LNAI 8109

Concha Bielza Antonio Salmerón
Amparo Alonso-Betanzos
J. Ignacio Hidalgo Luis Martínez
Alicia Troncoso Emilio Corchado
Juan M. Corchado (Eds.)

Advances in Artificial Intelligence

15th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2013
Madrid, Spain, September 2013, Proceedings

Springer

Lecture Notes in Artificial Intelligence 8109

Subseries of Lecture Notes in Computer Science

LNAI Series Editors

Randy Goebel
University of Alberta, Edmonton, Canada

Yuzuru Tanaka
Hokkaido University, Sapporo, Japan

Wolfgang Wahlster
DFKI and Saarland University, Saarbrücken, Germany

LNAI Founding Series Editor

Joerg Siekmann
DFKI and Saarland University, Saarbrücken, Germany

Concha Bielza Antonio Salmerón
Amparo Alonso-Betanzos
J. Ignacio Hidalgo Luis Martínez
Alicia Troncoso Emilio Corchado
Juan M. Corchado (Eds.)

Advances in Artificial Intelligence

15th Conference of the Spanish Association
for Artificial Intelligence, CAEPIA 2013
Madrid, Spain, September 17-20, 2013
Proceedings

Springer

Volume Editors

Concha Bielza
Universidad Politécnica de Madrid
28660 Madrid, Spain
E-mail: mcbielza@fi.upm.es

Antonio Salmerón
Universidad de Almería
04120 Almería, Spain
E-mail: antonio.salmeron@ual.es

Amparo Alonso-Betanzos
Universidad de A Coruña
15071 A Coruña, Spain
E-mail: ciamparo@udc.es

J. Ignacio Hidalgo
Universidad Complutense de Madrid
28040 Madrid, Spain
E-mail: hidalgo@dacya.ucm.es

Luis Martínez
Universidad de Jaén
23071 Jaén, Spain
E-mail: luis.martinez@ujaen.es

Alicia Troncoso
Universidad Pablo de Olavide
41013 Sevilla, Spain
E-mail: ali@upo.es

Emilio Corchado
Universidad de Salamanca
37008 Salamanca, Spain,
E-mail: escorchado@usal.es

Juan M. Corchado
Universidad de Salamanca
37008 Salamanca, Spain
E-mail: corchado@usal.es

ISSN 0302-9743
e-ISSN 1611-3349
ISBN 978-3-642-40642-3
e-ISBN 978-3-642-40643-0
DOI 10.1007/978-3-642-40643-0

Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013946545

CR Subject Classification (1998): I.2, F.1, I.4, H.3-4, F.2, C.2

LNCS Sublibrary: SL 7 – Artificial Intelligence

© Springer-Verlag Berlin Heidelberg 2013

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Table of Contents

Knowledge Representation, Reasoning and Logic

Machine Learning

Multiagent Systems

Multidisciplinary Topics and Applications

Metaheuristics

Uncertainty in Artificial Intelligence

.Cloud: Unified Platform for Compilation and Execution Processes in a Cloud

Fernando De la Prieta[1], Antonio Juan Sánchez[1], Carolina Zato[1], Sara Rodríguez[1], and Javier Bajo[2]

[1] University of Salamanca, Deparment of Computer Science and Automation Control
Plaza de la Merced s/n, Salamanca, Spain
[2] Technical Univesity of Madrid, Department of Artificial Intelligence
Bloque 2, Despacho 2101, Campus Montegancedo, Boadilla del Monte, Madrid, Spain
{fer,anto,carol_zato,srg}@usal.es, jbajo@fi.upm.es

Abstract. Compiling is not only running a program that interprets a given source file collection. In the compilation process, it is important the immediacy and way of display the results, and the user interaction. In this research it is proposed the creation of a unified platform (.Cloud) that supports editing, compiling and running applications in multiple languages and that can execute directly into the user's browser without installing any plugins, making it independent of the platform and operating system used. .Cloud will be independent of the platform on which it runs, which will favor mainly to devices with limited resources, both hardware and platform software. The use of a Cloud tool of this type also facilitates the work group within computing projects, allowing multiple programmers working on the same data with optimized workflow.

Keywords: Cloud Computing, AI Applications, Cloud Storage, Utility Computing, Cloud Compiling.

