Future Generation Computer Systems 27 (2011) 32–39

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

LAG: Achieving transparent access to legacy data by leveraging grid environment

Yuhui Deng [a,*], Frank Wang [b]

[a] Department of Computer Science, Jinan University, Guangzhou, 510632, PR China

[b] Cambridge-Cranfield High Performance Computing Facility, Cranfield University Campus, Bedfordshire MK430AL, United Kingdom

ARTICLE INFO

Article history:
Received 30 September 2007
Received in revised form
2 July 2010
Accepted 10 July 2010
Available online 16 July 2010

Keywords:
Legacy data
Grid
Architecture
Grid service
Web service

ABSTRACT

The world today is experiencing an explosive growth of data generated by information digitization. Due to the unprecedented advance in software and hardware, large amounts of data gradually becomes legacy data and inaccessible. This is building a digital black hole, and it is becoming a big challenge to access, process, and preserve the legacy data. Grid provides flexible, secure, and coordinated resource sharing among dynamic collections of individuals, institutions, and resources. It allows users and applications to access the aggregated resources in a transparent manner. This paper proposes a Legacy Application Grid (LAG) architecture. This architecture deploys diverse legacy applications in a grid environment and provides a transparent access to the remote LAG users who want to access the legacy data. In contrast to the existing methods which attempt to tackle legacy data and legacy applications, we wrap a display protocol into grid services. The service provider, who wants to deploy any legacy applications, just needs to deploy the protocol based grid service, describe and pass the parameters of those legacy applications to the service. Compared with the traditional approaches, the method proposed in this paper is very cost-effective because it avoids converting legacy data from one format to another format or upgrading legacy applications one by one. An implemented prototype validates that the LAG architecture trades acceptable performance degradation for a transparent and remote access to legacy data.

1. Introduction

According to a new report from IDC, 161 exabytes of digital information were created and copied in 2006. The growth will continue to increase exponentially. The amount of information in 2010 will surge more than six fold to 988 exabytes which amounts to a compound annual growth rate of 57% [1]. The explosive data is normally stored in autonomous repositories distributed across the Internet and varies in representation from structured (e.g. relational database) to semi-structured (e.g. e-mail and HTML pages) and unstructured formats (e.g. image and video) [2]. The ubiquitous Internet has provided an easy access to a large number of autonomous and heterogeneous information sources [3]. However, due to the unprecedented development of software and hardware, large amounts of legacy data is becoming a big challenge which we have to face when accessing the digital information. (Legacy data is the data which has been inherited from applications, software, languages, platforms, and techniques earlier than current technology.) The National Archives of the United Kingdom, which holds 900 years of written material, has more than 580 terabytes of data in legacy data formats that are no longer commercially available. Some digital documents held by the national archives had already been lost forever, because the programs which could access them no longer exist [4]. There are two reasons which cause the problem. The first one is the range of proprietary data formats that proliferated during the early digital revolution. The different data formats do no work together, which makes interoperability a big problem. The second one is that the data formats employed by software companies are not only incompatible with that of the rival companies, but also between different generations of the same program (e.g. Microsoft) [4,5].

The growing legacy data has propelled research on how to access, process, and preserve the legacy data. Some research efforts have been invested in tackling the growing challenge. Saving data in one format with one program makes it difficult to open in another program without sacrificing some information. Extensible Markup Language (XML) [6] offers portability and ease of machine processing. The wide spread and growing maturity of XML technologies bring new opportunities to tackle the legacy data. Chidlovskii and Fuselier [7] investigated data conversion from the rendering-oriented HTML markup into a semantic-oriented XML annotation defined by user-specific DTDs or XML Schema descriptions. They applied a supervised learning framework to the conversion task according to which the transformations are learned from a set of training examples. The data which are in proprietary formats such as PDF, MS Word, etc. have to be first converted to a standard format like HTML, and then the layout HTML annotations will be converted to the semantic XML.

[*] Corresponding author.
E-mail addresses: tyhdeng@jnu.edu.cn, yuhuid@hotmail.com (Y. Deng).

0167-739X/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2010.07.004

Kuikka et al. [8] developed a syntax directed approach to transform the XML documents from one structure to another. The aim is to automate a transformation between two grammars that have common parts, although the grammars and names of elements may differ. Other driving forces for the research on the legacy data are from industry. Open XML is a data format developed by Microsoft. This format can be adopted to save files from programs such as Word, Excel and Powerpoint. The open XML is an open international standard under independent control and is free for access [4]. Working with three partners, Microsoft also released a translation program which allows users to save Word documents in the ODF format favoured by the free Open Office application [5]. With support from Microsoft, the National Archive of the United Kingdom will be able to read older data formats in the format they were originally saved by running emulated versions of the older Windows operating systems on modern PCs. For example, if a Word document was saved using Office 97 under Windows 95, then the National Archives will be able to open that document by emulating the operating system and the corresponding software on a modern machine [4].

The legacy data and legacy application is normally in a one to one correspondence, which indicates that a specific legacy data format can only be accessed by the corresponding legacy application. If the legacy applications can be upgraded, the corresponding legacy data will be solved as well. A lot of research efforts have been invested in tackling the legacy applications. Generally, the existing solutions can be classified into three categories [9]. The first one is redevelopment. This method rewrites or reconstructs the existing legacy applications. The common activities include parsing the system and analyzing its syntax to obtain an abstract syntax tree representation of the source code, extracting the interface fragments from the system, and performing control flow analysis [10,11]. The redeployment method requires shutting down the legacy applications either during development or during the replacement [9]. The second one is wrapping. This approach surrounds the legacy component with a new interface. Thiran et al. [12] proposed and designed a generic and technology independent R/W wrapper architecture. The wrapper allows a smooth transition from the legacy and deficient databases to modern architectures, and makes the integration of a legacy database into current large applications easier. The third one is migration. This solution moves legacy applications to a new environment, while retaining the original system's data and functionality. Wrapping and migration are normally employed to reuse legacy applications. One or more approaches could be involved when tackling the legacy applications. Bi et al. [11] investigated a hybrid approach of wrapping and migration for the reuse of legacy applications from its original environment to the Internet-based platform based on a thin client using Java RMI.

However, for those companies and organizations that have a large number of diverse and legacy data, the conversion of all legacy data or the upgrading of all legacy applications could take a lot of time and money, and raise many technical problems. For personal users who have a little legacy data or just want to use the legacy data temporarily, it is not cost-effective to buy a software package to convert the data or upgrade the corresponding legacy application.

Grid is a flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources [13–16]. The objective is to virtualize resources, and allow users and applications to access shared resources in a transparent manner. In recent years, the research community has been very active in the area of investigating techniques in tackling the legacy applications in a grid environment. Kacsuk et al. [17] proposed a new approach to deploy legacy codes as grid services without modifying the original code. A workflow oriented grid portal is designed to apply the legacy code based grid services to complex business processes. Huang et al. [18] designed a wrapping and data mapping technique for converting the existing legacy code (e.g. libraries of scientific and mathematical software written in C language) into composing computational services within a grid environment. Bodhuin and Tortorella [19] designed a tool which can automatically transform the source code of legacy applications and make them compatible with the web or grid technologies. GEMLCA [20] is a front end OGSI grid service layer which surrounds the target host environment and executes legacy applications through the OGSI grid service. Plantikow [21] proposed a data management system architecture and discussed approaches for the integration of legacy applications and grid scheduling with the proposed architecture. An integrator is designed to instruct the VFS driver to add a new logical file system view. Such views are used to provide the legacy applications with input data and to collect the results. Each legacy program is run inside a jail/sandbox such that it only accesses its logical view. LGF [22] is a two-tier architecture in which the interface layer is decoupled from the legacy layer. This enables many benefits that surpass the performance penalty due to the additional interposition layer. The LGF enables semi-automatic virtualization of legacy codes as grid services. McGough et al. [23] proposed the use of a standards based job submission and monitoring system. This approach enables us to deploy legacy applications into the existing resources within a Grid, to map applications into Grid applications, and to use a web based portal to expose these applications to the end users.

In this paper, we propose a Legacy Application Grid (LAG) which is based on an existing grid environment (GT4) [15,24] consisting of a Monitoring and Discovery System (MDS) [25], a certificate authentication centre and several grid service providers. In contrast to the existing methods, we wrap a display protocol into grid service, which is registered in a MDS, instead of directly putting legacy applications into grid service or converting the legacy data from one format to another format. The service provider who wants to deploy any legacy applications just need to deploy the protocol based grid service, describe and pass the parameters (e.g. application name) of those legacy applications to the grid service. Therefore, all kinds of legacy applications can be deployed in the LAG without modifying the source code and the GUI. LAG users who want to access any legacy data can locate and discover the required legacy application in LAG, and then employ the application to access the corresponding legacy data transparently. The LAG can be maintained by companies or organizations. Thus, the method is very cost-effective for both the companies and personal users who want to access legacy data, because they just need to pay for what they use. A system prototype is constructed to investigate the overhead involved in the data access in LAG. The experimental results illustrate that the LAG can provide transparent access to the legacy data with acceptable performance degradation.

The remainder of the paper is organized as follows. Section 2 gives a brief description about the background knowledge of display protocols. Overview of the LAG architecture and workflow are introduced in Section 3. Section 4 describes how to build a legacy service, manage lifecycle and state, and service registration. Section 5 illustrates the prototype system and evaluates the system performance. Section 6 concludes the paper with remarks on the contributions of the paper. There is also a brief discussion in Section 6.

2. Background

Generally, there are two major display protocols which offer graphics sharing and guarantee network transparency: Virtual Networking Computing (VNC) [26] and X Window System (X11).