4th International Workshop on Software Services

Workshop information
Message from the Chairs
Papers by authors

Bled, Slovenia
25 October 2012

In conjunction with
The 1st International Conference on CLoud Assisted Services (CLASS 2012).
PROCEEDINGS
4th International Workshop on Software Services
—— WoSS 2012 ——
# Table of Content

Table of Content ............................................................... i

Message from the WoSS 4 Chairs ............................................. ii

E. Õunapuu: Model of e-government: Estonian experience ......................... 1

P. Czarnul: Dynamic Ranking of Cloud Providers ................................... 6

S. Šperka: Cloudication of Legacy Information Extraction System ............... 9

M. B. Vleju: A Client-Centric Identity Management Tool for Small and Medium Enterprises Using Cloud Services ................................................... 15


M. Dolenc: ISES Virtual Energy Lab: an Overview .................................. 22

G. Ferschl: Migrating a Simulation Framework from the Cloud to the Sky ........ 25
Message from the WoSS 4 Chairs

Welcome to the 4th Workshop on Software Services (WoSS 4). WoSS 4 is colocated with the 1st International Conference on Cloud Assisted Services (CLASS 2012), Bled, 22-25 October 2012.

The WoSS series is dedicated to recent significant developments in the field of software services. WoSS intends to be an open forum for academics, practitioners, and vendors, allowing them to discuss the current trends and scientific and technological challenges, such as service quality assurance, adaptability, reliability, interoperability, and automating service-oriented application construction and management. The first three editions of this forum were organized with the support of the FP7-ICT SPRERS project, with the aim to highlight the achievements of the on-going European collaborative projects subscribing to FP7-ICT programme and the activities of the teams from new Member States in the software services area. The fourth edition of WoSS focusses on Cloud and Grid Applications for Science and Industry and is supported by the FP7-ICT mOSAIC project. This year WoSS 4 accepted 6 Extended Abstracts for presentation at the Workshop, which is taking place on October 25, 2012 in two tracks along with the CLASS 2012 Conference programme. The Proceedings of the WoSS 4 are published online at the CLASS 2012 conference website.

We would like to take this opportunity to express our deep appreciation of the efforts of the Program Committee members who provided the technical reviews of the submitted papers: Pawel Czarnul, Janis Grabis, Matjaž B. Jurič, Andras Micsik, Enn Ūnapuu, Tomas Pitner. We would also like to gratefully acknowledge the CLASS 2012 Organisers for their strong support. The success of a workshop, like this one, requires also the dedication and contribution of several volunteers and staff support. Particularly, we would like to thank Gašper Stegnar for his overall support in a variety of technical issues.

We would like to take this opportunity to invite the WoSS 4 and the CLASS 2012 contributors to submit their extended works to the Special Issue of the Informatica journal also dedicated to Grid, Cloud and Sky Applications for Knowledge-based Industries and Businesses. Publication decision will be reached by following a standard peer-reviewing process.

We trust that you will enjoy this unique and interesting program. We look forward to seeing you at the WoSS 4 event in Bled.

Vlado Stankovski, University of Ljubljana, Slovenia
Dana Petcu, West University of Timisoara, Romania
Model of e-government: Estonian experience

Enn Ōunapuu
Institute of Informatics, Tallinn University of Technology, Raja 15, Tallinn, Estonia
Enn.ounapuu@ttu.ee and www.ttu.ee

Keywords: e-government, e-government model, maturity model

Research into e-government is relatively new. Nevertheless, much contemporary thinking and writing about e-government is driven by normative models that appeared less than a decade ago. The author presents empirical evidence from experience of implementing local e-government solutions in the Estonia to test whether these models are accurate or useful for understanding the actual development of e-government. The development of the information society and related activities have been the priority of European Union (EU) policies in the recent years. For the period 2007-2013, millions of euros of EU Structural Funds (SF) have been earmarked for carrying out activities and developments related to the transition of the Estonian public sector to the information society. Many projects, both successful and unsuccessful, have been initiated within the SF. Current paper describes works currently in progress (mean which are done past 8 years) that aim at improving information governance to facilitate the transition of local governments to the information society. In the article a e-Government Model, eGM is presented. This model is used as basis for analysis of local governments’ development. The authors offer grounded observations about e-government implementations that will be useful to scholars and practitioners alike.

1 Introduction

The connection of information governance with the management of organisation processes and workflows, along with digitalisation, has been a clear trend in recent years. Process design is the conscious evaluation and organization of the tasks that a business process is composed of. Designing a process that improves current performance and/or conformance is a challenging task that requires a cooperation of multiple specialists to define organizational strategies, goals, constraints and resources.

One challenge that many IT organizations undertake when looking at vendor-provided cloud products is determining which products best meet their business needs. After products have been chosen, it is another effort to turn the vendor-provided products into cloud solutions that meet the business needs of the organization. Going through this process for a number of customers, we have found that several common lessons can be learned to allow quicker time-to-value and overall success of a customer’s cloud project.

The Intelligent Enterprise is able to interact with its environment and change its behavior, structure, and strategy - behaving actually as an intelligent entity.

It is able to adapt to rapid changing market circumstances, gradually change its business model, and survive into the next market cycle. The Intelligent Enterprise as we see it is characterized by its ability to learn from and adapt to changes in its environment and reinvent itself, sometimes with surprising results. In order to keep up with the rapidly changing demands of doing business, most enterprises implement increasingly complex IT solutions. But these interconnected complex solutions introduce new inefficiencies within the organization.

We see future Intelligent Enterprises deriving
efficiencies through the automation of their core business processes and the exploitation of knowledge inherent in their organization. Their ability to respond quickly to changes will improve significantly as the knowledge base and “intelligence density” within the enterprise grows and problem-solving capabilities improve dramatically. Intelligent Enterprises will form dynamic partnerships with other enterprises to create dynamic business ecosystems, which will be self-managed, self-configured, and self-optimized. In short, future enterprises will become smarter - more intelligent - and by doing so will evolve automatically into organizations more suited to their changing environment.

First, it is critical for customers to have clear requirements of what they are trying to achieve with the cloud products they are purchasing. Often, separate parts of the organization have different goals that influence the scope and design of the solution. This can be detrimental to the success of the project. Gaps in expectations and requirements should be addressed early in the project to avoid larger problems later in the project.

Another key to successfully turning cloud products into cloud solutions is having clear standards and policies. This approach is critical from a service catalog perspective, when a service is planned for delivery through a public or hybrid cloud. A complete and clear set of policy statements is required, so they can then be automated to permit the flexibility that is usually required for cloud solutions. Furthermore, the service catalog should clearly state what the cloud solution will provide and the service levels around these services. This way can help ensure that expectations are met from users and customers of the cloud solution. If a customer has good standards or policies around certain aspects of service management, it is recommended to leverage those already existing policies and standards if possible. Many times this can save a lot of time in a project and allow for best practices, which might already exist in an organization, to be reused.

2 E-government model

The e-Government Model, eGM, represents a holistic and generic multi-discipline approach to study and illustrate the issues, that are considered important for eGovernment development, adoption and large scale breakthrough. The EU define the e-government as “organizational development in public administration, which takes advantage of ICT in combination with organizational changes and new competences”. The eGM model serves as a reference point for research, development, demonstration, implementation and benchmarking for the e-government development.

The eGM is inspired by our experience in e-commerce models [4].

The key background principle is the emerging new paradigm leading to a more flexible knowledge, resource and assets sharing ICT enabled business environment based on well defined interfaces and virtualization of infrastructures. The process is to be powered by eGM model support promoting a more collaborative mindset, the role of standards, regulations and contracts’, measures strengthening trust and confidence, and measures of alignment.

