Energy-Efficient Datacenter

A reinforcement learning based approach

# SLA Negotiation

# **Introduction**

## Motivation

Reducing energy consumption and using biodegradable materials has become the main purpose in all the existent industry fields. In the last few decades, researchers have been focusing on improving performance in all possible fields, neglecting overall efficiency considering energy consumption. With the growth of impact of industry and pollution on the environment, the direction of evolution has changed from a performance directed one to an efficiency based approach.

Green computing is a new trend in computer science, encouraging the development of environmentally sustainable computing. Basically, it regards the study and practice of using computing resources efficiently. Studies have been focused on designing computing resources which maximize the energy efficiency during product’s lifetime, are biodegradable and don’t use hazardous materials. Extensive research is being undertaken in design of algorithms and systems for environmentally-friendly computer technologies and efficient use of computers.

On 2007, the report to congress on server and data energy efficiency [1] gave a definite warning in what regards the energy consumption of servers and data centers. From 2000 to 2006 the energy consumption of datacenters has doubled in United States, rising to about 61 billion kilowatt-hours (kWh) and being similar to the amount of electricity consumed by approximately 5.8 million average U.S. households. This is why a great part of the effort of painting the computing industry in green has to be directed towards decreasing datacenters’ and servers’ overall energy consumption, including the cooling infrastructure that supports IT equipment in datacenters.

Reducing the energy consumption in datacenters can be approached by improving the efficiency of computer resources or by software solutions of intelligently using those resources. Several solutions have been approached for decreasing datacenters’ overall energy consumption. These can be categorized in two main directions: hardware solutions and software solutions. The hardware solutions intend to decrease energy consumption by designing low power components thereby reducing the entire’s machine energy consumption. The software solutions are at their beginnings but there have been researches focused on decreasing power consumption depending on the existing load or on distributing efficiently tasks in the datacenter in order to have optimal power consumption.

## Objectives and contributions

The objectives of this thesis project are the following:

* A **description of the datacenter context** composed from servers which have certain resources as CPU, Storage, Memory and this **context’s mapping on the <R,A,P>**(Resources, Actors, Policies) model presented in [2].

The datacenter is composed of a number of servers with different characteristics for different resources like memory, storage and CPU. These are being mapped on a context model composed of Resources, Actors and Policies. The role of Resource is being played by any server in the datacenter and the task plays the Actor role. Considering the previously mentioned elements, policies are being defined using SWRL describing the situations in which the GPI ( Green Performance Indicators) for the datacenter, and the KPI (Key Performance Indicators) for the task are being respected.

* **Developing a self-\* algorithm** with the purpose of optimally distributing the workload, while using as less energy as possible.

Using the above context, a self-adapting algorithm is being designed, for the getting the described model to the lowest possible energy consumption. This algorithm involves a reinforcement learning approach by adapting the self-healing algorithm presented in [3], and adding few improvements. The self-healing algorithm is also being used for controlling the temperature and humidity in the datacenter room.

* **Negotiating the Service Level Agreement** of the task for the case in which the task doesn’t fit in any server on the datacenter, to modify existing policies for the datacenter and make it possible for the tasks to be properly distributed.

Because the user is giving us a requested range for the resources of a server, there might be a situation in which there is no place for that task due to GPI Policies. In this case, a negotiation should be made in order to have a higher maximum accepted value and the task to fit. There is also the case where the task could get more than the minimum requested value, case in which a redistribution of available resources among existing tasks on that server will be made.

## Publications

* + 1. **A Reinforcement Learning based Self-healing Algorithm for Managing Context Adaptation** [3]

This paper presents a reinforcement learning approach for finding the optimum sequence of actions for healing a broken context. For this algorithm to be able to function weights are added to resources and policies and concepts like entropy and inter-independent resources group are being introduced.

* + 1. **An Autonomic Algorithm for Energy Efficiency in Service Centers** [4]

This paper presents an improvement of the previous self-healing algorithm, used for the self-adaption of the energy-efficient datacenter. The self-adapting approach features a closed feedback loop with four MAPE phases: Monitoring, Analysis, Planning and Execution.

