEEE/ACM, Springler-Verlags LNCS.

[36] Vincent Neri Cécile Germain, Gilles Fedak and Franck Cappello. Global Computing Systems. In *Proceedings of the Third International Conference on Large-Scale Scientific Computing (LSCC'01)*, volume 2179 of *Lecture Notes in Computer Science*, pages 218–227, London, UK, 2001. Springer-Verlag.

[37] Gilles Fedak. *XtremWeb: une plate-forme pour l'étude expérimentale du calcul global pair-à-pair*. PhD thesis, Université Paris XI, Orsay 2003.

[38] Oleg Lodygensky, Etienne Urbah, and Simon Dadoun. XtremWeb-HEP: Designing Desktop Grid for the EGEE Infrastructure. In Christophe Cérin and Gilles Fedak, editors, *Desktop Grid Computing*, pages 79–97. Chapman & All/CRC, 2012.

[39] Zoltán Balaton, Gábor Gombás, Péter Kacsuk, Adam Kornafeld, József Kovács, Attila Csaba Marosi, Gábor Vida, Norbert Podhorszki, and Tamás Kiss. Sztaki desktop grid: a modular and scalable way of building large computing grids. In *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 1–8. IEEE, 2007.

[40] Nazareno Andrade, Walfredo Cirne, Francisco Brasileiro, and Paulo Roisenberg. OurGrid: An Approach to Easily Assemble Grids with Equitable Resource Sharing. In *Proceedings of the 9th Workshop on Job Scheduling Strategies for Parallel Processing*, June 2003.

[41] J. Pruyne and M. Livny. A Worldwide Flock of Condors : Load Sharing among Workstation Clusters . *Journal on Future Generations of Computer Systems*, 12, 1996.

[42] Rajesh Raman, Miron Livny, and Marvin H. Solomon. Matchmaking: Distributed resource management for high throughput computing. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC)*, pages 140–, 1998.

[43] Adriana Iamnitchi, Ian T. Foster, and Daniel Nurmi. A peer-to-peer approach to resource location in grid environments. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC)*, page 419, 2002.

[44] Dayi Zhou and Virginia Mary Lo. Wavegrid: a scalable fast-turnaround heterogeneous peer-based desktop grid system. In *IEEE International Parallel & Distributed Processing Symposium IPDPS*, 2006.

[45] H. Abbes, C. Cérin, and M. Jemni. Pastrygrid: decentralisation of the execution of distributed applications in desktop grid. In *MGC '08: Proceedings of the 6th international workshop on Middleware for grid computing*, pages 1–6, New York, NY, USA, 2008. ACM.

[46] Jik-Soo Kim, Beomseok Nam, Peter J. Keleher, Michael A. Marsh, Bobby Bhattacharjee, and Alan Sussman. Resource discovery techniques in distributed desktop grid environments. In *Workshop on GRID Computing*, pages 9–16, 2006.

[47] Seyong Lee, Xiaojuan Ren, and Rudolf Eigenmann. Efficient content search in ishare, a p2p based internet-sharing system. In *Workshop on Volunteer and Desktop Grid Computing (PCGrid)*, 2008.

[48] Eddy Caron, Frédéric Desprez, Franck Petit, and Cédric Tedeschi. Dlpt: A p2p tool for service discovery in grid computing. In Nick Antonopoulos, Georgios Exarchakos, Maozhen Li, and Antonio Liotta, editors, *Handbook of Research on P2P and Grid Systems for Service-Oriented Computing: Models, Methodologies and Applications*. IGI Global, December 2009. Released: December 2009. ISBN-13: 978-1615206865.

[49] H. Abbes, C. Cerin, and M. Jemni. BonjourGrid: Orchestration of multi-instances of grid middlewares on institutional Desktop Grids. In *IEEE International Parallel & Distributed Processing Symposium IPDPS*, 2009.

[50] Jason D. Sonnek, Mukesh Nathan, Abhishek Chandra, and Jon B. Weissman. Reputation-based scheduling on unreliable distributed infrastructures. In *Proceedings of the International Conference of Distributed Computing Systems ICDCS*, page 30, 2006.

[51] Eric Heien, N. Fujimoto, and Kenichi Hagihara. Computing low latency batches with unreliable workers in volunteer computing environments. In *PCGrid*, 2008.

[52] D. Kondo, A. Chien, and Casanova H. Rapid Application Turnaround on Enterprise Desktop Grids. In *ACM Conference on High Performance Computing and Networking, SC2004*, November 2004.

[53] Derrick Kondo, Bruno Kindarji, Gilles Fedak, and Franck Cappello. Towards Soft Real-Time Applications on Enterprise Desktop Grids. In *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID '06)*, pages 65–72, Singapore, May 2006.

[54] Louis-Claude Canon, Emmanuel Jeannot, and Jon Weissman. A dynamic approach for characterizing collusion in desktop grids. In *IEEE International Parallel and Distributed Processing Symposium*, pages 1–12. IEEE Press, 2010.

[55] Simon Delamare, Gilles Fedak, Derrick Kondo, and Oleg Lodygensky. SpeQuloS: A QoS Service for BoT Applications Using Best Effort Distributed Computing Infrastructures. In *Proceedings of the 21st ACM International Symposium on High Performance Distributed Computing (HPDC'12)*, pages 173–186, Delft, The Netherlands, June 2012.

[56] F. Berman, R. Wolski, S. Figueira, J. Schopf, and G. Shao. Application-Level Scheduling on Distributed Heterogeneous Networks. In *Proc. of Supercomputing'96, Pittsburgh*, 1996.

[57] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman. Heuristics for Scheduling Parameter Sweep Applications in Grid Environments. In *Proceedings of the 9th Heterogeneous Computing Workshop (HCW'00)*, pages 349–363, May 2000.

[58] H. Casanova, G. Obertelli, F. Berman, and R. Wolski. The AppLeS Parameter Sweep Template: User-Level Middleware for the Grid. In *Proceedings of SuperComputing 2000 (SC'00)*, Nov. 2000.

[59] D. Kondo, A. A. Chien, and H. Casanova. Scheduling task parallel applications for rapid application turnaround on enterprise desktop grids. *Journal of Grid Computing*, 5(4):379–405, 2007.

[60] A. Andrzejak, P. Domingues, and L. Silva. Predicting Machine Availabilities in Desktop Pools. In *IEEE/IFIP Network Operations and Management Symposium*, pages 225–234, 2006.

[61] Artur Andrzejak, Derrick Kondo, and David P. Anderson. Ensuring collective availability in volatile resource pools via forecasting. In *19th IFIP/IEEE Distributed Systems: Operations and Management (DSOM 2008)*, Samos Island, Greece, 2008.