The scene of business practices is a moving target. The rapid development and new offerings of information and communication technologies

![Figure 1: E-government model](image-url)
new tools and application opportunities for implementing ICT enabled business practices. Likewise new business theories and business models appear setting pressure for streamlining business practices through new patterns, structures and value adding means.

Migration to new ICT enabled business practices is not easy. Attractive technologies and new business model concepts are basically available, but difficulties exist in their implementation in the business environment. Many technical, organizational, cultural and human factors are involved, providing both drive and inertia.

3 E-government model usage in Estonia

Drivers

The principles to be followed in the development of the information society in Estonia are the following [6]:

- the development of the information society in Estonia is a strategic choice with public sector leading the way in pursuing its principles;
- the information society is developed in a coordinated manner in co-operation between the public, private and third sector;
- the public sector is a smart customer, ensuring that in public procurements as much freedom as possible is left for innovative solutions;
- the information society is created for all Estonian residents, whereas particular attention is paid to the integration of social groups with special needs, to regional development and to the strengthening of local self-initiative.

Enablers

The two key ingredients in the infrastructure are the X-ROAD (Figure 2) and Digital-ID (Figure 3). The X-Road is a critical tool that connects all the decentralized components of the system together. It is the environment that allows the nation’s various e-services databases, both in the public and private sector, to link up and operate in harmony no matter what platform they use. Digital ID is the nationally standardized system for verifying a person’s identity in an online environment. It opens the door to all secure e-services while maintaining the highest level of security and trust [5].

Readiness

E-Readiness is generally defined as the degree to which a society is prepared to participate in the digital economy with the underlying concept that the digital economy can help to build a better society [7]. Estonia has been previously assessed by various E-readiness reports as one of the most successful adaptors and uptakers of new communication technologies in CEE countries.

The crucial factors for developing information society in Estonia have been:

1. building up modern infrastructure;
2. Tiger’s Leap Project in computerizing schools and universities;
3. adopting regulations for information society;
4. government IT-programs;
5. collaboration between the government, private sector and non-governmental initiatives; and last but not least
6. luck.

4 Estonian e-government success facts


Electronic ID-card. As of January 2012, more than 1.1 million people in Estonia (almost 90% of inhabitants) have ID cards. The Estonian ID card serves as an identity document and, within the European Union, also as a travel document. In addition to its physical use, the card is also used as proof of ID when utilising online services. In other words, the ID card is the key to almost every innovative e-service in Estonia. Inside this small document is a chip that not only holds information about the card’s owner, but also two certificates, one of which is used to authenticate identity and the second to render a digital signature. Thanks to its security, the card is used in many web environments where ID verification is needed. Internet banking, participation in e-elections, buying public transportation tickets, and much more can be accomplished using the electronic ID card. The ID card is secure, since PIN codes are also required for the card’s operation.

In addition to the ID card, one can also use a mobile phone to identify oneself for online services. This is even more convenient since one does not need an ID card reader for the computer. A mobile phone can act as a card and a card reader at the same time.

E-TAX Board. Estonian citizens can declare their income taxes electronically over the internet. Estonia’s e-Tax Board offers a pre-completed form which makes it easy and fast to submit your tax return. The system identifies persons with the help of an ID card or mobile ID. A citizen must only log on to the e-tax system, check the information that is automatically assembled, make additions or changes (if necessary), and approve the declaration. The service has become so popular among Estonian residents that in 2012 over 94% of income tax declarations were presented through the e-tax system.

Digital prescription. On 1 January 2010, an IT solution was applied to Estonia’s health care - a digital prescription system. In the past, patients had to carry paper prescriptions with them to the pharmacy. This system had several weaknesses: it was easy to lose the paper, the handwriting of the doctor could be illegible, etc. Electronic prescriptions have solved these problems because all prescriptions are sent to a central database. When the patient goes to the pharmacy, the pharmacist receives the prescription from the central database - there is no chance for the patient to lose the prescription or any risk that it might be unreadable.

The e-Governance Academy is a non-profit information society, development and analysis centre that aims to share Estonia’s experience in the areas of e-government, e-democracy, and information technology education. More than 700 individuals from 36 different nations have come here to study, including representatives from Canada, Japan, Georgia, India, Namibia, and Pakistan. Estonian experience and knowledge have aided many nations in making their election processes more transparent, democratic, and less encumbered by bureaucracy.

Since Estonia is a middle-income transition economy, it should be compared to a group of similar countries [8]. The most appropriate com-
parison would be with other Central and Eastern European countries. The closest fit with the definition and most straightforward operationalization of the concept is found in the 2004 study by The Economist Intelligence Unit (EIU) ranked Estonia as first of ten on the list of new European Union (EU) member states and Turkey (Economist Intelligence Unit, 2004, 2). Estonia scored 5.87 points out of 10, followed by the Czech Republic’s score of 5.67, and Slovenia’s 5.33. The EU benchmark study of online public e-services ranked Estonia eighth in the EU for online sophistication of its public sector services and fourth for online service availability (European Commission, 2005, 26). In both categories Estonia is a long way ahead of other Central and Eastern European countries and is clustered together with the richest EU members. The Economist Intelligence Unit and IBM Institute for Business Value E-readiness Index ranked Estonia 26th in the world and 1st in Central and Eastern Europe in 2005 (Economist Intelligence Unit, 2005).

5 Conclusion

In this study The e-Government Model, eGM is presented. This model has set out to analyse the significance of the introduction of public e-services in many Estonian municipal organizations. The model can be evaluated as useful.

References


Dynamic Ranking of Cloud Providers

Pawel Czarnul
Narutowicza 11/12, 80-233 Gdansk, Poland
pczarnul@eti.pg.gda.pl and http://pczarnul.eti.pg.gda.pl

Keywords: cloud computing, ranking cloud providers

The paper focuses on how to create an effective dynamic ranking service for IaaS, PaaS and SaaS cloud providers. It considers building of a quality model for this purpose along with definition of quality measurement procedures. The paper discusses several techniques known from already existing price comparison engines that could be modified and adopted for comparison of cloud providers. Furthermore, a technique for filtering measured data is proposed, in particular to avoid vendor lock-in issues.

1 Introduction

Cloud computing has become widespread in today’s world with many offerings regarding infrastructure, ready-to-use platforms and services [4]. These are formally described as:

– IaaS – Infrastructure as a Service - making an infrastructure (computing, storage, operating system) with a given configuration available to a client, examples: Google Compute Engine1, Amazon Elastic Compute Cloud (EC2)2, RackSpace Cloud Servers3, Rack Space Cloud Files4,

– PaaS – Platform as a Service - offering a complete platform with particular software required by users; examples include: Aneka [1], Google AppEngine5, Windows Azure6, RedHat Openshift7, RackSpace Cloud Sites8,

– SaaS – Software as a Service - particular software that is managed by its provider and accessed by users from any location. Examples include Google Apps9 and Salesforce10.

Following search engines and price comparison tools and engines for the traditional marketplaces, there have emerged tools for comparison of cloud offers as well. For instance, as of this writing a web search on “IaaS ranking” returns several surveys on IaaS: either static analyses11 12 13 or rankings that depend on actual parameters of the offers (such as prices) that can change in time14 15. Platforms such as Cloudorado16 allow to pre-select user requirements such as required processor computing capabilities or storage and return a ranking based on that. FindTheBest allows to select a cloud provider based on its type (IaaS, PaaS) but also the control interface, software license or subscription type.