1 Introduction

The latest paradigm to emerge is Cloud computing [14, 2] which promises reliable services delivered through next-generation data centers that are built on virtualized compute and storage technologies. Although at first glance this may appear to be simply a technological paradigm, reality shows that the rapid progression of Cloud Computing is primarily motivated by economic interests that surround its purely computational or technological characteristics [15]. As a result, the number of both closed and open source platforms has been rapidly increasing [10].

The term "Cloud Computing" defined the infrastructure as a "Cloud" from which businesses and users are able to access applications from anywhere in the world on demand. Thus, the computing world is rapidly transforming towards developing software for millions to consume as a service, rather than to run on their individual computers.

To this end, it is necessary to take into account not only the underlying infrastructure, but also the services that are offered to the end user. Cloud computing platforms

C. Bielza et al. (Eds.): CAEPIA 2013, LNAI 8109, pp. 219–227, 2013.
© Springer-Verlag Berlin Heidelberg 2013

has properties of clusters or grids environments, with its own special attributes and capabilities such strong support for virtualization, dynamically composable services with Web Service interfaces, value added services by building on Cloud compute, application services and storage. The infrastructure revolves around the concept of elasticity that autonomous, dynamic and automatic adaption, and learns from past experiences, with the aim of offering computational services of any type. The elasticity model constitutes the core of the system that, if correctly designed, will facilitate the remaining processes and their deployment in any environment independently of the physical features, operating system, etc. In conclusion, since user requirements for cloud services are varied, service providers have to ensure that they can be flexible in their service delivery while keeping the users isolated from the underlying infrastructure.

One of these new kinds of services and applications has to provide the capacity to develop software for cloud computing environments over the cloud itself. In this sense, this study presents .Cloud that is an IDE directly deployed over a Cloud Computing environment. This means, in fact, that .Cloud is an IDE that permit to develop software directly in the cloud (through the browser) without the need of install any kind of software in the user computer. Moreover, .Cloud allows to developers not only to program Web applications (HTML/CSS, PHP, Phyton, Perl an so on) but also traditional software (Java, C, etc.).

This paper is structured as follows, next section shows the state-of-the art of both Cloud Computing and cloud distributed compilation. Section 3 provides an overview of the .Cloud platform; and, finally, section 4 presents the results, conclusions and future work.

2 Compiling Software over the Cloud

The compilation can be defined as the process to translate a program written in a high level programming language into a machine code that the computer, where the program is going to be executed, is able to understand it. This definition has been accepted for a long time, however Cloud Computing is changed the traditional concept of computing environment and, for so, the compiling process has to evolve.

Cloud Computing can be considered as a metaphor for speaking about Internet. This technological paradigm helps to amplify the feeling of decentralization and obfuscation of the origin of information. Actually, Cloud is much more than Internet, or rather, it is over Internet and extends its services. So that, it can be consider as an abstraction of a complex mechanism that simplifies the services and provides a secure remote access to information (among other things).

A Cloud computing environment can be shown from two viewpoints [16]:

- At the *internal level*, the system consists of a set of physical machines (servers), which contribute to the system by means of their computational resources (processing capacity, volatile memory, etc.). These physical server forms the low layer of the infrastructure within the cloud environment. Over this physical layer, there is a virtual layer formed by units of hardware abstraction called virtual machines. This split of the infrastructure in two layers makes possible the external feel of unlimited resources, although obviously the infrastructure is limited to available resources.

- At the *external level*, a cloud computing system is composed of a set of services that are offered to the users. These services are commonly known as XaaS (XaaS: X as a Service) [17]. The most usual division consists in to split the services in three groups: Software, Platform, and Infrastructure. Software and platform services can be considered as web applications: the software layer know as SaaS (Software as a Service) provides a service with GUI (Graphical User Interface) to the end users similarity to the traditional software. the platform layer called PaaS (Platform as a Service) provides a set resources addressed to the developers. Infrastructure layer (IaaS, Infrastructure as a Service) is a layer that offers computational resources (Computational resources, storage, network, etc.) thanks to the virtualization layer described in the internal level.

Actually, the compilation and execution processes are very similar because the process of compiling consists of to act in a given set of files. The difference between executing and compiling is that in the compilation some parameters, such as the immediacy of generate and visualize the results, are less important. Besides, in compiling time, the interaction with the end-user is not import because the communication is only performed at starting and ending of the process.