[62] Filipe Araujo, Patricio Domingues, Derrick Kondo, , and Luis Moura Silva. Using cliques of nodes to store desktop grid checkpoints. In *Coregrid Integration Workshop*, Crete, Greece, April 2008.

[63] Patricio Domingues, Filipe Araujo, and Luis Moura Silva. A dht-based infrastructure for sharing checkpoints in desktop grid computing. In *Conference on e-Science and Grid Computing (eScience '06)*, Amsterdam, The Netherlands, December 2006.

[64] Samer Al-Kiswany, Matei Ripeanu, Sudharshan Vazhkudai, and Abdullah Gharaibeh. stdchk: A checkpoint storage system for desktop grid computing. In *International Conference on Distributed Computing Systems (ICDCS'08)*, Beijing, China, 2008.

[65] Fatiha Bouabache, Thomas Herault, Gilles Fedak, and Franck Cappello. Hierarchical Replication Techniques to Ensure Checkpoint Storage Reliability in Grid

Environments. In *Proceedings of 8th IEEE International Symposium on Cluster Computing and the Grid CCGRID'08*, pages 475–483, Lyon, France, may 2008.

[66] Derrick Kondo, Filipe Araujo, Patricio Domingues, and Luis Moura Silva. Result error detection on heterogeneous and volatile resources via intermediate checkpointing. In *Coregrid Integration Workshop*, Rhodes Island, 2007.

[67] G. Ghare and L. Leutenegger. Improving Speedup and Response Times by Replicating Parallel Programs on a SNOW. In *Proceedings of the 10th Workshop on Job Scheduling Strategies for Parallel Processing*, June 2004.

[68] S. Leutenegger and X. Sun. Distributed Computing Feasibility in a Non-Dedicated Homogeneous Distributed System. In *Proc. of SC'93, Portland, Oregon*, 1993.

[69] M. Mutka and M. Livny. The Available Capacity of a Privately Owned Workstation Environment . *Performance Evaluation*, 4(12), July 1991.

[70] Petar Maymounkov and David Mazières. Kademlia: A Peer-to-peer Information System Based on the XOR Metric. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*. MIT, 2002.

[71] C. Gkantsidis and P. Rodriguez. Network Coding for Large Scale Content Distribution. In *Proceedings of IEEE/INFOCOM 2005*, Miami, USA, March 2005.

[72] Y. Fernandess and D. Malkhi. On Collaborative Content Distribution using Multi-Message Gossip. In *Proceeding of IEEE International Parallel & Distributed Processing Symposium (IPDPS)*, Rhodes Island, 2006.

[73] Atul Adya and all. Farsite: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment. *SIGOPS Oper. Syst. Rev.*, 36(SI):1–14, 2002.

[74] Ali Raza Butt, Troy A. Johnson, Yili Zheng, and Y. Charlie Hu. Kosha: A Peer-to-Peer Enhancement for the Network File System. In *Proceeding of International Symposium on SuperComputing SC'04*, 2004.

[75] S. Vazhkudai, X. Ma, V. Freeh, J. Strickland, N. Tammineedi, and S. L. Scott. FreeLoader: Scavenging Desktop Storage Resources for Scientific Data. In *Proceedings of Supercomputing 2005 (SC'05)*, Seattle, 2005.

[76] Alessandro Bassi, Micah Beck, Graham Fagg, Terry Moore, James S. Plank, Martin Swany, and Rich Wolski. The Internet BackPlane Protocol: A Study in Resource Sharing. In *Second IEEE/ACM International Symposium on Cluster Computing and the Grid*, Berlin, Germany, 2002.

[77] John Kubiatowicz and all. OceanStore: An Architecture for Global-scale Persistent Storage. In *Proceedings of ACM ASPLOS*. ACM, November 2000.

[78] Baohua Wei, Gilles Fedak, and Franck Cappello. Collaborative Data Distribution with BitTorrent for Computational Desktop Grids. In *Proceedings of the The 4th IEEE International Symposium on Parallel and Distributed Computing (ISPDC'05)*, pages 250–257, Lille, France, June 2005.

[79] Fernando Costa, Luis Silva, Gilles Fedak, and Ian Kelley. Optimizing the Data Distribution Layer of BOINC with BitTorrent. In *Proceedings of IPDPS'08, 2nd Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2008)*, pages 1–8, Miami, Florida, apr 2008.

[80] Cyril Briquet, Xavier Dalem, Sébastien Jodogne, and Pierre-Arnoul de Marneffe. Scheduling data-intensive bags of tasks in p2p grids with bittorrent-enabled data distribution. In *Proceedings of the second workshop on Use of P2P, GRID and agents for the development of content networks*, UPGRADE '07, pages 39–48, New York, NY, USA, 2007. ACM.

[81] Fernando Costa, Luis Silva, Ian Kelley, and Ian Taylor. Peer-to-peer techniques for data distribution in desktop grid computing platforms. In *Making Grids Work*, pages 377–391. Springer US, 2008.

[82] Jiangyan Xu and Renato Figueiredo. Gatorshare: a file system framework for high-throughput data management. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, HPDC '10, pages 776–786, New York, NY, USA, 2010. ACM.

[83] Ali Kaplan, Geoffrey C. Fox, and Gregor Von Laszewski. Gridtorrent framework: A high-performance data transfer and data sharing framework for scientific computing.

[84] C. Mastroianni, P. Cozza, D. Talia, I. Kelley, and I. Taylor. A scalable super-peer approach for public scientific computation. *Future Generation Computer Systems*, 25(3):213 – 223, 2009.

[85] Ian Kelley and Ian Taylor. Bridging the data management gap between service and desktop grids. In Springer, editor, *Distributed and Parallel Systems*, Hungary, 2008.

[86] Atticfs. `http://www.atticfs.org`.

[87] Adriana Iamnitchi, Shyamala Doraimani, and Gabriele Garzoglio. Filecules in High-Energy Physics: Characteristics and Impact on Resource Management. In *proceeding of 15th IEEE International Symposium on High Performance Distributed Computing HPDC 15*, Paris, 2006.

[88] Ekow Otoo, Doron Rotem, and Alexandru Romosan. Optimal File-Bundle Caching Algorithms for Data-Grids. In *SC '04: Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, page 6, Washington, DC, USA, 2004. IEEE Computer Society.

[89] Elizeu Santos-Neto, Walfredo Cirne, Francisco Brasileiro, and Aliandro Lima. Exploiting Replication and Data Reuse to Efficiently Schedule Data-intensive Applications on Grids. In *Proceedings of the 10th Workshop on Job Scheduling Strategies for Parallel Processing*, 2004.