## Overview of the report

In the next parts of this report, I will present a theoretical background needed in order to present this project, with available software and technologies needed for implementing it on a real datacenter. The next chapter, chapter 3 gives a view on the existing work in the domain. The fourth chapter describes the algorithms and models and from here on the architecture, the design and implementation is presented ending with testing and conclusion.

# Theoretical Background

## Context-Aware Computing

Awareness is one of the main problems arising in nowadays computing systems. Building systems which are aware about what happens and about their awareness becomes crucial in hospitals, modern buildings and even personal houses. To handle contexts in all of these environments the system must have sensors for each monitored element and a way to control that element, through different actuators. In order for the system to know when the context is not in the state desired by the user, policies are being described for all elements composing the context. For example, in a smart laboratory, we need to know that if no one is inside the light should be off. Generally the system takes the sensor information and enforces actions on actuators through web services, this way having a low coupled architecture.

## Energy-Aware Computing

Energy-awareness is a subset of context awareness, improving the systems towards green use of the computing infrastructure. The green use of a datacenter implies reducing the energy consumption of computers and other information systems as well as using them in an environmentally sound manner. For the real servers and datacenters to have this behavior there are several options like programmatically assign loads to resources or programmatically assigning loads to servers. The load assignment for datacenters is easily accomplished through virtualization. By this, a system administrator could combine several physical systems into virtual machines on one single, powerful system, thereby unplugging the original hardware and reducing power and cooling consumption.

### Virtualization

According to Wikipedia, the term "virtualization" was coined in the 1960s, to refer to a virtual machine (sometimes called pseudo machine), a term which itself dates from the experimental IBM M44/44X system [5].Virtualization is a new technique reproducing computer hardware through software. In a typical server environment there exist different servers each hosting only one task, for example a web server and a file server. By using server virtualization, both the previously mentioned servers will be running on the same machine, one independently of the other, therefore reducing the costs and energy consumption of a second machine. The center of the entire virtualization process is the virtual machine, it being defined as a software implementation of a machine that executes programs like a physical machine. There are two types of virtual machines: system virtual machines and process virtual machines. From the process virtual machines, the JVM (Java Virtual Machine) and the .NET Framework, which runs on a VM called the Common Language Runtime, are the most known one. Process virtual machines run as a single application inside the operating system, and support one single process. They are created when that process is started, and destroyed when it exits. We are interested with the system virtual machines which allow the sharing of the underlying physical machine resources between different virtual machines, each running its own operating system [5]. There are several advantages coming with the use of system virtual machines, like the fact that multiple operating system environments can run on the same computer, in strong isolation from each other or that the virtual machine can provide an instruction set architecture different from that of the real virtual machine. The software layer providing virtualization for system virtual machines is called a virtual machine monitor or hypervisor. Due to the fact that it is an important part of datacenter administration having virtual machines in the role of tasks, hypervisor description will be detailed in the followings.

#### Hypervisors

The hypervisor, also called Virtual Machine Monitor (VMM) provides the guest operating system a virtual platform and monitors their execution. Despite the fact that the virtual machines can commonly used resources, the failure of one virtual machine won’t produce the failure of all the other virtual servers running on that machine. The isolation ensured by the hypervisor is one of the main features of virtualization, which brings it to the top of technologies to be used in modern datacenters. The hypervisors are split into two categories: software and bare-metal. Software hypervisors need a host operating system to run on and have lower I/O performance due to the overhead resulting from the hypervisor-host OS communication. Bare-metal hypervisors received their name from the fact that they run on “bare metal”, needing no host operating system. Actually the server must be formatted before this hypervisors are installed. This close connection to the underlying hardware brings better I/O performance and is also faster due to the removal of the layer introduced by the operating system.