It seems, however, that many of these rankings use unstructured quality comparison models, do not consider how qualities have been changing over time for providers and do not address

1http://cloud.google.com/products/compute-engine.html
2http://aws.amazon.com/ec2/
3http://www.rackspace.com/cloud/cloud_hosting_products/servers/
4http://www.rackspace.com/cloud/cloud_hosting_products/files/
5https://developers.google.com/appengine/
6http://www.windowsazure.com
7https://openshift.redhat.com/app/
8http://www.rackspace.com/cloud/cloud_hosting_products/sites/
9http://www.google.com/Apps
10http://www.salesforce.com/eu/
11http://my-inner-voice.blogspot.com/2011/02/here-are-results.html
12http://insidehpc.com/2011/02/10/survey-results-on-cloud-iaas-providers/
13http://www.opsource.net/Info-Tech-Cloud-IaaS-Vendor-Landscape
14http://www.cloudreviews.com/top-ten/cloud-hosting-services.html
15http://cloud-computing.findthebest.com/
16http://www.cloudorado.com/
issues such as vendor lock-in. It is a known fact
that some Internet providers or shops used to of-
fer very cheap prices to gain a market share (by
being on top places in comparison rankings) only
to deceive some customers later. The paper pro-
poses a quality model for a dynamic ranking of
cloud providers that addresses these issues.

2 Quality Assessment of Cloud
Offers

However, although the aforementioned services
can be invoked dynamically i.e. when there is an
actual need for a particular service, it is necessary
to incorporate measurable quality assessment of
the given service – this comprises several aspects
that need to be addressed:

1. quality model/ontology that defines metrics
to be measured,

2. quality measurement procedures – e.g. how
frequently the metrics should be measured
– this may be different for various metrics;
for instance availability may require more fre-
cquent monitoring than the price,

3. filters applied on top of the measured values
– such may be used to address several issues
such as:

- preventing from short-term peaks in
measured values to affect output; poss-
ibly only longer lasting changes should
do that,

- preventing from one or few providers to
take top places all the time by offering
too good to be true conditions,

- considering or not sudden changes in the
history of the provider which may affect
user decisions who might be afraid of
similar changes in the future – it may
depend on the user whether he or she
wants to consider this aspect.

For quality, it is possible to adopt and extend
the already used techniques for marketplaces in
the Internet. Namely, evaluation of the providers
on a numerical scale such as [0,10] which is of-
fered for almost any price comparison engine to-
day along with physical location of a particular
provider. In this case, a quality ontology is pro-
posed for quality service evaluation of particular
IaaS, PaaS, SaaS that will incorporate the follow-
ing:

`accessibility_location_x` – may characterize
the network between the client in `location_x`
and the service, several entries of this type
could be inserted,

`availability` – characterizes the availability of
the service itself. It could be measured by
e.g. checking its availability vs availability of
other services/servers in a similar geograph-
ical/provider location,

`cost-effectiveness` – evaluated by clients,

`reconfiguration ability` – applicable to IaaS
and PaaS,

`access` – how easy it is to access the infrastruc-
ture and upload/download/execute applica-
tions.

For filters applied on top of measured values
the author suggests to apply a low pass filter on
the resulting measured metrics. For instance, a
one time peak in measurements of a certain value
might not change the overall score of the given
metric. Only a longer lasting change would ini-
tiate this. A simple average would do this. The
regular average suffers from the historical effect
i.e. results from the past affect the final average
in the same way as the last input. It may depend
on the client whether to rely more just on recent
measurements. This could be further extended to
a running score e.g. a running average of 10 or 100
values. Alternatively, the history of the provider
might be important for the given client.

In order to avoid a situation when one provider
wants to dominate the given segment of the mar-
ket by e.g. using too good to be true prices it is
possible to consider a certain number of best offers
and rotation on the first ranking places, provided
that results returned for the services are closer to
each other than a predefined threshold. Even one
company could then try to use different providers
for parts of their businesses to avoid the lock-in
problem.
3 Proposal of an Evaluation Engine

The proposed approach in a way can be seen as a solution aiding sky computing [2] as the proposed engine tries to sort out available cloud options and offer best options at a higher level of cloud integration.

As mentioned above, the goal of the engine is to be able to:

1. monitor QoS dynamically,
2. avoid potential vendor lock-in problem.

From the client point of view, it would be desirable to have access to a comparison engine like Cloudorado with the aforementioned features. First of all, the engine will consider three categories of: IaaS, PaaS and SaaS. In order to make search better, two solutions are feasible:

1. categorization of features such as hardware and software parameters desired by the client:
   - memory size,
   - processor/core capabilities,
   - GPU capability,
   - storage,
   - operating system,
   - particular software,
   - access interface.

   This is especially suitable for IaaS and PaaS offerings.

2. full text search as in [3]. This allows formulation of desired functions in the form of human readable text. Useful mainly for SaaS as it would allow searching and presentation of SaaS offers for particular application.

The full text search mechanism could also be applied to any type of service when looking for comments of already existing clients.

This would also naturally lead to creation of runtime registries of particular IaaS, PaaS and SaaS offers. SaaS options could then be categorized into various categories. One possibility is to adopt the well-known technique from photo sharing sites i.e. augmenting descriptions with tags. Then selection of particular tags would narrow search results.

4 Experiments

In this section experiments will be provided on what particular ranking results will look like for particular behavior of cloud providers to assess how the aforementioned filtering techniques work.

References


Cloudification of Legacy Information Extraction System

Svatopluk Šperka, Petr Škoda, Pavel Smrž
Brno University of Technology, Faculty of Information Technology, IT4Innovations Centre of Excellence, Bozetechova 2, 612 66 Brno, Czech Republic
{isperka, iskoda, smrz}@fit.vutbr.cz

Keywords: cloud, legacy systems, ReReSearch, mOSAIC, information extraction

This paper presents the process of the cloudification of a legacy information extraction system targeting scientific papers. The original system is described along with architectural and implementational decisions that were made in order to gain performance and to conform to the philosophy of the mOSAIC cloud middleware that is used as a basis for implementation.

1 Introduction

With consistently growing popularity and availability of cloud computing, the focus of developers and enterprises turns also to legacy applications. The importance of the legacy systems lays in the fact that they are often crucial parts of the organisations’ workflows, and it would be error prone and/or cost ineffective to redevelop them. In the same way the scientists do not have enough resources or time to redevelop numbers of legacy applications. Reasons for porting the legacy applications into a Cloud are mostly the same as for targeting a Cloud with new ones, e.g., great horizontal scalability, better cost efficiency and easy access to vast amount of computational resources on demand [1][2][3].

At our faculty we develop a system for building an accessible knowledge base about scientific research called ReReSearch. This system consists of many old modules that we need to run in adequate number of instances in order to achieve the required overall performance [4]. We can say that these modules are legacy because the cost of their porting into another language is unacceptable - they contain fine tuned complex natural language processing algorithms. Current way of parallelisation is a usage of the faculty’s N1GE-based grid but it brings various limitations and problems. To overcome this we decided to port some modules to a Cloud. The first ported module, which is a crucial part of the ReReSearch, is the scientific paper information extraction module [4].

This paper addresses a process of porting the information extraction module written in Python language to the mOSAIC cloudware solution. We start by describing the ReReSearch and mOSAIC systems in sections 2 and 3, and continue by analysis of possible porting approaches in section 4. Then we describe the design of communication interface of the Cloud application in section 5 and the implementation of mOSAIC-based solution in section 6. Finally, we conclude by discussing further steps in development.

2 ReReSearch System

ReReSearch is an experimental project being developed by our faculty’s Natural Language Processing group. It aims at building a knowledge base and derived personalized portals about research. The key entities it operates on include researchers, teams and universities, papers, reports and deliverables, books, journals, proceedings and various collections, conferences, workshops and seminars, projects and funding agencies. Information on all of these entities has to be interconnected in order to be useful. To gather the data, the system first identifies relevant sources on the Web and then downloads and processes specific web pages and papers. In order to transform data from unstructured form into a structured one, in-
formation extraction methods are applied.

Papers are collected by special crawlers that search the Web for pages possibly containing links to scientific papers (e.g., online proceedings or lists of publications linked from homepages of authors that the system already “knows”). Printable formats (PDF, PS, DOC, etc.) are transformed into a semi-structured text, i.e., text-structuring information such as sections or list items can be recovered. Then the information extraction from the text itself is being done. This process takes from a few seconds to several minutes depending on the complexity and length of the document.

It is hard or practically impossible to exactly predict the number of scientific papers that the system can acquire in a given time and it is essential to process all the acquired documents as fast as possible in order to maintain the database actual. Currently, we employ the N1 Grid Engine (N1GE, formerly SGE) to process large batches of papers [4].

On N1GE we compute serialized extractions of ten documents per job since scheduling jobs with only one document would result in grid’s poor performance because of the scheduling overhead. Whole extracting process, from printable format to structured metadata, of one document is then carried out on one grid node. Executables of the metadata extractor that is written in Python are run from fileserver so we do not need to install the extractor on every node. After the successful extraction, resulting metadata are copied back to the fileserver and the ReReSearch control system takes care about uploading them into the database.

There are some drawbacks of current metadata extraction solution using the faculty grid. Mainly, the grid is not dedicated to our research group so it is often highly utilized by other groups. Secondly, the time from finding a document by ReReSearch system to resulting metadata is pretty long (i.e. we must wait for ten documents before enqueuing it to grid’s queue and then all of them must be extracted at once). Lastly, the faculty grid is primarily intended to a new research so it should not be occupied by projects in their deployment phase.

3 mOSAIC Cloud Middleware

It was essential for us to avoid the Cloud provider lock-in - ReReSearch is a representative of systems that can profit from the portability and deployment across clouds and cloud providers. The mOSAIC middleware provides such benefits. It is a language- and platform-agnostic API for usage of multi-Cloud resources and at the same time a portable platform for utilization of Cloud services based on the API and Cloud usage patterns [5]. It allows one version of an application to be deployed to any supported provider. Moreover, it aims at auto-scalability of applications [6], i.e., elasticity.

The mOSAIC’s API has several layers: Connector API abstracts types of cloud resources commonly offered by cloud providers such as message queues, key-value stores or distributed file systems. Driver API is at the bottom of the mOSAIC API hierarchy; it sits on top of a native API for a particular resource and enables a uniform access protocol. This layer offers plugins where each plugin enables access to one implementation of a resource type. Intermediate Interoperability API ensures language independency of the API and thus it mediates between the Connector API implementation and a compatible driver implementation. Connector API and Cloudlet API - the APIs offered by the platform to application developers - are currently only available in the Java programming language but Python and Erlang versions are being developed.

Cloud components, that implement parts of an application’s functionality, are called cloudlets. They use the Cloudlet API that handles lifecycle of cloudlets, enables initialization, configuration, migration and obtaining bindings to used resources [6]. There are one or several containers within which one or more cloudlet instances execute. The number of instances is under control of the container and is managed in order to grant scalability [7], i.e., the mOSAIC platform controls scaling up and scaling down of an application. In order to ensure the scalability, components should communicate indirectly - usually by using message queues as intermediary so more cloudlet instances can dispatch messages from the same queue.
4 Identifying Architecture

The scientific paper information extractor is easily parallelisable by just running many instances simultaneously because there is no dependency between extractions of individual documents. But there are still several approaches to do such computation in a cloud environment. These differ mainly in amount of necessary changes in the original application.

4.1 Minimal Changes in Original Application

It is possible to prepare a software package, very same to the current N1GE one, to be run in a Cloud. The mOSAIC cloudlet would then execute the whole process from a printable version to metadata for each document.

The main advantage of this solution is the simplicity. There is practically no workflow to be programmed in mOSAIC, everything is run as one cloudlet, and there is no communication between components. Scaling is achieved by running more computing nodes with more cloudlet instances. However, there is no space for further optimization or speed up. The situation is better than on N1GE because each document is processed alone (no need to process ten documents at once as on N1GE), but this solution still does not bring any parallelization in the scope of one document.

Further step could be to separate metadata extraction from plaintext transformation. The processing of one document actually consists of two different parts - the transformation from a printable format to a plaintext and the information extraction process itself. By separating these two parts we would get two cloudlets connected by a queue. The mOSAIC workflow then consists of the document to plaintext transformation and the information extraction.

This design is still very straightforward because even in N1GE solution there are two subprograms - one for each part - and there is a text file passed from the document to text transformer to the information extractor. On the other hand, this solution doesn’t bring many improvements over the first one in terms of parallelization.

4.2 Decomposition of Original Application

Finally, we can dig deeper into the original application and identify its inner structure. Information extractor consists of multiple metadata extractors, e.g. language identifier, email extractor, entity extractor or document splitter (see Figure 1). Then in mOSAIC there is a cloudlet for each extractor and communication between them is done through queues. The original extractors must be interfaced so that we can run them separately in cloudlets.

In order to do that, we need to precisely analyse original information extractor, write new wrapper for each extractor, and design new interfaces between newly separated parts. Advantage, however, is that with separated extractors we can easily parallelise the information extraction process of one document. Extractor cloudlets can then run on many nodes and time needed to extract one document can almost reach the time of the slowest extractor - more precisely the time of a longest chain of extractors that depend on each other and thus must be pipelined. Moreover, we can scale up the number of extractors’ cloudlets with respect to their average speed i.e., the slow extractors will have slightly more instances than the fast ones.

Figure 1: Metadata extractor - from document source to metadata.
Because the speed of each document is important for the ReReSearch system, we decided for the last option. This also brings us the ability to easily add new extractors and to run only necessary extractors for different kinds of documents.

5 Communication Interface

In order to be able to communicate with the system, it needs to be given an interface for the outside world. Conceptually, there are two communication use cases the interface must satisfy - requesting extractions and their monitoring.

As there are only these two trivial use cases, a lightweight communication mechanism is appropriate. RESTful interface based on HTTP protocol fulfills this requirement perfectly. Moreover, it is easy to implement and interact with - given the abundance of tools, libraries and implementations.

Two mentioned use cases map to the two HTTP methods. For requesting an extraction, there is the POST method available at URL of the form http://ip\$/documents. The body of the request must contain a JSON object (Content-Type header of the request must therefore be “application/json”) with the field url whose value is the URL of the document to process.

The request can be either successful or not. In case of a success, the response code is 202 (“The request has been accepted but the processing has not been completed”[8]). Content type of the body is again “application/json” and the JSON object is of the form:

```json
{
    "url": "<document_url>"
    "url-hash": "<url_hash>"
    "status-url": "http://<ip>/documents/<url_hash>/status"
}
```

The url-hash field contains a hash of the URL computed by SHA-256 function. The status-url field contains the URL where monitoring information will be available.

Results of extractions themselves are actively pushed to the control system through its HTTP gateway because it is necessary for the application to be a seamless part of the ReReSearch system.

Figure 2: The architecture of the cloudified application.

5.1 Monitoring

Monitoring of the extraction of a particular document can be done by obtaining status information. It is available at a status-url contained in the JSON object from an HTTP response when requesting the extraction. One can obtain it by issuing a GET request to that URL. The successful response is of the code 200 and enclosed JSON object contains fields named as the cloudlets containing objects with following fields:

- \textit{start-timestamp} - time when cloudlet started processing the document,
- \textit{end-timestamp} - time when cloudlet finished processing the document,
- \textit{error} - contain error if one occurred in this component.

The monitoring response is of the code 404 if there is no information available about such a document.

6 Implementation

Given that we were forced to use the legacy Python software implementing extraction algorithms, two implementation tasks were solved in parallel after designing the application’s architecture - dissecting the Python software so that it could be wrapped into cloudlets, and development of the wrapping cloudlets along with related code built upon the mOSAIC platform.

\footnote{IP is the IP address of the machine where the gateway is running.}
As we said above, the Python software was split into parts so that each logical task (text extraction from PDF, language detection, keyword or citation extraction, etc.) could be executed autonomously. This way, each task usually runs a reasonable amount of time - in order of seconds. Such tasks are more suitable for the mOSAIC platform than longer running ones. Also, it enables us to parallelise those tasks that are not dependent on each other in the future. Each task is implemented by a single Python script with unified input/output interface - they all accept and produce JSON objects.

Source File Processor - that wraps the document to text transformer - is not considered to be an extractor in the code because it has to process a binary file such as PDF that is read from the standard input. It is then different from other extractors and has to be treated specifically in the Java code. Its output, however, is a JSON object that is already accepted by extractors.

A cloudlet that would wrap extractors was created such that it could be parameterised upon creation to execute a particular Python script. This parameter is specified in the cloudlet descriptor, which is a configuration file necessary to start any cloudlet, and is passed along with the cloudlet’s code (the JAR archive in this case).

After finishing works on the Python part, it had to be packaged together with all its numerous dependencies and a specific Python interpreter version - as a TazPkg package of the SliTaz Linux distribution upon which the mOS2 is based. By creating such a package and pushing it into the mOSAIC package repository, we became able to deploy all the necessary software into newly created mOS-based virtual machines. Then we could proceed to integrating both layers of the application.

The integration phase consisted of making Extractors and also Source File Processor spawn a new process executing a particular Python script on the cloudlet’s input. ProcessBuilder and Process Java API classes were used to achieve the execution as the mOSAIC API does not provide its own API for spawning processes under its control yet.

7 Conclusion

We have overviewed our approach to porting a legacy information extraction application to the cloud by using the mOSAIC cloudware along with the designed architecture and some implementation details. To this day, we have finished a main part of the development and the application was successfully tested on the mOSAIC Portable Testbed Cluster3. Next, it is necessary to test it on a real cloud solution to see the performance of the implementation in mOSAIC and also its behaviour when connected to the ReReSearch control system. In case of unsatisfactory results, the application will have to be further improved.

Acknowledgment

The research leading to these results has received funding from the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement number 256910 (mOSAIC Cloud) and by the IT4Innovations Centre of Excellence project, Registration number CZ.1.05/1.1.00/02.0070, supported by Operational Programme “Research and Development for Innovations” funded by Structural Funds of the European Union and the state budget of the Czech Republic.

References


2mOS is an operating system developed to power VM instances running the mOSAIC platform

3PTC is a small-sized cluster for testing and development of mOSAIC applications on local computers


A Client-Centric Identity Management Tool for Small and Medium Enterprises Using Cloud Services

Mircea Boris Vleju
Christian Doppler Laboratory for Client-Centric Cloud Computing (CDCC)
Softwarepark 21, 4232 Hagenberg im Mühlkreis, Austria
E-mail: b.vleju@cdcc.faw.jku.at

Keywords: cloud computing, abstract state machine, identity management, client centric, small and medium enterprises

We propose a client-centric tool to facilitate the adoption of cloud-based services by small and medium enterprises (SMEs). Our approach allows for a quick and seamless integration between the cloud provider’s identity management system and the system used by the client (SME). We introduce the concept of an Identity Management Machine (IdMM), based on abstract state machines, whose purpose is to handle the interaction between the two systems. The IdMM serves as a proxy that performs automatic authentication and authorization for individual users as well as handle identity provisioning and de-provisioning. With the adoption of such a system the client’s directory service remains hidden and the IdMM only offers as much information as necessary, thus providing a privacy enhanced solution.

Povzetek:

1 Introduction

Cloud computing offers a new and cost efficient approach for small and medium enterprises with respect to their IT infrastructure. From a SME’s perspective there are many incentives to adopt cloud based services. Reduced costs, high availability, increased redundancy and flexibility are but a few examples. However, there are also deterrents in adopting cloud services, such as: security and privacy issues [9, 5], vendor lock-in, loss of control [4] and cloud migration [7]. These issues, while pertaining to cloud computing in general, can also be found within the aspects of identity management in cloud computing. With respect to identity management (IdM), the issues above are further compounded by problems related to the increased risk of phishing, single point of failure for identity providers and likability across different domains [1, 3, 6].

2 The Identity Management Machine (IdMM)

To facilitate the adoption of cloud services by SMEs we propose an identity management tool based on abstract state machines [2]. The purpose of this tool is to expedite the adoption or migration to cloud services [8], while providing a secure privacy-enhanced integration between a client¹ and any given cloud provider with respect to all aspects related to identity management. To achieve this goal we introduce an Identity Management Machine (IdMM), described in [10], which serves as a single sign-on tool that automatically authenticates a user to a given cloud service. The machine also serves as an identity provider implementing any given identity authentication or authorization protocol.

The usage of the IdMM system allows a client to keep a private directory service (figure 1). This,

¹We correlate the term client with that of SME. Similarly we define a user as an identity within an SME that will at one point use a cloud service.
Proceedings of the 4th Workshop on Software Services, Bled, 25 October 2012

Figure 1: IdMM Architecture

coupled with the fact that the IdMM only offers as much information as necessary, represents an advantage with respect to privacy. Since the system also serves as a single sign-on tool, the risk of phishing is greatly decreased (a user may not be aware of his credentials for a given service but rather only know the credentials for the IdMM). To further enhance privacy, we make use of obfuscated and partially obfuscated identities [10]. Concerning the issue of cloud migration, the IdMM is designed such that its adoption will require scarce alterations of the client’s directory service [11]. The overall architecture of the IdMM is presented in figure 1.\textsuperscript{2}

The IdMM represents the core of the system. The rules described in [10] make use of abstract functions to achieve the functionalities of the system. These abstract functions must be implemented by one of the submachines shown in figure 1 depending on the type of the function. The specification for the IdMM currently includes the rules for automatic authentication. We make the assumption that the identities exist both on the client’s directory and the cloud system. We will further detail this specification to include identity provisioning and de-provisioning.

The IdMM\textsubscript{Client} submachine is responsible for the interaction between the client’s directory service and the IdMM. Each of the client dependent abstract functions in [10] must be implemented according to the type of the client’s directory. Using this submachine the client’s directory can remain hidden within the local network. The IdMM\textsubscript{Client} is placed on the client’s gateway so that it may be accessible for any external entity. The interaction between the client’s directory and the IdMM is detailed in [11]. The IdMM\textsubscript{User} submachine handles any interaction between the IdMM and a user. The interaction consists of prompting the user for the credentials, displaying error messages and redirecting a user to a service. The IdMM\textsubscript{Provisioning} submachine will be responsible for automatic user provisioning and de-provisioning.

In case of direct client-to-cloud interaction [10] the IdMM\textsubscript{Cloud} submachine handles the authentication, attributes synchronization and de-

\textsuperscript{2}The user may or may not reside within the client’s network. In the latter case the user connects to the IdMM\textsubscript{Client} via the gateway.
authentication for a given cloud service. This is done via a plugin-based system in which all required methods are implemented using the cloud provider’s API or some generic APIs if available. Since the direct client-to-cloud interaction requires the usage of provider-dependent APIs, there must exist an implementation for each authentication service used by the client. Similar to the IdMM\textsubscript{Cloud} submachine, the IdMM\textsubscript{Protocol} submachine implements existing protocols for authentication and authorization (such as OpenID) and acts as an identity provider for the given protocol. The submachine, together with the IdMM\textsubscript{Cloud} submachine, then performs the authentication to the service.

3 An Application Scenario

Let us consider a fictional SME. FooBar GmbH is a startup company which wants to create a cloud-based solution where users can upload, execute and demonstrate examples. To illustrate the problem when using multiple providers let us assume thatFooBar wants to use a combination of services currently available. To manage their documents and allow for collaborative work, they use Google Drive. Even though they have their own email servers installed they want to use Gmail as a webmail service. To develop, test and deploy their solution they use Amazon AWS. Critical files are backed up using Box (box.com). The information about the various accounts and employees is kept in a private ApacheDS server.

FooBar installs the IdMM\textsubscript{Client} on their network making sure this resource is public. The IdMM\textsubscript{Client} is implemented as a web server feeding JSON requests for each of the client-side functions described in \cite{10,11}. To achieve this, the IdMM\textsubscript{Client} submachine implements the IdMM\textsubscript{findService}, IdMM\textsubscript{findUser}, IdMM\textsubscript{getProtocolAttrs}, IdMM\textsubscript{getProtocolAttrs} and IdMM\textsubscript{users} submachines described in \cite{11}. Since most of the services used by FooBar can be accessed via a web-browser the IdMM, IdMM\textsubscript{Cloud} and IdMM\textsubscript{User} submachines will be implemented as a browser plugin.\footnote{For our proof of concept implementation we used an Apache Tomcat server for the IdMM\textsubscript{Client} and a Google Chrome extension for the IdMM. To aid in the implementation of the Chrome extension we also used Google Web Toolkit. All network traffic is encrypted using SSL.}

Each user will have this plugin installed in their browser. The IdMM\textsubscript{User} submachine listens for any user-based events such as opening a URL or closing a tab. When an event is triggered the appropriate actions are taken by the IdMM. In case of new URL, the IdMM attempts to authenticate the user to the service (see \cite{10}) using the IdMM\textsubscript{Cloud} and IdMM\textsubscript{Protocol} submachines. For the examples presented above, it would be necessary for the IdMM to authenticate to three different providers: Google, Amazon and box.com. Since the IdMM\textsubscript{Cloud} submachine synchronizes the user’s attributes upon a successful authentication, any changes made in FooBar’s directory are reflected on the cloud. This synchronization also allows the user to use Gmail as a webmail tool while having a separate email provider, as the POP3 and SMTP attributes are among the attributes synchronized.

4 Further Work

In order to achieve the goals described above, each of the IdMM’s submachines must be further refined. Upon the successful specification of the IdMM\textsubscript{Client}, IdMM\textsubscript{User} and IdMM\textsubscript{Cloud} submachines a proof of concept can be implemented. The inclusion of the IdMM\textsubscript{Protocol} submachine will entail further refinement of the IdMM and its submachines. With a refined IdMM\textsubscript{Protocol} submachine the main purpose of the IdMM is achieved: automatic single sign-on authentication to cloud services. With this in mind, the IdMM\textsubscript{Provisioning} submachine can be specified to allow for automatic user provisioning and de-provisioning which will make the use of obfuscated identities easier. In parallel, the IdMM\textsubscript{Client} specification can be further refined to allow for a better adaptation to the client’s directory service, so that the modification required to adopt the IdMM system will be as few as possible.

With the requirements of the IdMM system fulfilled, we can then focus on enhancing the system by introducing an access management component. This component will aid in both the automatic provisioning (especially for obfuscated identities) as well as in user authentication and authorization transforming the IdMM into an identity access management tool for cloud services.
References


Towards Programmable Infrastructures: the Steps made by Cloud Computing and their Technical Support

Dana Petcu
Institute e-Austria Timișoara and West University of Timișoara
petcu@info.uvt.ro and http://web.info.uvt.ro/~petcu

Keywords: Programmable Infrastructure, Cloud Computing, Platform as a Service

Programmatic access to the devices connected to the Internet is becoming nowadays a reality, involving data center resources, network switches, as well as simple instruments. Cloud computing contribution to the programmability of the e-infrastructures is expected to be be considerable, as promising a step-forwards towards an almost automated and cost-efficient management of the resources. However the vision of fully automated management of the infrastructures is far to be fulfilled.

In this paper we argue that the Platform-as-a-Service (PaaS) has a high potential to improve the state-of-the-art of infrastructure programmability. In this context we analyze the basic requirements for programmability and their fulfillment by the current PaaSs. As case study, the readiness for programming infrastructure of a recent developed open-source PaaS is analyzed.

Povzetek: […]

Extended abstract

Programmable Infrastructure. The notion of Programmable Infrastructure is related to the programmatic access to the devices connected to the Internet. It has been identified until recently with programmable networks [2, 8]. Nowadays it is expected to refer to data center resources (Supercomputers, Clusters, Grids, Clouds, Storage), network switches (Cloud networking), as well as simple gadgets or instruments (Internet-of-Things). The main problem is the fact that still nowadays the manually intervention is still required in several processes involving e-infrastructure settings.

State-of-the-art. The virtualization techniques have provide an initial big step in the emergency of programmable infrastructures [1].

Considerable steps have been made by the middleware handling Grids (see for example [3]). At least three cornerstones can be identified: (1) the emergency of Grid services supported by Globus1, (2) the release of SAGA2 which offers uniform access to different types of distributed computing environments; (3) the release of gEclipse3 integrating services for applications developers and operators of Grids.

The steps expected to be made by Clouds towards the programmable infrastructures are related to: (i) programmatic elasticity in terms of resources; (ii) uniform treatment of infrastructure, software, and networking as programmable services.

However, not all deployment models in Cloud computing are supporting these steps. Software-as-a-Service is hiding completely the e-infrastructure from the user; the programmability of the infrastructure behind it is not necessarily requested. Infrastructure-as-a-Service offers a certain degree of programmability through the web services usually exposed; this is ensuring a considerable advance of the state-of-the-art; however the manual intervention of the application deployers to set the execution environment is still required.

1http://www.globus.org
2http://saga-project.github.com/
3http://www.geclipse.org/
Platform-as-a-Service (PaaS), exposing tools for developing, deploying and executing applications that are consuming infrastructure resources, is expected to allow the highest degree of programmability of the resources compared with the other models. Therefore we are focusing our attention on this model.

**Requirements.** The main requirement to reach the goal of programmability of the infrastructures is to establish an abstract model of the resources that is sufficiently general to catch the characteristics of a large variety of resources as well as to be able to be instantiated as unique resource representative by using the model parameters. Then a proper programming paradigm should be used to express the actions applied to these models. However, to reach in practice the goal of programmability, proper tools should support the resource models and programming paradigm.

Two views can be considered: from an application developer perspective and from an infrastructure provider perspective.

The application developer is interested to control programmatically the resources that are used for a particular applications, so that basic requirements are fulfilled, like one point of access, immediate reaction in case of a resource fault or the control of the number of resources that are used.

The infrastructure provider is interested to reach a certain level of automatization by programmatic self-management, self-tuning, self-configuration, self-diagnosis, and self-healing of the resource provisioning system. These high goals are not currently supported by Platform-as-a-Services.

**Application developer perspective.** The approaches undertaken by different PaaS in what concerns the application support are quite diverse:

1. Deploy application code to specific instances of virtual machines, as in the case of Azure or Elastic Beanstalk;

2. Develop application code according to the platform rules and let the platform deal will with the deployment issues, as in the case of AppEngine or Heroku;

3. Create metadata for an application interpreted by the PaaS at the run-time, as in the case of Force.com or OrangeScape.

The first approach ensures more control for the application developer. The second approach is the most common one. The third approach is the most appealing by allowing a high degree of abstraction and productivity. Therefore we are focusing on this last one.

mOSAIC⁴ is a good example for the third approach. It has introduce some basic concepts that are partially fulfilling the above mentioned requirements of the developer, by proposing an abstract API based on the notion of Cloud components. The description of these components is done in a vendor agnostic manner in order to avoid the current vendor-lock in problem which Cloud computing is facing. Proof-of-concept implementations of the proposed API were done in Java, Python, Erlang and Node.js. Moreover, a proof-of-concept Platform-as-a-Service was build, which has incipient mechanisms for self-organization and which is expected to impact the market of Cloud service provisioning through its openness and deployability in Private and Public Clouds. Details can be found for example in [6, 7] (a complete list of articles about mOSAIC can be found on the project site).

At the level of the PaaS, similar offers are provided by well-known companies like VMWare (CloudFoundry) and Redhat (OpenShift), but with a limited vendor-independence and without a programming support. Another similar IDE for Clouds based Eclipse is reported in [10].

The differentiating particularities of mOSAIC offer are related to one-stop-shop for the application developers, the portability of applications between different infrastructure services, the support for multiple Clouds and so on.

**Provider perspective.** The reduction of the human intervention at the Cloud provider sites is expected to be the result of applying autonomic computing techniques. These are particular suited for the Cloud where rapid elasticity is requested for adaptation to a variable number of

requests or to ensure the high level of reliability despite the potential massive failures.

The Autonomic Cloud, emerging as result of applying autonomic computing techniques to Cloud Computing, represent the highest target of a programmable Cloud, from the provider point of view. Currently several proof-of-the-concept Automatic Cloud are based on known methods from artificial intelligence, like multi-agents systems, genetic algorithms, neural networks, multi-objective optimization heuristics, semantic engines and so on.

Such techniques are also used in the proof-of-the-concept mOSAIC PaaS intending to be a deployable middleware for Cloud service providers, but currently, only in specific parts, like the scheduler and the scaler, or the Cloud service selection mechanism for the deployment phase (based on a multi-agent system, namely Cloud Agency).

Future expectations. Taking mOSAIC offer as reference for the state-of-the-art, a to-do list for the future of programmable Clouds can be done. This list includes at least the further development of a programming paradigm for Cloud resources (extending the one proposed in [5] and the mOSAIC one), the usage of intelligent techniques in the management and governance of Cloud resources (as was done for example for networking resources in [9]), as well as their extension to any type of distributed computing resource.

Acknowledgement

This research is partially supported by the grant FP7-ICT-2009-5-256910 (mOSAIC), and the Romanian grant PN-II-ID-PCE-2011-3-0260 (AMI-CAS).

References


ISES Virtual Energy Lab: an Overview

Matevz Dolenc, Robert Klinč
University of Ljubljana, Faculty of Civil and geodetic engineering, Ljubljana, Slovenia

Peter Katranuschkov
Technical University of Dresden, Dresden, Germany

Keywords: energy-efficient design, life cycle simulation, virtual lab, computing system, civil engineering, ISES

The paper presents early design considerations of the ISES Virtual Energy Lab (VEL) that will through stochastic life-cycle product analysis enable evaluation, simulation and optimisation of energy efficiency of products for built facilities and facility components in variations of real life scenarios before their realisation.

Povzetek: lanek opisuje osnovni nart izvedbe ISES virtualnega energijskega laboratorija, ki bo s stohastinimi analizami celotne ivljenskega cikla omogoil oceno, simulacijo ter optimizacijo energetske uinkovitosti izdelkov za grajene objekte v razlinih uporabnikih scenarijih.

1 Introduction

An important domain of energy and emission reduction is the phase of the planning of new buildings and facilities or such to be refurbished. Another important domain is the design of new products. Products are often not stand-alone objects but part of a host product, such as buildings or facilities. Therefore the most important domain is the design of new products, concurrently considering their own energy behaviour and their interaction with the host product, both in dedicated design cycles. For the energy-efficient design and operation of products the semantic contexts of several different disciplines need to be integrated; e.g. product development, building design and construction, and facility management.

2 Objectives

The objective of the EU project ISES (Intelligent Services for Energy-Efficient Design and Life Cycle Simulation) [2] is to develop ICT building blocks to integrate, complement and empower existing tools for design and facility management to an ISES Virtual Energy Lab (VEL). This will allow evaluating, simulating and optimising the energy efficiency of products for built facilities and facility components in variations of real life scenarios before their realisation, acknowledging the stochastic life-cycle nature. The ISES VEL will be configured as an ontology-controlled SOA system with distributed services, distributed modelling and analysis/simulation tools and distributed data sources. The ISES VEL will allow end-user of the Architecture, Engineering and Construction (AEC) as well as Facility Management (FM) domain to handle the complex analysis of the energy-efficient design of products and take efficient and informed decisions not only of new but in particular of existing buildings and facilities. This is highly important as the mean period to rebuild buildings is about 100 years, for office buildings about 50 years and for factories about 20 years, in comparison with the renewing period (refurbishment, retrofitting) of existing building and facilities is about 10 to 20 years.

3 Functionality of the VEL

The ISES VEL will allow engineers and experienced architects to handle better the complex analysis of the energy-efficient design of products and take efficient and informed decisions.
This comprises three, partially conflicting tasks, namely: (1) the consideration of the stochastic nature of the energy performance and consumption profiles of the new product life-cycle, (2) the balanced design of the new products (components), their functionality and behaviour for the various possible life-cycle demands, and (3) the balanced interaction of the product component with the host product, i.e. the context system in which the new product is applied (or built in).

For the design of the new product (component), this means that several characteristic host products have to be analysed, which results in the design of several new variant products. However, for the use of the new product, the best-fitting variants of the product have to be selected and an optimal integration in the host product, i.e. the context system has to be realized. This requires several simulations with feedback cycles in order to reach an optimally balanced solution (see Fig. 1).

The design feedback cycles occur at different phases of design and production, whereby the first feedback cycle, i.e. the simulation feedback, is needed as sub-cycle in the two other feedback cycles. This means that the number of resulting feedback cycles is multiplicative. The ISES VEL will automate the configuration, management and evaluation of these models for the various needed simulations by means of a set of innovative services and tools including navigation and inspection tools for the engineer, allowing him to involve only on high-level decision making tasks.

4 Structuring of the VEL

The ISES VEL design is structured into four sequential tiers and two supporting tiers (see Fig. 1). The first tier includes the domain modelling and input tier to the ISES VEL. It also includes tools and databases for (1) the modelling of the new product, including a model schema extender to STEP, (2) the modelling of the host product itself, (3) the modelling of energy profiles and (4) patterns of energy consumption. The second tier is the multi-model combiner that combines the different domain models into one investigation model is complemented with a simulation configurator, which has the task to configure the simulation models automatically according to a few general input directives by the engineer provided via an graphical user interface. The third tier provides access to HPC/HTC tools and services for analyses of configured simulation models. Implemented ISES VEL services support scalability through use of different cloud services providers. The forth tier includes tools and services for the evaluation of multi-models, including prioritisation of the results and four supporting services, namely multi-model filter, navigator, evaluator and manager, providing easy user access with proactive support for requesting and selecting simulations to be compared.

The overall workflow is organized in two feedback cycles, one for handling the stochastic life-cycle nature of climatic and usage conditions and one for the optimization of the design - to lay out the most beneficial variants for the new product and to select, configure and assemble the right variant products in the host product or even redesign the host product partially or as a whole.

5 System Architecture of the VEL

The ISES VEL system architecture (see Fig. 2) builds upon previous similar efforts [1]. Based on the SOA principles it is logically structured into five layers: (1) numerical, (2) service, (3) core
functionality, (4) core data, and (5) auxiliary data layers. The new ICT building blocks to be developed are drawn with dashed line. The ISES VEL will be based on available commercial modellers and information management systems for design (CAD) and for operation management (FM) and their related standardised data structures for the Building Information Models (BIM) [3], i.e. the product model standards IFC (ISO/PAS 16739) and STEP (ISO 10303), and the Building Automation System Models (BAS) industry standards BACnet¹, LON² and the newer European standard KNX³.

Figure 2: System Architecture of the ISES Virtual Energy Lab.

There are several drawbacks to available numerical analysis tools for energy related calculations: (1) tools directly connected with usually one of the two existing functional layer tools, often only with special proprietary interfaces, (2) model configuration and modification have to be carried out manually for each particular model individually, (3) representations of numerical results are mostly done via individual proprietary graphical user interfaces, (4) the analysis work is tool-oriented and not appropriate for many design simulation feedback cycles, etc. To overcome the above mentioned drawbacks the ISES VEL will include a high-level model and system ontology, that will enable automation of various tasks. Different modellers and managers (CAD/FM) will be complemented with an intelligent access controller, which will provides intelligent access to the various tools, services, databases and cloud services.

6 Conclusion

The paper provides motivation and an overview of functional, structural and architectural system design considerations of the ISES Virtual Energy Lab that is being developed within the 7th FP EU project Intelligent Services for Energy-Efficient Design and Life Cycle Simulation (ISES).

It must be emphasised that the presented ISES VEL is still in early stage of development and is currently only available to ISES project partners. When ISES project nears its final review in 2014 the ISES VEL will be available to public at http://ises.eu-project.info/vel.

Acknowledgement

The presented research has received funding from the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement number 288819 (ISES) as well as by the industrial partners. The contribution of the founding agency and all project partners is gratefully acknowledged.

References


¹Building Automation and Control Networks
²Local Operating Network
³KNX is a standardised (EN 50090, ISO/IEC 14543), OSI-based network communications protocol for intelligent buildings.
Migrating a Simulation Framework from the Cloud to the Sky

Gabor Ferschl, Tamas Mahr
gferschl@aitia.ai and tmahr@aitia.ai

Keywords: cloud, application, mOSAIC, simulation

1 Abstract

The existing application by AITIA, the Model Exploration Service (MES) [1] is an online service to run agent-based simulations in the cloud. Researchers, mainly from the field of social simulations can use AITIA’s standalone tool Model Exploration ModulE (MEME) [2] to explore their agent-based models locally, and if they register on the Model Exploration Web Portal (MEP), they can use cloud resources from Amazon to run larger parameter sweeps of their models.

The Intelligent Applications and Web Services department of AITIA, the developer of MES has been involved in the social simulation field for at least ten years, and is devoted to develop tools to make creating and analysing agent-based models easier for researchers of the field.

mOSAIC [3] is an ongoing project under the 7th Framework Programme of the European Commission, which promises to create a platform that provides vendor-agnostic API and tools for the developers, maintainers and users of cloud-based applications.

The existing application (MES) runs on Amazon, and it uses Amazon specific calls to manage the virtual infrastructure. Porting the application to mOSAIC does not simply mean to change such calls from the Amazon API to the mOSAIC API. The internal architecture of the application is also changed to adopt mOSAIC design principles (the cloudlet model) to facilitate scalability, portability and autonomous reconfiguration. Using the mOSAIC Cloud Agency services the application can be deployed on a cloud infrastructure that is best suited for the task and that is the case with the simulation tasks as well. Also, by implementing the internal workflow of the application with cloudlets, it becomes possible to scale up and distribute the application with higher user demand.

Developing the application with mOSAIC frees the developers from vendor lock-in, and makes it easier to develop new cloud-aware features. By providing support for Service Level Agreements, negotiation and monitoring, the platform offers these tools for the cloud provider of our choice, which otherwise developers had to learn or create themselves.

In this paper, we present the progression of the development process, with details of the used platform components, the newly developed components, the design decisions and the reasons behind them with special aspects of the migration to the mOSAIC platform. Our intention is to show the reader the advantages of developing cloud-based applications in mOSAIC, and a possible way to migrate an existing application to mOSAIC.

The figure on the next page shows the architecture of the proposed application, with components still under development with dashed borders and lines.

In the figure, the types of components are distinguished by colors:

The green components are the ones that AITIA developed, and conform to the mOSAIC cloudlet and component requirements.

The orange components are mOSAIC components or resources that are offered by the platform.

The purple components are either developed by AITIA or a 3rd party and are used by other components.

The end user interacts with the application through MEME and the web portal. On the por-
tal, the user can buy simulation credits, which can be used to run the experiments created with MEME. The chain of cloudlets works as a pipeline, where each item does its job, and passes the remaining work further. The first step in the chain is the Model Exploration Cloudlet, which authenticates the user and handles the user’s credits. It also requests virtual machines from the Cloud Agency. Then it passes the job to the Simulation Job Executor, which divides the job into small tasks, and passes them to the Simulation Task Runner cloudlets that are running on the previously acquired virtual machines.

Each simulation task creates a result, which is written to an existing Hadoop Distributed File System (HDFS) [4] with the help of the mOSAIC Distributed File System Driver. The Result Merger Cloudlet merges these results, zips them and stores them for the user to download.

The business model of the application is a typical cloud approach: use only as many resources as necessary. The provided service can be first accessed through a web portal, where users can create accounts and pre-pay their experiment credits. This web portal should be always up-and-working and it needs a database to store its data. That is all that is needed by AITIA’s service, when no users are running experiments.

If a user wants to run experiments, only then does the application need more computational power. It is up to the user to decide how many virtual machines is necessary, so the application just blindly acquires the resources and runs the job. When finished, the resources are not needed anymore, so they are deleted. For security reasons, the application does not reuse virtual machines for a different experiment job.

AITIA also wants to facilitate the multi-cloud aspect of mOSAIC, when searching for virtual machines for the experiments to run on, the application presents the possibilities returned by the Cloud Agency to the users, who can choose the best-fitting option. Here the user has the option to choose a provider and infrastructure that can successfully run the simulation, and by selecting the number of virtual machines, the user can vary the speed vs. cost of the simulation experiment.

By the end of the development of the Model Exploration Service, the application will have improved in aspects related to cloud-awareness. By using mOSAIC, the application is no longer tied to Amazon, it can be deployed on every available and suitable cloud provider. By avoiding vendor lock-in, the application can be moved across ‘cloud borders’, or the application can be multi-cloud in the sense that certain parts run on different provider infrastructure.

The development process also benefited from the cloudlet approach, which forced a programming style on developers that is highly recommended for developing cloud applications. By adopting this style, the application becomes more flexible and scalable, because of the loosely coupled stateless components. In the end it is eas-
ier to quantitatively measure how the important components of the application perform, and they can be reconfigured based on actual needs.

2 References

[1] Model Exploration Service:
    http://modelexploration.aitia.ai/

[2] Model Exploration ModulE:
    http://mass.aitia.ai/intro/meme

[3] mOSAIC Open Source API and Platform for Multiple Clouds:
    http://www.mosaic-cloud.eu/

[4] Apache Hadoop Distributed File System r1.0.1:
    http://hadoop.apache.org/docs/r1.0.1/hdfs_user_guide.html