[90] Luis F. G. Sarmenta. Sabotage-Tolerance Mechanisms for Volunteer Computing Systems. *Future Generation Computer Systems*, 18(4):561–572, 2002.

[91] Cécile Germain-Renaud and Nathalie Playez. Result checking in global computing systems. In *Proceedings of the 17th annual international conference on Supercomputing*, ICS '03, pages 226–233, New York, NY, USA, 2003. ACM.

[92] SungJin Choi and Rajkumar Buyya. Group-based adaptive result certification mechanism in desktop grids. *Future Gener. Comput. Syst.*, 26(5):776–786, May 2010.

[93] Patricio Domingues, Bruno Sousa, and Luis Moura Silva. Sabotage-tolerance and trust management in desktop grid computing. *Future Generation Computer Systems*, 23(7):904 – 912, 2007.

[94] Axel W. Krings, Jean-Louis Roch, and Samir Jafar. Certification of large distributed computations with task dependencies in hostile environments. In *IEEE Electro Information Technology Conference*. IEEE Press, May 2005.

[95] Li Gao and Grzegorz Malewicz. Internet computing of tasks with dependencies using unreliable workers. *Lecture Notes in Computer Science*, 3544:443–458, 2005.

[96] WL Du, J Jia, M Mangal, and M Murugesan. Uncheatable grid computing. In *24Th International Conference on Distributed Computing Systems, Proceedings*, pages 4–11. IEEE Press, 2004.

[97] P Golle and I Mironov. Uncheatable distributed computations. In *Topics in Cryptology - CT-RAS 2001, Proceedings*, volume 2020 of *Lecture Notes in Computer Science*, pages 425–440. Springer Verlag, 2001.

[98] Kan Watanabe, Masaru Fukushi, and Susumu Horiguchi. Optimal Spot-checking for Computation Time Minimization in Volunteer Computing. *Journal of Grid Computing*, 7(4, Sp. Iss. SI):575–600, 2009.

[99] Gheorghe Cosmin Silaghi, Filipe Araujo, Luis Moura Silva, Patricio Domingues, and Alvaro E. Arenas. Defeating colluding nodes in desktop grid computing platforms. *Journal of Grid Computing*, 7(4):555–573, December 2009.

[100] Eugen Staab and Thomas Engel. Collusion Detection for Grid Computing. In *CCGRID: 2009, 9th IEEE International Symposium on Cluster Computing and the Grid*, pages 412–419. IEEE Press, 2009.

[101] Gonzalez D.L. Gil G.G. de Vega F.F. Segal B. Centralized boinc resources manager for institutional networks. *IEEE International Parallel & Distributed Processing Symposium IPDPS*, pages 1–8, 2008.

[102] Werner Herr, DI Kaltchev, F Schmidt, and E McIntosh. Large scale beam-beam simulations for the cern lhc using distributed computing. Technical report, CERN, 2006.

[103] Boinc and atlas. https://twiki.cern.ch/twiki/bin/view/lhcathome/boincandatlas.

[104] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6):599–616, 2009.

[105] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *International journal of high performance computing applications*, 15(3):200–222, 2001.

[106] Thomas A DeFanti, Ian Foster, Michael E Papka, Rick Stevens, and Tim Kuhfuss. Overview of the i-way: Wide-area visual supercomputing. *International Journal of High Performance Computing Applications*, 10(2-3):123–131, 1996.

[107] Fabrizio Gagliardi, Bob Jones, Mario Reale, and Stephen Burke. European DataGrid Project: Experiences of deploying a large scale Testbed for e-Science applications. In *Performance 2002 Tutorial Lectures Book "Performance Evaluations of Complex Systems: Techniques and Tools"*, 2002.

[108] Ewa Deelman, Carl Kesselman, Gaurang Mehta, Leila Meshkat, Laura Pearlman, Kent Blackburn, Phil Ehrens, Albert Lazzarini, Roy Williams, and Scott Koranda. Griphyn and ligo, building a virtual data grid for gravitational wave scientists. In *High Performance Distributed Computing, 2002. HPDC-11 2002. Proceedings. 11th IEEE International Symposium on*, pages 225–234. IEEE, 2002.

[109] Jamie Shiers. The worldwide lhc computing grid (worldwide lcg). *Computer physics communications*, 177(1):219–223, 2007.

[110] Hai Jin. ChinaGrid: Making Grid Computing a Reality. In *Lecture Notes in Computer Science, Volume 3334*, pages 13–24, Springer-Veralag Berlin Heidelberg, 2004.

[111] Mattias Ellert, Aleksandr Konstantinov, Balázs Kónya, Oxana Smirnova, and Anders Wäänänen. The nordugrid project: Using globus toolkit for building grid infrastructure. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 502(2):407–410, 2003.

[112] Charlie Catlett, William E Allcock, Phil Andrews, Ruth A Aydt, Ray Bair, Natasha Balac, Bryan Banister, Trish Barker, Mark Bartelt, Peter H Beckman, et al. Teragrid: Analysis of organization, system architecture, and middleware enabling new types of applications. In *High Performance Computing Workshop*, pages 225–249, 2006.

[113] Satoshi Matsuoka, S Shinjo, Mutsumi Aoyagi, Satoshi Sekiguchi, Hitohide Usami, and Kenichi Miura. Japanese computational grid research project: Naregi. *Proceedings of the IEEE*, 93(3):522–533, 2005.

[114] Wolfgang Gentzsch, Denis Girou, Alison Kennedy, Hermann Lederer, Johannes Reetz, Morris Riedel, Andreas Schott, Andrea Vanni, Mariano Vazquez, and Jules Wolfrat. Deisa—distributed european infrastructure for supercomputing applications. *Journal of Grid Computing*, 9(2):259–277, 2011.

[115] Gregor Von Laszewski, Geoffrey C Fox, Fugang Wang, Andrew J Younge, Archit Kulshrestha, Gregory G Pike, Warren Smith, Jens Voeckler, Renato J Figueiredo, Jose Fortes, et al. Design of the futuregrid experiment management framework. In *Gateway computing environments workshop (GCE)*, pages 1–10, 2010.

[116] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *International Journal of High Performance Computing Applications*, 11(2):115–128, 1997.

[117] M. Ellert, M. Grønager, A. Konstantinov, B. Kónya, J. Lindemann, I. Livenson, J.L. Nielsen, M. Niinimäki, O. Smirnova, and A. Wäänänen. Advanced resource connector middleware for lightweight computational grids. *Future Generation computer systems*, 23(2):219–240, 2007.