Microsoft Hyper-V is the hypervisor which is the most present one in the data centers all over the world. It can be run on an x64 version of Windows Server 2008, the R2 version having the live migration feature enabled. In order for Hyper-V role to be enabled for a windows server, the processor needs to have hardware assisted virtualization. This is available for processors that include a virtualization option (Intel VT or AMD Virtualization). Live migration is supported with the use of cluster shared volumes (CSV). This feature is extremely important for enabling the movement of a virtual machine from a server to another, from efficiency reasons. Live migration of virtual machines from a server to another is done automatically for situations in which the node (the Hyper-V server) fails. In this situation, each virtual machine running on the failed node may migrate to other live nodes independently of other virtual machines. Due to the fact that this diploma project is undertaken within a larger group with different missions, we have chosen Hyper-V as a hypervisor, it providing both a high-level and a low level API. In there are presented on short two other hypervisors, one which offers full virtualization and one offering paravirtualization, together with a description of paravirtualization and comparison to full virtualization.

VMWare ESXi is a “bare metal” hypervisor, meaning that it doesn’t need to run on top of other operating systems. This implies a lower overhead and a better control and granularity for allocating resources (CPU time, disk bandwidth, etc.) and a considerable increase of security. VMWare ESX and VMWare ESXi offers advanced resource management capabilities to improve performance and increases consolidation ratios.

Both Hyper-V and VMWare ESXi offer a full virtualization approach which allows datacenters to run an unmodified guest operating system, thus maintaining the existing investments in operating systems and applications and providing a nondisruptive migration to virtualized environments. On the other hand, the paravirtualization approach modifies the guest operating system to eliminate the need for binary translation. Therefore it offers potential performance advantages for certain workloads but requires using specially modified operating system kernels [6].

The Xen open source project was designed initially to support paravirtualized operating systems. While it is possible to modify open source operating systems, such as Linux and OpenBSD, it is not possible to modify “closed” source operating systems such as Microsoft Windows. It is also not practical to modify older versions of open source operating systems that are already in use. As it turns out, Microsoft Windows is the most widely deployed operating system in enterprise datacenters. For such unmodified guest operating systems, a virtualization hypervisor must either adopt the full virtualization approach or rely on hardware virtualization in the processor architecture [7].

#### Virtual appliances

A software appliance is a full application stack containing the operating system, the application software and any required dependencies, and the configuration and data files required to operate. Everything is preinstalled, preintegrated and ready to run. Software appliances come in the form of a file which can be a virtual machine image, an ISO, a USB key image, or an Amazon EC2 AMI. They run a JeOS(Just Enough Operating System) is a customized operating system that precisely fits the needs of a particular application (Ubuntu JeOS, OEL JeOS, SUSE JeOS, OpenSolaris JeOS, OpenSolaris JeOS, Orange JeOS, and Windows Server Core).The virtual appliances, a sub-class of software appliances, add to the advantages of software appliances the benefits of virtualization.

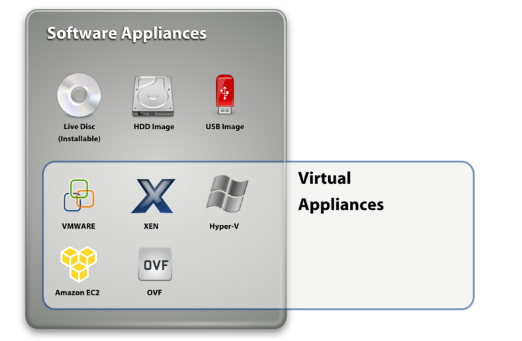


Figure 2.1: Software Appliances Taxonomy- © Novell Inc

The biggest advantage of virtual appliances is that there is a considerable energy economy because we don’t have any more tasks waiting one for another and redirection of conflicting tasks, being an alternative to massive changes and increased complexity on the software stack. Virtual appliances allow for rapid movement of virtual appliances between physical execution environments, provide an improved isolation between several appliances sunning on the same server and improve fault tolerance .For tasks that need to be sent to a data center and run on a server, a virtual appliance can be created with all the needed applications and only the needed part of the operating system and sent to be run secure and isolated from other tasks.