Although, there are many examples of traditional IDEs (Integrated Developed Environment) such as Netbeans [13], Eclipse [6], JBuilder [12], .NET [1], App Cloud [3], etc. So far, there are few examples of compilation tools that are specific deployed for a Cloud Computing environment. Thus, it is possible to find the tool named Ideone [9] that offers a compiler and debugger for more than 40 programming languages through a web application. Other example is Compilify [5], which is similar to Ideone, and the first beta version allow to develop .NET applications writing in C#. And, finally, Cloud Compiler [4] which is framed under IBM operating systems OS/390 and z/OS, from the end user viewpoint works like a traditional compiler, but the compilation is done in a remote server..Cloud platform.

This section presents the .Cloud which is an IDE that allows to develop focus for and, also, in a Cloud Computing environment. This means that the developers do not have to install any software in computers and they only have to access to .Cloud deployed over a Cloud Computing environment.

.Cloud is deployed in the platform +Cloud [16, 7] that is a Cloud platform that makes it possible to easily develop applications in a cloud. This platform allows services to be offered at the PaaS (Platform as a Service) and SaaS (Software as a Service) levels. Both PaaS and SaaS layers are deployed using an internal layer, which provides a virtual hosting service with automatic scaling and functions for balancing workload. A more detailed description of each layer is provided below:

- **SaaS Layer.** This layer hosts a wide set of Cloud applications. +Cloud as environment offers a set of native applications to manage the complete Cloud environment: virtual desktop, user control panel and administration panel.

- **PaaS Layer.** The PaaS layer is oriented to offer services to the upper layer, and is supported by the lower IaaS layer. The PaaS layer provides services through RESTful web services [16] in an API format (i) the *File Storage Service* (FSS), which provides an interface for a container of files, emulating a directory structure in which the files are stored with a set of metadata, thus facilitating retrieval, indexing, search, etc; (ii) the *Object Storage Service* (OSS), which provides a simple and flexible schemaless database service oriented towards documents; and finally (iii), the IdentityManager (iM), which is the module of +Cloud in charge of offering authentication services to clients and applications;

2.1 +Cloud Architecture

.Cloud is divided into two main and independent components: the client application and the server application. This architecture is shown in Figure 1. Although, both client and server application can be deployed in the same remote server and the access from the end user can be done from an user agent, like a web browser. The advantage of this this splitting is that the client can be moved to another web server or infrastructure that is not directly supported by the Cloud environment; while the server is kept over +Cloud platform and it is in charge to perform the communications with other services within the cloud platform. The communication between them is done using web services.

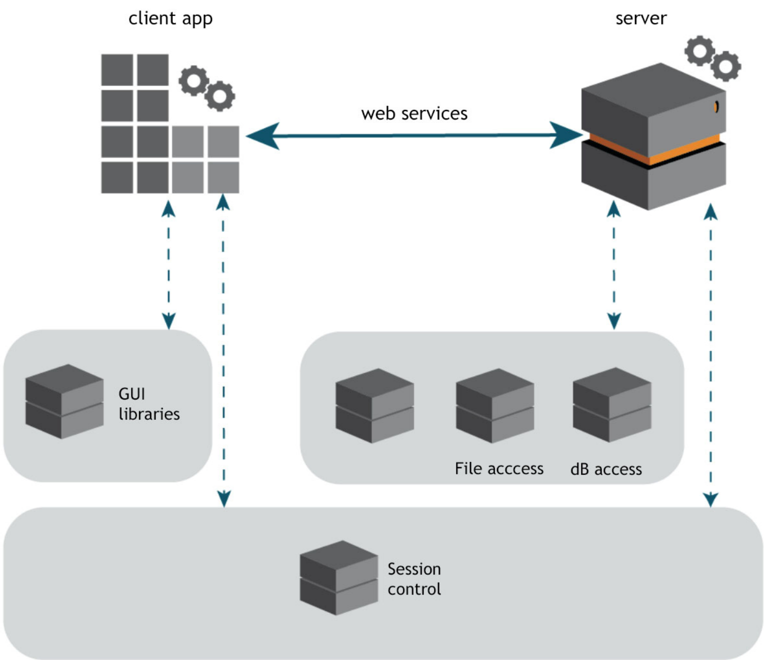

Fig. 1. Cloud architecture

A more detailed description of each layer is provided below

- The server application is in charge of manage the persistence of the data in the OSS and FSS. This server application can be framed in the PaaS layer of +Cloud. The access to OSS (object storage) is done through a specific entity classes called Compilator, Range, User and Project. The file storage is managed by the class Directory and it takes care of read/write operations.
- The client application includes the .Cloud GUI, this means, HTML for the visualization /CSS and Javascript for the control and generation of components. This client is deployed in the SaaS layer of the cloud architecture.

This division facilitates the execution of .Cloud in other user agent such as light clients for mobile device (smartphones, tablets, etc.), heavy desktop applications, etc. These new clients can be created without the need of changing the server side of the application.

2.2 .Cloud Services

Mainly, .Cloud provides two kind of services, one for program execution and another for compiling. Firstly, it is described the **programs' compiling** within the Cloud has to be modeled such as another kind of service (library, tool, etc.). To this end, the compiling process has to be structured such as black box where the end user invokes the service, the service execute the compilation process, and then the users gathers the results. This high level schema is shown in Figure 2.

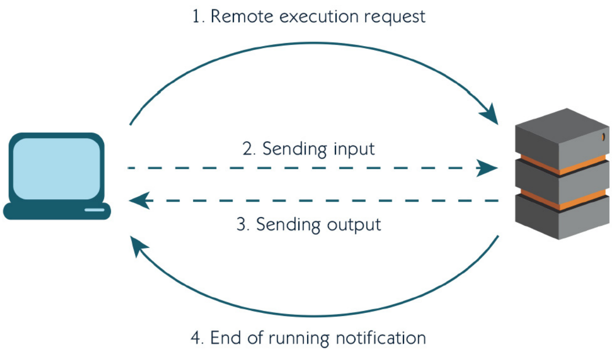

Fig. 2. Remote compiling process mechanism

As we said, the mechanism of **remote execution** is similar to the compiling process. It is shown in Figure 3. In the execution process, the client makes a remote execution request. This service creates two pipes within the system, one for read of data as input of the program, and the other one to write data as output of the program that is executed.

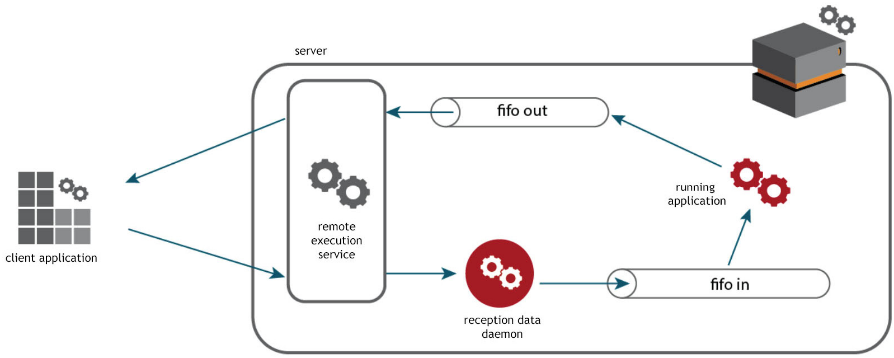

Fig. 3. Remote execution process mechanism

Both pipes are launched jointly with the program in execution, besides a daemon is launched and it is in charge of read the program output and send it to the execution service. This service, finally, returns this output to the client to be shown in the user agent, usually, a browser.

This daemon has a key role in the architecture, because it is in charge to keep the pipe open during the execution of the program. If it does not exist, the pipe will close