[118] E. Laure, S. Fisher, A. Frohner, C. Grandi, P. Kunszt, A. Krenek, O. Mulmo, F. Pacini, F. Prelz, J. White, et al. Programming the grid with glite. *Computational Methods in Science and Technology*, 12(1):33–45, 2006.

[119] Jason Novotny, Steven Tuecke, and Von Welch. An online credential repository for the grid: Myproxy. In *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, pages 104–111. IEEE, 2001.

[120] Roberto Alfieri, Roberto Cecchini, Vincenzo Ciaschini, Luca dell'Agnello, Akos Frohner, Alberto Gianoli, Karoly Lorentey, and Fabio Spataro. Voms, an authorization system for virtual organizations. In *Grid computing*, pages 33–40. Springer, 2004.

[121] W. Allcock, J. Bresnahan, R. Kettimuthu, M. Link, C. Dumitrescu, I. Raicu, and I. Foster. The Globus Striped GridFTP Framework and Server. In *Proceedings of Super Computing (SC05)*, Seattle, USA, 2005.

[122] C Aiftimiei, P Andreetto, S Bertocco, SD Fina, SD Ronco, A Dorigo, A Gianelle, M Marzolla, M Mazzucato, M Sgaravatto, et al. Job submission and management through web services: the experience with the cream service. *Journal of Physics: Conference Series*, 119(6):062004, 2008.

[123] Dietmar W Erwin and David F Snelling. Unicore: A grid computing environment. In *Euro-Par 2001 Parallel Processing*, pages 825–834. Springer, 2001.

[124] Christine Morin. Xtreemos: A grid operating system making your computer ready for participating in virtual organizations. In *Object and Component-Oriented Real-Time Distributed Computing*, 2007.

[125] Eddy Caron and Frédéric Desprez. Diet: A scalable toolbox to build network enabled servers on the grid. *International Journal of High Performance Computing Applications*, 20(3):335–352, 2006.

[126] Steven Tuecke. Grid security infrastructure (gsi) roadmap. In *Grid Forum Security Working Group Draft*, 2001.

[127] Ali Anjomshoaa, Fred Brisard, Michel Drescher, Donal Fellows, An Ly, Stephen McGough, Darren Pulsipher, and Andreas Savva. Job submission description language (jsdl) specification, version 1.0. In *Open Grid Forum, GFD*, volume 56, 2005.

[128] I. Foster, A. Grimshaw, P. Lane, W. Lee, M. Morgan, S. Newhouse, S. Pickles, D. Pulsipher, C. Smith, and M. Theimer. Ogsa basic execution service version 1.0, 2007.

[129] Peter Tröger, Hrabri Rajic, Andreas Haas, and Piotr Domagalski. Standardization of an api for distributed resource management systems. In *In Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007)*, pages 619–627, Rio de Janeiro, Brazil, May 2007.

[130] G. Von Laszewski, I. Foster, and J. Gawor. Cog kits: a bridge between commodity distributed computing and high-performance grids. In *Proceedings of the ACM 2000 conference on Java Grande*, pages 97–106. ACM, 2000.

[131] T. Goodale, S. Jha, H. Kaiser, T. Kielmann, P. Kleijer, G. Von Laszewski, C. Lee, A. Merzky, H. Rajic, and J. Shalf. Saga: A simple api for grid applications. high-level application programming on the grid. *Computational Methods in Science and Technology*, 12(1):7–20, 2006.

[132] Peter Kacsuk. P-grade portal family for grid infrastructures. *Concurrency and Computation: Practice and Experience*, 23(3):235–245, 2011.

[133] Mary Thomas, Stephen Mock, Maytal Dahan, Kurt Mueller, Don Sutton, and John R Boisseau. The gridport toolkit: a system for building grid portals. In *High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on*, pages 216–227. IEEE, 2001.

[134] Dan Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. The eucalyptus open-source cloud-computing system. In *Cloud Computing and Its Applications workshop (CCA'08)*, Chicago, IL, 2008.

[135] Openstack open source cloud computing software. `http://www.openstack.org/`.

[136] Borja Sotomayor, Rubén Santiago Montero, Ignacio Martın Llorente, and Ian Foster. Capacity leasing in cloud systems using the opennebula engine. In *Workshop on Cloud Computing and its Applications*, volume 3, 2008.

[137] Derrick Kondo, Bahman Javadi, Paul Malecot, Franck Cappello, and David P Anderson. Cost-benefit analysis of cloud computing versus desktop grids. In *IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*, pages 1–12. IEEE, 2009.

[138] Mayur R Palankar, Adriana Iamnitchi, Matei Ripeanu, and Simson Garfinkel. Amazon s3 for science grids: a viable solution? In *Proceedings of the 2008 international workshop on Data-aware distributed computing*, pages 55–64. ACM, 2008.

[139] Kate Keahey, Renato Figueiredo, Jos Fortes, Tim Freeman, and Maurcio Tsugawa. Science clouds: Early experiences in cloud computing for scientific applications. *Cloud computing and applications*, 2008:825–830, 2008.

[140] Marc-Elian Bégin and Konstantin Skaburskas. StratusLab Toolkit 2.0. Technical report, StratusLab Consortium, May 2012.

[141] Paul Marshall, Kate Keahey, and Tim Freeman. Elastic site: Using clouds to elastically extend site resources. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 43–52. IEEE Computer Society, 2010.

[142] P. Ruth, P. Mcgachey, , and Dongyan Xu. Viocluster : Virtualization for dynamic computational domains. In *IEEE International Conference on Cluster Computing*, Tsukuba, Japan, 2005.

[143] Arijit Ganguly, Abhishek Agrawal, P. Oscar Boykin, and Renato J. O. Figueiredo. Wow : Self-organizing wide area overlay networks of virtual workstations. In *In HPDC'15 : Proceedings of the 15th IEEE International Sympo- sium on High Performance Distributed Computing*, Paris, France, 2006.

[144] Ala Rezmerita, Tangui Morlier, Vincent Neri, and Franck Cappello. Private virtual cluster : infrastructure and protocol for instant grids. In *Proc. of the Int. Euro-Par Conf. on Parallel Processing (Euro-Par 2006)*, Dresden, Germany, 2006.

[145] Tamas Kiss and Gabor Terstyanszky. Programming applications for desktop grids. In Christophe Cerin and Gilles Fedak, editors, *Desktop Grid Computing*. CRC Press, 2012.

[146] Attila Csaba Marosi, Peter Kacsuk, Gilles Fedak, and Oleg Lodygensky. Sandboxing for Desktop Grids Using Virtualization. In *Proceedings of the 18th Euromicro International Conference on Parallel, Distributed and Network-Based Computing PDP 2010*, pages 559–566, Pisa, Italy, February 2010.

[147] Gilles Fedak. Recent advances and research challenges in desktop grid and volunteer computing. In *Proceedings of the EuroPAR 2009 Workshops, CoreGrid ERCIM Working Group Workshop on Grids, P2P and Service Computing*, pages 171–185, Delft, Netherlands, Aug 2009. LNCS.

[148] Oleg Lodygensky, Etienne Urbah, Simon Dadoun, Anthony Simonet, Gilles Fedak, Simon Delamare, Derrick Kondo, Laurent Duflot, and Xavier Garrido. FlyingGrid : from Volunteer Computing to Volunteer Cloud. In *poster in Computing in Hign Energy and Nuclear Physics (CHEP'12)*, New York, USA, 2012.

[149] Attila Marosi, József Kovács, and Peter Kacsuk. Towards a volunteer cloud system. *Future Generation Computer Systems*, 29(6):1442–1451, 2013.

[150] Zoltan Balaton, Zoltan Farkas, Gabor Gombas, Peter Kacsuk, Robert Lovas, Attila Csaba Marosi, Ad Emmen, Gabor Terstyanszky, Tamas Kiss, Ian Kelley, Ian Taylor, Oleg Lodygensky, Miguel Cardenas-Montes, Gilles Fedak, and Filipe Araujo. EDGeS: the Common Boundary Between Service and Desktop Grids. *Parallel Processing Letters*, 18(3):433–453, September 2008.

[151] Oleg Lodygensky, Gilles Fedak, Franck Cappello, Vincent Neri, Miron Livny, and Douglas Thain. XtremWeb & Condor : Sharing Resources Between Internet Connected Condor Pools. In *Proceedings of CCGRID'2003, Third International Workshop on Global and Peer-to-Peer Computing (GP2PC'03)*, pages 382–389, Tokyo, Japan, 2003. IEEE/ACM.

[152] Mark Silberstein, Artyom Sharov, Dan Geiger, and Assaf Schuster. Gridbot: execution of bags of tasks in multiple grids. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, page 11. ACM, 2009.

[153] Francisco Brasileiro, Alexandre Duarte, Diego Carvalho, Roberto Barber, and Diego Scardaci. An approach for the co-existence of service and opportunistic grids: The eela-2 case. In *Latin-American Grid Workshop*, 2008.

[154] P. Kacsuk, Z. Farkas, and G. Fedak. Towards Making BOINC and EGEE Interoperable. In *Proceedings of 4th IEEE International Conference on e-Science (e-Science 2008), International Grid Interoperability and Interoperation Workshop 2008 (IGIIW 2008)*, pages 478–484, Indianapolis, USA, December 2008.

[155] Haiwu He, Gilles Fedak, Peter Kacsuk, Zoltan Farkas, Zoltan Balaton, Oleg Lodygensky, Etienne Urbah, Gabriel Caillat, and Filipe Araujo. Extending the EGEE Grid with XtremWeb-HEP Desktop Grids. In *Proceedings of CCGRID'10, 4th Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2010)*, pages 685–690, Melbourne, Australia, May 2010.

[156] E. Urbah, P. Kacsuk, Z. Farkas, G. Fedak, G. Kecskemeti, O. Lodygensky, A. Marosi, Z. Balaton, G. Caillat, G. Gombas, A. Kornafeld, J. Kovacs, H. He, and R. Lovas. EDGeS: Bridging EGEE to BOINC and XtremWeb. *Journal of Grid Computing*, 7(3):335–354, September 2009.

[157] Mircea Moca, Cristian Litan, Gheorghe Cosmin Silaghi, and Gilles Fedak. Multi-criteria Task Scheduling Method for Hybrid Distributed Computing Infrastructures. *Future Generation in Computer Systems*, 2015.

[158] Orna Agmon Ben-Yehuda, Assaf Schuster, Artyom Sharov, Mark Silberstein, and Alexandru Iosup. Expert: Pareto-efficient task replication on grids and a cloud. In *IEEE 26th International Parallel & Distributed Processing Symposium (IPDPS)*, pages 167–178. IEEE, 2012.

[159] Paul Malécot, Derrick Kondo, and Gilles Fedak. XtremLab: A System for Characterizing Internet Desktop Grids. In *Poster in The 15th IEEE International Symposium on High Performance Distributed Computing HPDC'06*, Paris, France, June 2006.

[160] Nicolas Capit, Georges Da Costa, Yiannis Georgiou, Guillaume Huard, Cyrille Martin, Grégory Mounié, Pierre Neyron, and Olivier Richard. A batch scheduler with high level components. In *IEEE International Symposium on Cluster Computing and the Grid, CCGrid.*, volume 2, pages 776–783. IEEE, 2005.

[161] Orna Agmon Ben-Yehuda, Muli Ben-Yehuda, Assaf Schuster, and Dan Tsafrir. Deconstructing amazon ec2 spot instance pricing. *ACM Transactions on Economics and Computation*, 1(3):16, 2013.

[162] Paul Marshall, Kate Keahey, and Timothy Freeman. Improving utilization of infrastructure clouds. In *Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on*, pages 205–214. IEEE, 2011.

[163] Fernando Costa, Luis Silva, Gilles Fedak, and Ian Kelley. Optimizing Data Distribution in Desktop Grid Platforms. *Parallel Processing Letters*, 18(3):391–410, September 2008.

[164] Baohua Wei, Gilles Fedak, and Franck Cappello. Towards Efficient Data Distribution on Computational Desktop Grids with BitTorrent. *Future Generation Computer Systems*, 23(7):983–989, November 2007.

[165] Baohua Wei, Gilles Fedak, and Franck Cappello. Scheduling Independent Tasks Sharing Large Data Distributed with BitTorrent. In *6th IEEE/ACM International Workshop on Grid Computing*, pages 219–226, Seattle, USA, November 2005.

[166] James Broberg, Rajkumar Buyya, and Zahir Tari. Metacdn: Harnessing 'storage clouds' for high performance content delivery. *Journal of Network and Computer Applications*, 32(5):1012–1022, 2009.

[167] Heshan Lin, Xiaosong Ma, Jeremy Archuleta, Wu-chun Feng, Mark Gardner, and Zhe Zhang. Moon: Mapreduce on opportunistic environments. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, pages 95–106. ACM, 2010.