In terms of solutions to energy efficient datacenters, software appliances are a feasible alternative to virtual machines, considering that by having all the needed software and no more than that on top of a lightweight JeOS the time for programmatically creating virtual machines with different characteristics is reduced to 0 and the dimension of the virtual machine is reduced. Not all the hypervisors are able yet to host virtual appliances, therefore this is a subject for implementation after virtual appliances leave the research state and people get acquainted with them.

### Hyper-V WMI Provider

For monitoring virtual machines in a network of servers running Hyper-V, the Hyper-V WMI Provider is being used. It is a high level API, giving information about virtual machines and their status. It enables developers and scripters to build custom tools, utilities and enhancements for the virtualization platform, managing all aspects of the Hyper-V Services [8]. Most functions in this API are available in Basic, PowerShell, C# and C++, therefore the limitations from the programming language point are not too high.

## Autonomic Features

In a manifesto in 2001, IBM invites the world, their customers, competitors and colleagues to accept the Grand Challenge of building and deploying computing systems that regulate themselves and remove complexity from the lives of administrators and users. On short, they consider that the new Grand Challenge in computing industry is the overgrowing software complexity both in terms of management and in terms of maintenance. They believe that the growing complexity of the I/T infrastructure threatens to undetermine the very benefits that information technology aims to provide. Human intervention and administration to manage software complexity is starting to be overwhelmed. It is estimated that at current rates of expansion, there will not be enough skilled IT people to keep the world’s computing systems running.

Considering the fact that “in the evolution of humans and human society, automation has always been the foundation for progress” [9], IBM states that it’s time to design and build computing systems capable of running themselves, adjusting to varying circumstances and preparing their resources to handle most efficiently the workloads we put upon them. These autonomic systems must anticipate the needs and allow users to concentrate on what they want to accomplish rather than figuring how to rig the computer systems to get them there [10].

They give four directions of approach in terms of self-management, described bellow: self-configuration, self-healing, self-optimization and self-protection. Systems having all these capabilities are also called CHOP systems or self-\* systems. This thesis presents a self-healing approach for the datacenter room, and a self-optimizing approach for having a datacenter which consumes the optimum amount of energy.

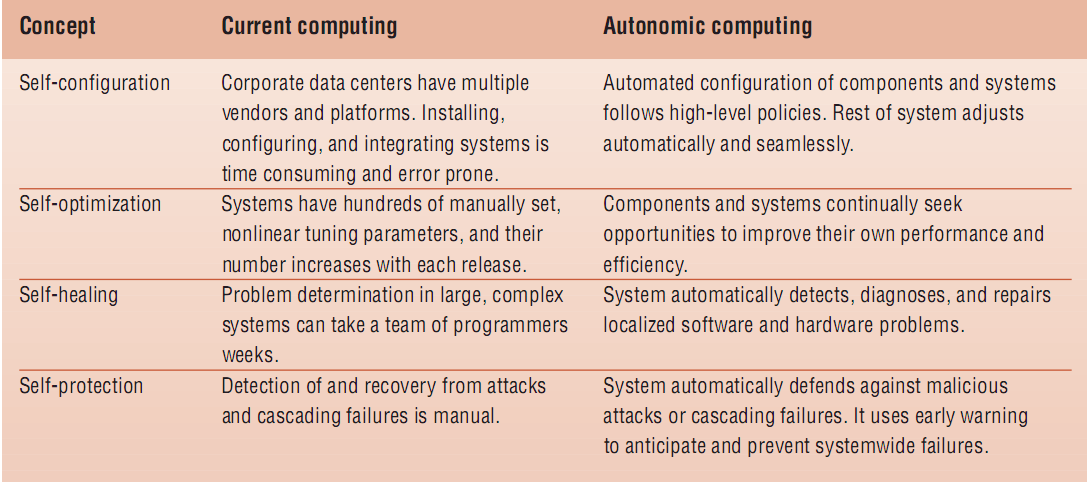


Figure 2.2: Four aspects of self-management © IBM

## Context Model

The most relevant context modeling approaches are presented in [11]. They are classified by the scheme of the data structures which are used to extract information: key-value models, markup scheme models, graphical models, object oriented models, logic based models and ontology based models. Out of these, the logic based model and ontology based model are the ones that store information on which it can easily be reasoned upon. The logic based model is a very formal model, defining conditions on which concluding expression or facts may be derived from a set of other expression or facts. Ontologies are of the most promising instruments to specify concepts and interrelations [11]. They are suitable both for projecting parts of the information used in our everyday life onto a data structure utilizable by computers, and for reasoning in that information with the help of a rule language layered on top of RDF. This is why the chosen context model for our context-aware system is an ontology model.!!!!!!!! OWL , RDF :D prescurtari