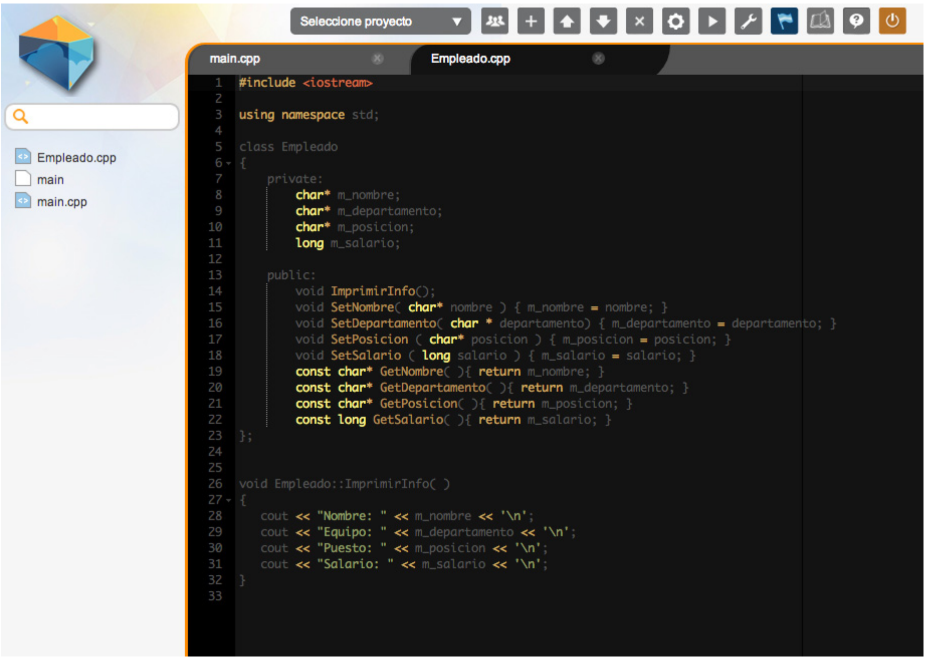

Fig. 4. Project editor

after the first output. This also triggers that the execution process will be canceled due to operating system restrictions. The writing daemon and the service exchange the information by means of a socket. This socket allows keeping open the communication during all the remote execution and avoiding busy-waiting.

With regards on the communication the first approach was to use RESTful web services, however there were some difficulties with this approach because it is not able to keep the session between the peers open. Finally, we have chosen to use an intermediate approach with REST service and PUSH technology. All services of .Cloud use JSON as exchange data language.

The server application has to provide a communication mechanism between the client and service, to gather and share the importation in execution time. This time depends on many factors (bandwidth, etc.); these factors also depend on the communication strategy (PUSH technologies). .Cloud has used long-polling mechanism in front of other strategies such as web-sockets, because long-polling offers a more reliable communication and more compatibility with the web browsers.

With the regards on the design of the GUI, it has been taken as reference other graphic features widely distributed web applications as Google Apps or iCloud, in which a simple and straightforward design helps the user to quickly guided by the interface. The interface is shown in Figure 4.

2.3 .Cloud Integration in +Cloud

.Cloud consider an environment with the following characteristics:

- +Cloud provide an object-oriented database (OSS) and storage files service (FSS).
- .Cloud has to allow the scalability of the system in terms of deployment of the system in many virtual machines.
- There is a virtual machine that centralized the information.

In order to allow that .Cloud can be executed in a Cloud environment, it is necessary to take into account that .Cloud can be executed in several virtual machines at the same time. If different end users are working with the same project at the same time, and this project is deployed in different virtual machines, .Cloud has to provide a process to update the information in execution time among all copies of the project (each of them used by a specific end user). To this end, there is a background program that ensure that every information is stored in the persistence layer (FSS and OSS) and at the same time this information is updated among all virtual machines that provide resources to .Cloud.

3 Results and Conclusions

This study presents .Cloud that is IDE specially developed to be deployed within a Cloud environment. .Cloud has been test in many traditional user agents (Internet Explorer 9, Safari, Google Chrome) as well as user agents of tablets and Smartphones

Table 1. Comparision between .Cloud and other similar platforms

	Netbeans	Eclipse	JBuilder	Visual Studio	App Cloud	Ideone	Cloud Compiler	Compilify	.Cloud
Projects	✓	✓	✓	✓	✓		✓		✓
Desktop version	✓	✓	✓	✓	✓		✓		
Browser version						✓		✓	✓
Debug tools	✓	✓	✓	✓	✓		✓		
Cloud storage					✓		✓	✓	✓
Multilingual	✓	✓	✓	✓					✓
Test tool	✓	✓	✓	✓	✓				
Workgroup tool	✓	✓	✓	✓			✓		✓
Additional complements.	✓	✓	✓	✓	✓				
Compiling and execution in a Cloud.					✓	✓	✓	✓	✓

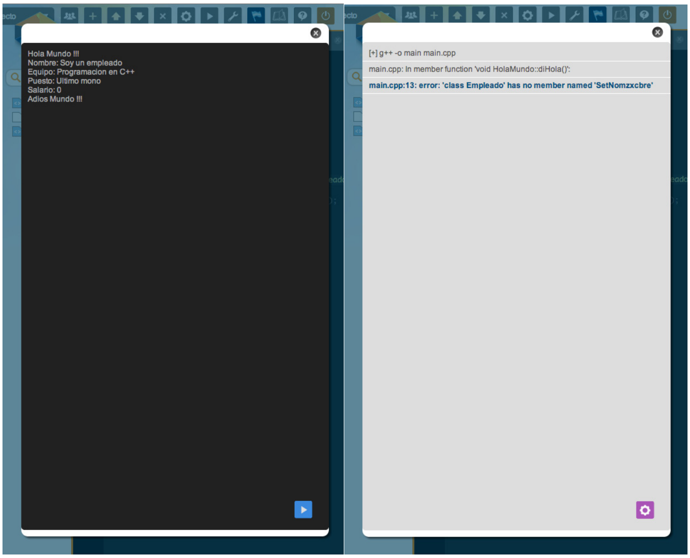

Fig. 5. Compiling and executing a Project

(Chrome Mobile and Safari for iOS) as shown in Figure 5. The forthcoming steps within the development of this platform will be the test the platform in a real environment with real end users.

As a conclusion, .Cloud presents a set of characteristics that are not provided for other IDES (traditional or cloud-based). Table 1 provides show a comparison between our platform and other platforms.

Acknowledgments. This research has been supported by the project SOCIEDADES HUMANO-AGENTE: INMERSION, ADAPTACION Y SIMULACION. TIN2012-36586-C03-03. Ministerio de Economía y Competitividad. Feder Funds.

References

1. .NET Framework, `http://www.microsoft.com/net` (Access: April 22, 2013)
2. Carolan, J., et al.: Introduction to Cloud Computing. Architecture. White Paper, 1st edn. Sun Microsystems (June 2009)
3. Cloud APP Studio, `http://www.cloudappstudio.com/www/index.html` (Access: April 22, 2013)
4. Cloud Compiling, `http://www.cloudcompiling.com/` (Access: April 22, 2013)
5. Compilify, `https://compilify.net/` (Access: April 22, 2013)
6. Eclipse, `http://www.eclipse.org/` (Access: April 22, 2013)
7. De la Prieta, F., Rodríguez, S., Bajo, J., Corchado, J.M.: A multiagent system for resource distribution into a Cloud Computing environment. In: Demazeau, Y., Ishida, T., Corchado, J.M., Bajo, J. (eds.) PAAMS 2013. LNCS, vol. 7879, pp. 37–48. Springer, Heidelberg (2013)
8. (2008), `http://www.gtsi.com/eblast/corporate/cn/09_09_2009/PDFs/Sun.pdfKLEMS`
9. Ideone, `http://ideone.com/` (Access: April 22, 2013)
10. Peng, J., Zhang, X., Lei, Z., Zhang, B., Zhang, W., Li, Q.: Comparison of several cloud computing platforms. In: 2nd In- ternational Symposium on Information Science and Engineering, ISISE 2009, pp. 23–27. IEEE Computer Society (2009)
11. Jacobson, I., Booch, G., Rumbaugh, J.: The united software development process. Addison Wesley (1999)
12. JBuilder, `http://www.embarcadero.com/products/jbuilder` (Access: April 22, 2013)
13. Netbeans IDE, `https://netbeans.org/` (Access: April 22, 2013)
14. Mell, P., Grance, T.: The Nist Definition of Cloud Computing. In: NIST Special Publication 800-145. NIST, 1-3 (2011)
15. Buyya, R., Yeo, C.S., Venugopal, S.: Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In: Department of Computer Science and Software Engineering (CSSE), The University of Melbourne, Australia. He, pp. 10–1016 (2008)
16. Heras, S., De la Prieta, F., Julian, V., Rodríguez, S., Botti, V., Bajo, J., Juan, M.: Corchado Agreement technologies and their use in cloud computing environments. Progress in Artificial Intelligence 1(4), 277–290 (2012)
17. Schaffer, H.E.: X as a Service, Cloud Computing, and the Need for Good Judgment. IT Professional 11(5), 4–5 (2009), doi:10.1109/MITP.2009.112