[168] Fernando Costa, Luis Silva, and Michael Dahlin. Volunteer cloud computing: Mapreduce over the internet. In *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on*, pages 1855–1862. IEEE, 2011.

[169] Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. P2p-mapreduce: Parallel data processing in dynamic cloud environments. *Journal of Computer and System Sciences*, 78(5):1382–1402, 2012.

[170] Luis Rodero-Merino, Gilles Fedak, and Adrian Muresan. MapReduce and Hadoop. In Luis M. Vaquero, Juanjo Hierro, and Juan Cáceres, editors, *Open Source Cloud Computing Systems: Practices and Paradigms*. IGI Global, 2011.

[171] Gabriel Antoniu, Julien Bigot, Cristophe Blanchet, Luc Bougé, François Briant, Franck Cappello, Alexandru Costan, Frédéric Desprez, Gilles Fedak, Sylvain Gault, Kate Keahey, Bogdan Nicolae, Christian Pérez, Anthony Simonet, Frédéric Suter, Bing Tang, and Raphael Terreux. Scalable Data Management for MapReduce-Based Data-Intensive Applications: a View for Cloud and Hybrid Infrastructures. *International Journal on Cloud Computing*, 2(2-3), January 2013.

[172] Heshan Lin, Wu-Chun Feng, and Gilles Fedak. Data-Intensive Computing on Desktop Grids. In Christophe Cérin and Gilles Fedak, editors, *Desktop Grid Computing*, pages 237–259. Chapman & All/CRC Press, 2012.

[173] Lu Lu, Hai Jin, Xuanhua Shi, and Gilles Fedak. Assessing MapReduce for Internet Computing: a Comparison of Hadoop and BitDew-MapReduce. In *Proceedings of the 13th ACM/IEEE International Conference on Grid Computing (Grid 2012)*, Beijing, China, September 2012.

[174] Mircea Moca, Gheorghe Cosmin Silaghi, and Gilles Fedak. Distributed Results Checking for MapReduce on Volunteer Computing. In *Proceedings of IPDPS'2011, 4th Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid 2011)*, Anchorage, Alaska, May 2011.

[175] Bing Tang, Mircea Moca, Stéphane Chevalier, Haiwu He, and Gilles Fedak. Towards MapReduce for Desktop Grid Computing. In *Fifth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC'10)*, pages 193–200, Fukuoka, Japan, November 2010. IEEE.

[176] Bing Tang, Haiwu He, and Gilles Fedak. HybridMR: A New Approach for Hybrid MapReduce Combining Desktop Grid and Cloud Infrastructures. *Concurrency Practice and Experience*, 2015.

[177] Bing Tang, Haiwu He, and Gilles Fedak. Parallel Data Processing in Dynamic Hybrid Computing Environment Using MapReduce. In *14th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP'14)*, volume 8631 of *Lecture Notes on Computer Science*, pages 1–14, Dalian, China, August 2014. Springer Verlags.

[178] Julios C. S. dos Anjos, Gilles Fedak, and Claudio F. R. Geyer. BIGhybrid - A Toolkit for Simulating MapReduce in Hybrid Infrastructures. In *Workshop on Parallel and Distributed Computing for Big Data Applications (WPBA'14)*, Paris, October 2014.

[179] Asma Ben Cheikh, Heithem Abbes, and Gilles Fedak. Ensuring Privacy for MapReduce on Hybrid Clouds Using Information Dispersal Algorithm. In *7th International Conference on Data Management in Cloud, Grid and P2P Systems (Globe '14)*, volume 8648 of *Lecture Notes on Computer Science*, pages 37–48, Munich, Germany, September 2014. Springer Verlags.

[180] Bing Tang and Gilles Fedak. Analysis of Data Reliability Tradeoffs in Hybrid Distributed Storage Systems. In *Proceedings of Parallel and Distributed Symposium Workshops and PhD Forum (IPDPSW'12), 17th IEEE International Workshop on Dependable Parallel, Distributed and Network-Centric Systems (DPDNS'12)*, Shanghai, China, May 2012.

[181] David Anderson and Gilles Fedak. The Computational and Storage Potential of Volunteer Computing. In *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, pages 73–80, Singapore, May 2006.

[182] Derrick Kondo, Michela Taufer, C Brooks, Henri Casanova, and Andrew Chien. Characterizing and evaluating desktop grids: An empirical study. In *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS)*, page 26. IEEE, 2004.

[183] Derrick Kondo, Gilles Fedak, Franck Cappello, Andrew A. Chien, and Henri Casanova. On Resource Volatility in Enterprise Desktop Grids. In *Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing (eScience'06)*, pages 78–86, Amsterdam, Netherlands, December 2006.

[184] Derrick Kondo, Gilles Fedak, Franck Cappello, Andrew A. Chien, and Henri Casanova. Resource Availability in Enterprise Desktop Grids. *Future Generation Computer Systems*, 23(7):888–903, August 2007.

[185] Derrick Kondo, Felipe Araujo, Paul Malecot, Patricio Domingues, Luis M. Silva, Gilles Fedak, and Franck Cappello. Characterizing Result Errors in Internet Desktop Grids. In *European Conference on Parallel and Distributed Computing EuroPar'07*, Rennes, France, August 2007.

[186] Derrick Kondo, Gilles Fedak, Paul Malecot, Franck Cappello, Henri Casanova, and Andrew Chien. Desktop Grid Traces Archive.

[187] Arnaud Legrand, Loris Marchal, and Henri Casanova. Scheduling distributed applications: the simgrid simulation framework. In *3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid)*, pages 138–145. IEEE, 2003.

[188] Bogdan Nicolae, Gabriel Antoniu, Luc Bougé, Diana Moise, and Alexandra Carpen-Amarie. Blobseer: Next-generation data management for large scale infrastructures. *Journal of Parallel and Distributed Computing*, 71(2):169–184, 2011.

[189] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. Planetlab: an overlay testbed for broad-coverage services. *ACM SIGCOMM Computer Communication Review*, 33(3):3–12, 2003.

[190] Gilles Fedak, Jean-Patrick Gelas, Thomas Hérault, Victor Iniesta, Derrick Kondo, Laurent Lefèvre, Paul Malécot, Lucas Nussbaum, Ala Rezmerita, and Olivier Richard. DSL-Lab: a Platform to Experiment on Domestic Broadband Internet. In *Proceedings of the The 9th IEEE International Symposium on Parallel and Distributed Computing (ISPDC'10)*, pages 141–148, Istanbul, Turquey, July 2010.

[191] Simon Delamare and Gilles Fedak. Towards Hybridized Clouds and Desktop Grid Infrastructures. In Christophe Cérin and Gilles Fedak, editors, *Desktop Grid Computing*, pages 261–285. Chapman & All/CRC Press, 2012.

[192] Gabriel Caillat, Oleg Lodygensky, Gilles Fedak, Haiwu He, Zoltan Balaton, Zoltan Farkas, Gabor Gombas, Peter Kacsuk, Robert Lovas, Attila Csaba Maros, Ian Kelley, Ian Taylor, Gabor Terstyanszky, Tamas Kiss, Miguel Cardenas-Montes, Ad Emmen, and Filipe Araujo. EDGeS: The art of bridging EGEE to BOINC and XtremWeb. In *Proceedings of Computing in High Energy and Nuclear Physics (CHEP'09) (Abstract)*, Prague, Czech Republic, March 2009.

[193] Gilles Fedak, Haiwu He, Oleg Lodygensky, Zoltan Balaton, Zoltan Farkas, Gabor Gombas, Peter Kacsuk, Robert Lovas, Attila Csaba Maros, Ian Kelley, Ian Taylor, Gabor Terstyanszky, Tamas Kiss, Miguel Cardenas-Montes, Ad Emmen, and Filipe Araujo. EDGeS:A Bridge Between Desktop Grids and Service Grids. In *IEEE computing society Proceeding of the 3rd ChinaGrid Annual Conference*, pages 1–9, Dunhuang, Gansu, China, August 2008.

[194] Miguel Cárdenas-Montes, Ad Emmen, Attila Csaba Marosi, Filipe Araujo, Gábor Gombás, Gilles Fedak, Ian Kelley, Ian Taylor, Oleg Lodygensky, Peter Kacsuk, Robert Lovas, Tamas Kiss, Zoltán Balaton, Zoltán Farkas, and Gabor Terstyanszky. EDGeS: Bridging Desktop and Service Grids. In *Proceedings of IBERGRID, 2nd Iberian Grid Infrastructure Conference*, pages 212–226, Porto, Portugal, May 2008.

[195] Zoltan Balaton, Zoltan Farkas, Gabor Gombas, Peter Kacsuk, Robert Lovas, Attila Csaba Marosi, Ad Emmen, Gabor Terstyanszky, Tamas Kiss, Ian Kelley, Ian Taylor, Oleg Lodygensky, Miguel Cardenas-Montes, Gilles Fedak, and Filipe Araujo. EDGeS: the Common Boundary Between Service and Desktop Grids. In *Proceedings of the CoreGrid Integration Workshop (CGIW08)*, Hersonissos-Crete, Greece, April 2008.

[196] Gabriel Caillat, Gilles Fedak, Haiwu He, Oleg Lodygensky, and Etienne Urbah. Towards a Security Model to Bridge Internet Desktop Grids and Service Grids. In *Proceedings of the Euro-Par 2008 Workshops (LNCS), Workshop on Secure, Trusted, Manageable and Controllable Grid Services (SGS'08)*, Las Palmas de Gran Canaria, Spain, August 2008.

[197] Simon Delamare, Gilles Fedak, Derrick Kondo, and Oleg Lodygensky. SpeQuloS : A QoS Service for Hybrid and Elastic Computing Infrastructures. *Journal of Cluster Computing*, 17(1):79–100, 2014.

[198] Oleksandr Gatsenko, Oleksandra Baskova, Oleg Lodygensky, Gilles Fedak, and Yuri Gordienko. Statistical Properties of Deformed Single-Crystal Surface under Real-Time Video Monitoring and Processing in the Desktop Grid Distributed Computing Environment. In *Proceedings of the Sixth International Conference on Materials Struncture and Micromechanics of Fracture (MSMF6)*, Brno, Czech Republic, June 2010.

[199] O. Gatsenko, O. Baskova, G. Fedak, O. Lodygensky, and Yu. Gordienko. Porting Multiparametric MATLAB Application for Image and Video Processing to Desktop Grid for High-Performance Distributed Computing. In *Proceedings of 3rd Grid Experience workshop - Desktop Grid Applications for eScience and eBusiness*, Alemere, Nederlands, March 2010. EnterTheGrid.

[200] O. Gatsenko, O. Baskova, G. Fedak, O. Lodygensky, and Yu. Gordienko. Kinetics of Defect Aggregation in Materials Science Simulated in Desktop Grid Computing Environment Installed in Ordinary Material Science Lab. In *Proceedings of 3rd Grid Experience workshop - Desktop Grid Applications for eScience and eBusiness*, Alemere, Nederlands, March 2010. EnterTheGrid.

[201] Oleg Lodygensky, Gilles Fedak, Vincent Neri, Cécile Germain, Franck Cappello, and Alain Cordier. Augernome & XtremWeb : Monte Carlo Computation on a Global Computing Platform. In *Proceedings of CHEP03 Conference for Computing in High Energy and Nuclear Physics*, San Diego, USA, 2003.

[202] Oleksandr Gatsenko, Oleksandra Baskova, Oleg Lodygensky, Gilles Fedak, and Yuri Gordienko. Statistical Properties of Deformed Single-Crystal Surface under Real-Time Video Monitoring and Processing in the Desktop Grid Distributed Computing Environment. *Journal of Key Engineering Materials*, Materials Structure & Micromechanics of Fracture VI(465):306–309, January 2011.

[203] Henri Casanova, Arnaud Legrand, Dmitrii Zagorodnov, and Francine Berman. Heuristics for scheduling parameter sweep applications in grid environments. In *Heterogeneous Computing Workshop, 2000.(HCW 2000) Proceedings. 9th*, pages 349–363. IEEE, 2000.

[204] Mircea Moca and Gilles Fedak. Using Promethee Methods for Multi-Criteria Pull-based Scheduling on DCIs. In *Proceedings of the 8th IEEE International Conference on eScience (eScience'12)*, Chicago, USA, October 2012.

[205] Mircea Moca, Cristian Litan, Gheorghe Cosmin Silaghi, and Gilles Fedak. Advanced Promethee-based Scheduler Enriched with User-Oriented Methods. In *In Proceedigs of the 10th IEEE Conference on Economics of Grids, Clouds, Systems, and Services (GECON 2013)*, volume 8193 of *Lecture Notes in Computer Science*, pages 161–172, Zaragoza, Spain, September 2013. Springer International Publishing.

[206] Jean-Pierre Brans, Ph Vincke, and Bertrand Mareschal. How to select and how to rank projects: The promethee method. *European journal of operational research*, 24(2):228–238, 1986.

[207] José Figueira, Salvatore Greco, and Matthias Ehrgott. *Multiple criteria decision analysis: state of the art surveys*, volume 78. Springer Science & Business Media, 2005.