The ancient Greek philosopher Parmenides of Elea was among the first to propose an ontological characterization of the fundamental nature of reality. An ontology together with a set of individual instances of classes constitutes a knowledge base. In reality, there is a fine line where the ontology ends and the knowledge base begins.For creating and modifying the ontology, the Protégé ontology editor and knowledge-base framework is being used.

## Mobile Agents – JADE

Java Agent DEvelopment(JADE) Framework is a mobile agents framework which simplifies the implementation of multi-agent systems. A mobile agent is a software agent with features like autonomy, social ability, learning and mobility. Mobile agents are extremely important in context-aware applications, each agent taking a responsibility and acting independently of the other agents. They can communicate and can be distributed on different machines, or can move even at run-time from one machine to another and resume the execution on the new machine.

# Related Work

Self-\* and context-aware systems have become leading domains in terms of research. The self-\* initiative has been started by IBM, in the manifesto from 2001, stating that developing autonomic systems is the new challenge in IT [9]. For a system to be self-\* it has to be aware of the context in which it has to function. Therefore, for creating a self-adapting datacenter, it has to be aware of the surrounding context (waiting tasks, temperature, and humidity) and of the energy context (servers’ power consumption, resources used by each deployed task). It will also need to have policies for it to know how to adapt in new situations. On the other hand, considering that the user will tell the datacenter how many resources he would need, and that the user tends to ask too much therefore the datacenter can be in a situation where it can’t fit the task on none of the servers, a negotiation process is needed for SLA. In the following subchapters, related work is presented for all the above mentioned problems.

## Context Aware Systems

Ubiquitous Computing is a term coined by Mark Weiser around 1988, and refers to the seamless integration of devices into the users’ everyday life. The appliances should be hidden into the background, to make the user aware of his tasks and priorities rather than computing devices and technical issues. One field of pervasive of ubiquitous computing is the context-aware (sentient) computing. Context aware systems operate without necessary user intervention, by taking environmental context into account. Smart rooms, which can be seen as intense ubiquitous computing environments, are a step toward Weiser’s [12] vision. Currently, there are already some smart rooms functioning at different organizations, like MIT’s Intellingent Room[13] or Stanford’s iRoom Project [14].

What seems to be a problem in context-aware computing is the privacy of people whose activity is monitored by pervasive computing systems and on whose behalf actions are being undertaken. This is why, in [15] Hong and Landay describe an architecture for privacy sensitive ubiquitous computing which addresses the issue generated by people’s concerns about the strong potential for abuse over a potential lack of control. Confab is a toolkit for facilitating the development of privacy-sensitive ubiquitous computing applications. Confab provides a framework for ubiquitous computing applications, where information is collected and processed as much is possible on the end-user’s computer. In this way, the user can specify how much of the information on his computer is available, and for what purposes. Confab therefore gives “an extendable design that provides a software architecture support for building privacy-sensitive ubicomp applications that are optimistic, pessimistic, and mixed-initiative” [15].

Currently most attempts to use context-awareness within ubiquitous computing have been focusing on the physical elements of the environment, or the user’s device. In contrast with that, a new context-awareness direction towards capturing the cognitive elements of a user’s context is explored by Prekop and Burnett with a conceptual model of Activity Centric Context [16]. The focus of the activity-centric view of the context is on the information surrounding the performance of an activity undertaken by an agent. The activity-centric view has, as its name states, the agents and activities in the centre of the entire perspective on the context. By monitoring and storing agent’s activities, each of the activity belonging to a higher lever activity together with its context, Prekop and Burnett’s paper provides a model for supporting complex context-aware applications after capturing agent’s behavior.