[208] Fabrice Bellard. Qemu, a fast and portable dynamic translator. In *USENIX Annual Technical Conference, FREENIX Track*, pages 41–46, 2005.

[209] Tony Hey, Stewart Tansley, and Kristin Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, Redmond, Washington, 2009.

[210] Geoffrey Fox, Tony Hey, and Anne Trefethen. Where does all the data come from? Technical report, Indiana University, November 2011.

[211] Gilles Fedak, Haiwu He, and Franck Cappello. BitDew: A Programmable Environment for Large-Scale Data Management and Distribution. In *Proceedings of the ACM/IEEE SuperComputing Conference (SC'08)*, pages 1–12, Austin, USA, November 2008.

[212] Gilles Fedak, Haiwu He, and Franck Cappello. BitDew: A Data Management and Distribution Service with Multi-Protocol and Reliable File Transfer. *Journal of Network and Computer Applications*, 32(5):961–975, September 2009.

[213] Haiwu He, Gilles Fedak, Bing Tran, and Franck Cappello. BLAST Application with Data-aware Desktop Grid Middleware. In *Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid CCGRID'09*, pages 284–291, Shanghai, China, May 2009.

[214] Gilles Fedak, Haiwu He, and Franck Cappello. A File Transfer Service with Client/Server, P2P and Wide Area Storage Protocols. In *Proceedings of the First International Conference on Data Management in Grid and P2P Systems (Globe'2008)*, LNCS, pages 1–11, Turin, Italy, September 2008. Springer Verlag.

[215] Nicolas Kourtellis, Joshua Finnis, Paul Anderson, Jeremy Blackburn, Cristian Borcea, and Adriana Iamnitchi. Prometheus: User-controlled p2p social data management for socially-aware applications. In *Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware*, Middleware '10, pages 212–231, Berlin, Heidelberg, 2010. Springer-Verlag.

[216] Mohamed Labidi, Bing Tang, Gilles Fedak, Maher Khemakem, and Mohamed Jemni. Scheduling Data and Task on Data-Driven Master/Worker Platform. In *The 13th IEEE International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT-12)*, Beijing, China, December 2012.

[217] Fatiha Bouabache, Thomas Herault, Gilles Fedak, and Franck Cappello. Hierarchical Replication Techniques to Ensure Checkpoint Storage Reliability in Grid Environment. *Journal of Interconnection Networks*, 10(4):345–364, 2009.

[218] Fatiha Bouabache, Thomas Herault, Gilles Fedak, and Franck Cappello. *A Distributed and Replicated Service for Checkpoint Storage*, volume 7 of *CoreGRID Books: Making Grids Work*, chapter Checkpointing and Monitoring, pages 293–306. M. Danelutto, P. Fragopoulou and V. Getov, eds. Springer, 2008.

[219] Fatiha Bouabache, Thomas Herault, Gilles Fedak, and Franck Cappello. A Distributed and Replicated Service for Checkpoint Storage. In *CoreGRID Workshop on Grid programming model, Grid and P2P systems architecture and Grid systems, tools and environments*, Heraklion, Greece, June 2007.

[220] Bing Tang and Gilles Fedak. WukaStore: Hybrid Storage using Clouds and Desktop GRids. Technical Report XX, INRIA, 2012.

[221] L.B. Costa, H. Yang, E. Vairavanathan, A. Barros, K. Maheshwari, G. Fedak, D. Katz, M. Wilde, M. Ripeanu, and S. Al-Kiswany. The Case for Workflow-Aware Storage: An Opportunity Study using MosaStore. *Journal of Grid Computing*, pages 1–19, 2014.

[222] Gabriel Antoniu, Julien Bigot, Cristophe Blanchet, Luc Bougé, François Briant, Franck Cappello, Alexandru Costan, Frédéric Desprez, Gilles Fedak, Sylvain Gault, Kate Keahey, Bogdan Nicolae, Christian Pérez, Anthony Simonet, Frédéric Suter, Bing Tang, and Raphael Terreux. Towards Scalable Data Management for Map-Reduce-based Data-Intensive Applications on Cloud and Hybrid Infrastructures. In *The 1st International IBM Cloud Academy Conference (ICA CON 2012)*, North Carolina, USA, April 2012.

[223] Bogdan Nicolae, Diana Moise, Gabriel Antoniu, Luc Bougé, and Matthieu Dorier. Blobseer: Bringing high throughput under heavy concurrency to hadoop map-reduce applications. In *International Symposium on Parallel & Distributed Processing (IPDPS)*, pages 1–11. IEEE, 2010.

[224] Anthony Simonet, Gilles Fedak, and Matei Ripeanu. Active Data: A Programming Model for Managing Big Data Life Cycle. *Future Generation in Computer Systems*, 2015. to appear.

[225] Anthony Simonet, Gilles Fedak, Matei Ripeanu, and Samer Al-Kiswany. Active Data: A Data-Centric Approach to Data Life-Cycle Management. In *8th Parallel Data Storage Workshop (PDSW), Proceedings of SC13 workshops*, pages 39–44, Denver, USA, November 2013. ACM.

[226] Tadao Murata. Petri nets: Properties, analysis and applications. In *Proceedings of the IEEE*, pages 541–580, April 1989.

[227] A. Simonet, K. Chard, G. Fedak, and I. Foster. Active Data to Provide Smart Data Surveillance to E-Science Users. In *Proceedings of IEEE Euromicro-PDP'15*, Turku, Finland, March 2015.

[228] Adrien Lebre, Jonathan Pastor, Marin Bertier, Frédéric Desprez, Jonathan Rouzaud-Cornabas, Cédric Tedeschi, Anne-Cécile Orgerie, Flavien Quesnel, and Gilles Fedak. Beyond The Clouds, How Should Next Generation Utility Computing Infrastructures Be Designed ? In Zaigham Mahmood, editor, *Cloud Computing: Challenges, Limitations and R&D Solutions*. Springer, November 2014.

[229] Ínigo Goiri, William Katsak, Kien Le, Thu D Nguyen, and Ricardo Bianchini. Parasol and greenswitch: Managing datacenters powered by renewable energy. In *ACM SIGARCH Computer Architecture News*, volume 41, pages 51–64. ACM, 2013.

[230] Jie Liu, Michel Goraczko, Sean James, Christian Belady, Jiakang Lu, and Kamin Whitehouse. The data furnace: heating up with cloud computing. In *Proceedings of the 3rd USENIX conference on Hot topics in cloud computing, HotCloud*, volume 11, pages 15–15, 2011.