For supporting context-aware systems different ontologies and languages have been described, with the purpose of easing the pervasive frameworks.

The COBRA-ONT ontology for COntext BRoker Architecture (COBRA) is “a collection of ontologies for describing places, agents, events and their associated properties in an intelligent meeting room domain” [17]. The design of COBRA addresses important issues like supporting resource-limited mobile computing and addressing concerns for user privacy, and contains four essential components: a context knowledge base, a context reasoning engine, a context acquisition module and a policy management module.

In [18] Kagal et al. present a policy language specialized for pervasive systems. Rei is a policy language based on deontic concepts, which can be used to describe several kinds of policies. With its help, security policies can be described, restricting access to resources of organizations. It can be used to create actions on resources and describe restrictions and availabilities for users in an organization or defining conversation policies very important in autonomous environments. All of the above possibilities makes Rei a versatile and expressive policy language especially for context-aware computing systems, it being a branch of pervasive computing.

## Energy Aware Systems

## Self-\* Systems

## SLA Negotiation

# References

[1] **U.S. Environmental Protection Agency, ENERGY STAR Program**, “Report to Congress on Server and Data Center Energy Efficiency”, Public Law 109-431, 2 august 2007

[2] **Tudor Cioara, Ionut Anghel, Ioan Salomie**

[3] **Ionut Anghel, Tudor Cioara, Ioan Salomie, Mihaela Dinsoreanu, *Georgiana Copil*, Daniel Moldovan**, “A Reinforcement Learning based Self-healing Algorithm for Managing Context Adaptation”, 8th Annual IEEE Int. Conf. on Perv. Comp. and Communications, 2010.

[4] **Tudor Cioara, Ionut Anghel, Ioan Salomie, Mihaela Dinsoreanu, *Georgiana Copil*, Daniel Moldovan**, “A Self-Adapting Algorithm for Context Adaptation”, !!!! roedunet?

[5]**Wikipedia**, Web Site: [www.en.wikipedia.org](http://www.en.wikipedia.org)

[6] **James Carter**, “**A look inside IBM's green data center research**”, Proceedings of the 14th ACM/IEEE international symposium on Low power electronics and design.

[7] **VMWare Team**, “A performance Comparison of Hypervisors”, Jan 31, 2007

[8] **Hyper-V WMI Provider**, Web Site: <http://msdn.microsoft.com/en-us/library/cc136992%28v=VS.85%29.aspx>

[9] **IBM manifesto,** Web Site:<http://www.research.ibm.com/autonomic/manifesto/>

[10] **Jeffery Kephart, David Chess**,”The vision of autonomic computing”,Computer Magazine, IEEE, 2003

[11] **Thomas Strang, Claudia Linnhoff-Popien**, “A context modeling survey”, Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England

[12] **M. Weiser**, “The Computer for the 21st Century,” in Scientific American, 1991.

[13] **MIT's Intelligent Room**, Web Site: [www.ai.mit.edu/projects/iroom](http://www.ai.mit.edu/projects/iroom)

[14] **Stanford’s iRoom project**, Web Site: <http://graphics.stanford.edu/projects/iwork>

[15] **Jason I. Hong, James A. Landay**, “An Architecture for Privacy-Sensitivity Ubiquitous Computing”, International Conference On Mobile Systems, Applications And Services,  
Proceedings of the 2nd international conference on Mobile systems, applications, and services

[16] **Paul Prekop, Mark Burnett,** “Activities, Context and Ubiquitous Computing”, Computer Communications, Volume 26, Number 11, 1 July 2003 , pp. 1168-1176(9)

[17] **Harry Chen, Tim Finin, Anupam Joshi,** “An Ontology for Context-Aware Pervasive Computing Environments”, The Knowledge Engineering Review, Volume 18 ,  Issue 3  (September 2003)

[18] **Lalana Kagal, Tim Finin, Anupam Joshi,** “A Policy Language for a Pervasive Computing Environment”, Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks