Teseus (2015)
Assembling a new computing system, based on grid configurations that brings efficiency improvements

Simone Petersen, Quim Marsà, Kuan Wei
Export and business, electronic and electric, telecommunication

Abstract—TESEUS is being carried out within the umbrella of the ongoing *Urban Node*[1] study and focuses on the development of a computational system capable of handling a huge amount of data received from urban nodes in a smart city.

There are already existing computer systems with computational power capable of handling the amount of data a smart city requires. Although these computers may provide all the power required to maintain and manage a smart city, they can always be made more efficient. The lack of efficiency comes from the inability to perform in a scalable way. A computer is not always required to work with its full power. The way supercomputers are designed is rather inefficient when managing lesser amounts of data than expected. The Teseus project objectives focus mainly on efficiency and the facilities a scalable system offers.

In order to create a new computational system, research and possibilities for both hardware and software that offers scalability and its management have been carried out and explored. The hardware part of the system is a low consumption single board computer and the software is based on a grid configuration.

Keywords—data transfer and acquisition, computational system efficiency, scalability, Urban node, grid, single board computers.

I. INTRODUCTION

The Teseus project is a collaboration between national and international engineering students of the EPS (European Project Semester) of the university EPSEVG (Escola Politècnica Superior d'Enginyeria de Vilanova i la Geltrú) and Neàpolis with supervision of professors from EPSEVG. Since the content of the project was closely into an informatics framework, two extra students that are currently studying computer science at the University of Vilanova were included from the beginning. The purpose of the project is to create a more efficient, cheaper and a scalable single computational system consisting of single-board computers in urban nodes. The project is named Teseus and it is an attempt to be created to function as a city brain and process city data more efficiently in smart cities.

II. FRAMEWORK AND OBJECTIVES

Teseus is a part of the urban node project. The urban node project created at Neàpolis consists of four main parts: Teseus, Hermes, Zeus and Medusa. Hermes is the improvement of mobile communications in cities, Zeus is the energy generation, Medusa is distributed sensors around cities for big data and the Teseus project is distributed computing into different urban nodes forming a grid.

This project is created to assist companies who are interested in implementing efficient computer systems which works with high data transfers such as smart cities.

The objectives for Teseus was provided by Neàpolis[2] and aim of the Teseus project is to create a distributed computer system consisting of four to five SBC[3] (single board computer) forming a grid but appears as one system. The goals of the project include research and investigating possible SBC solutions, considering energy, computational power and economic efficiency. In order to connect the SBCs to function as a single system the best software solution to form a grid, must be found and tested. In addition, the system must be able to manage a large amount of data very fast and therefore a consideration of the use of either switches or routers to connect the SBC are made. The system will be tested on a usual commercial application and integrated on a different platform.

III. EFFICIENCY AND SCALABILITY - ARM PROCESSOR

One of the first acknowledgments was that in most current computer systems that use an x86 processor

(CISC architecture) a lot of energy is wasted since the whole power is not always required. The internal structure of the CPU can tell apart when part of the power is being wasted but its ability to split and use just part of the capacity to handle a simple task is still experimental.
This processor minimizes the number of instructions per program but on the other hand takes multiple clock cycles to execute the entire program.
Nowadays new types of processor such as ARM are focusing on efficiency rather than great performances. This processor minimizes the clock cycle per instruction but has more number of instructions per program.
Therefore it requires less transistors so takes up less hardware space and for that reason the power consumption is less. [4,5]
This little space they take and the fact that ARM emphasizes in efficiency, makes it ideal for being used in a grid. This way this interconnected array of more efficient but less powerful computers can sum up the power of a powerful x86 computer but with the peculiarity that it is scalable. Scalability reduces even more the costs of the new computer configuration by turning off those computers that are not necessary at some point. Or on the other hand, it also offers the possibility to add up more computers in case more power is needed for new application that takes up a lot of data or in case the city grows.

IV. ENERGY SAVING CALCULATIONS

The average power consumption of an ARM processor is around 3W and the SBC of this project, the Raspberry Pi 2 card consumes 2,25W. Six of them equal a standard x86 computer in computational power. A standard x86 consumes 45W and six RBPi2 consumes 13,5W, which is almost three times less consumption of energy.

For a system that requires all the computational power 24 hours a day, the calculation of the amount of watts used per month is the following:

-x86 computer: 0.045kWh · 24h · 365 days= 394.2kWh per year.

-6 RBPi2 (computational power equivalent to one x86)

0.0135kW ·24h · 365 days= 118.26 kWh per year.

Below, we now see how much money we save per month if the cost per KWh is 0.12€.[6]

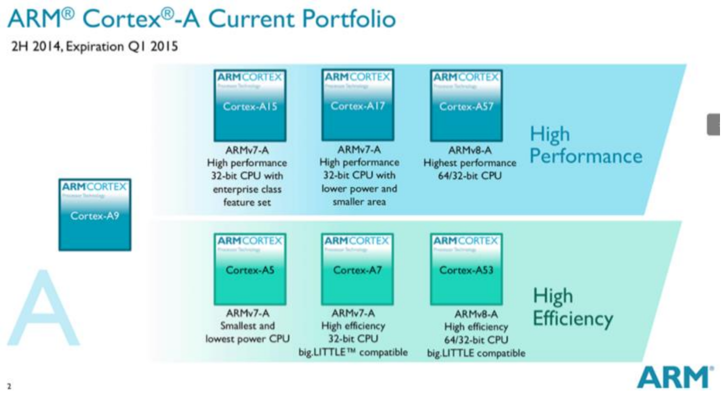
394.2kWh ·0.12€/kWh=47.31€
118.26 kWh ·0.12€/kWh =14.19 €

33.12 € saved every year. As we see it is a remarkable amount of money saved. Yet this is a project and so far only five SBC are connected among them. This is meant to be forwarded to higher scales at big centers of data, labs, universities and also basic home users. The more data we want to handle, the more sense makes to acquire an array with thousands of SBC interconnected.

V. HARDWARE

- Cortex-A7 PROCESSOR

Since the Urban Node project, where a Raspberry Pi was proposed to be the microcomputer of the node, new SBCs have been developed. Therefore research was required to find the best SBC that fulfill the requirements of Teseus. The design of the hardware can be crucial when it comes to energy efficiency, computational power and power consumption that can save time and money. Thereby this also results in being less of a toll on the environment as well as being an important piece in the development of society. The project needed to use a computer with a processor that focuses in efficiency. Besides, another requirement was that its processor had to support virtualization in order to facilitate the software configurations and its price had to be below the supervisor's available budget, which was 50 euros per SBC.
It was explained in a previous point that those computers that use ARM processors were best for this project. Nevertheless what ARM processor to choose was another aspect to take into account. Among the studies we found out that *arm.com* provides a great variety of processors depending on efficiency or performance. The focus lie on efficiency and below we can see that Cortex A-5, Cortex-A7 and Cortex-A53 were the three best options.[7]


ARM® Cortex®-A Current Portfolio
2H 2014, Expiration Q1 2015
Cortex-A15
ARMv7-A High performance 32-bit CPU with enterprise class feature set
Cortex-A17
ARMv7-A High performance 32-bit CPU with lower power and smaller area
Cortex-A57
ARMv8-A Highest performance 64/32-bit CPU
High Performance
Cortex-A9
Cortex-A5
ARMv7-A Smallest and lowest power CPU
Cortex-A7
ARMv7-A High efficiency 32-bit CPU big.LITTLE™ compatible
Cortex-A53
ARMv8-A High efficiency 64/32-bit CPU big.LITTLE compatible
High Efficiency
A
ARM
2

arm.com

V. HARDWARE

- Raspberry Pi2, SBC

After discovering that ARM technology would be the best option and Cortex A7 would be one of the most efficient processors of ARM, it was still needed to know what exact SBC the team would choose for the final design. At first there were several options: *Beaglebone Black, Cubieboard, ObroidC1, RaspberryPi2, Banana Pro.*[8]

Knowing what the requirements were, it was not difficult to conclude that Raspberrypi2 was the best SBC for the project. Below all the reason why RPpi2 was chosen:

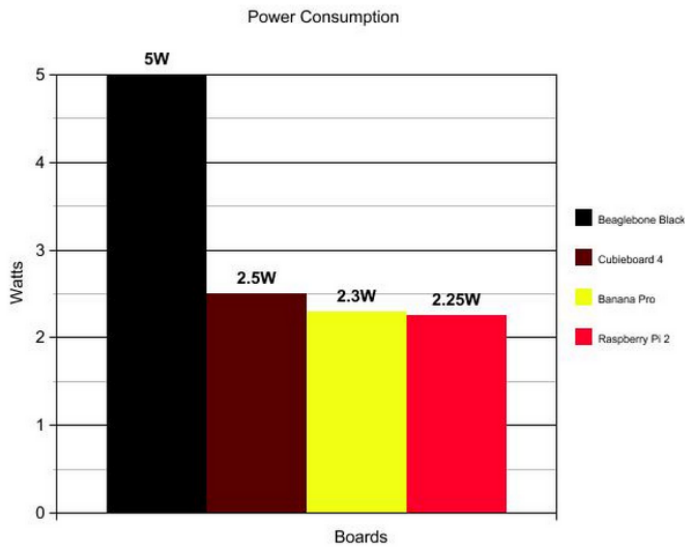
-Uses Cortex-A7 processor which is among the most efficient processors of ARM and from those that are in the efficient groups is the most powerful computational wise.

-It allows virtualization, which was very necessary for thee software grid configuration.

-It is included in the budget range the supervisors of Neapolis provided.

-It has four Cores, unlike Banana Pro which was its best competitor. It only had two Cores.

-It had the lowest power consumption. Below we can see it:



V. HARDWARE

- Connection, Switch

The main difference between a router and switch is that router forwards networks IP addresses and switch forwards MAC addresses, so routers operate on layer three and switches operate on layer two in the OSI model. Another difference is that switches are for LAN and routers for WAN. Furthermore routers transmit packets and switches transport frames.

For our project we decided to connect the different boards to a switch, since one of the goals is to connect different devices in the same network and connect the switch which allows connection to the internet with any router. This means that we can control our own network with a switch configuration as needed.[9]

VI. SOFTWARE

-Beowulf, grid configuration

First off it was required to install the Raspberry Pi 2 software it is provided in their webpage *raspberrypi.org*; it is called Raspbian and it is simple to install, they provide very clear steps to follow.
After that a new study was carried out to find out all the option and afterwards all the feasible option there were to create a cluster configuration with the Raspberry Pi 2.
The different options for connecting all the Raspberry Pi 2 and make them work together was:

-Cloud
-Grid
-Distributed Operating system

In order to realize our vision of a single system it was necessary to configure a distributed computer system, were all the nodes cooperatively work together as a single unified system. As it has been previously mentioned, another requirement was the use of ARM architecture in the system, which ruled out software options such as MOSIX[10] which was the best distributed operating system option. The best solution if a cluster system is chosen is the Linux Virtual Server that has the load balancing ability. However, if the system were to be a grid, the Globus toolkit appeared to be the best solution that makes is possible to design and create your own grid system.

Eventually most of the options were ruled out due to their unavailability and complex structure that would have extended the limited period of time we were given for the EPS project. Therefore Beowulf was concluded as the operating system we would use.

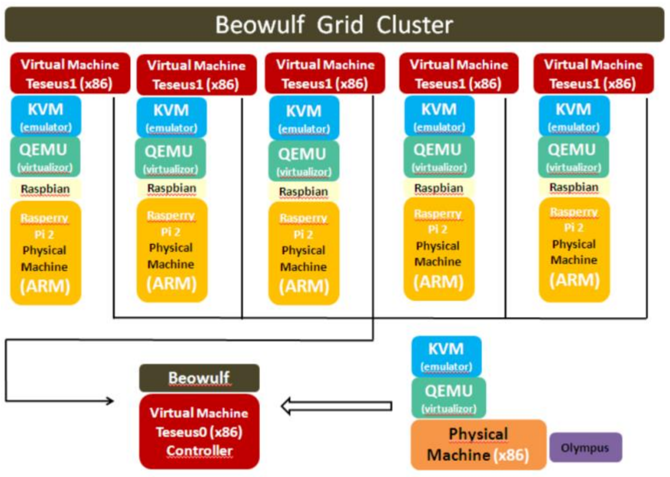
"Beowulf cluster is a computer grid of what is normally identical, commodity-grade computers networked into a small local area network with libraries and programs installed which allow processing to be shared among them. The result is a high-performance parallel computing cluster from inexpensive personal computer hardware."[11]

VI. SOFTWARE

-Design of the Teseus grid

Since the objective is to run x86 applications over ARM processors and using a cluster to provide scalability, we needed to create virtual machines that emulated an x86 machine over ARM hardware.
As a virtualizor we would use QEMU[12] and as emulator we decided to use KVM[13]. In order to emulate and virtualize at the same time we carried out a complex procedure with a lot of libraries installations such as libvirt and other configurations and long procedures. The result was a virtual machine that operated as an x86 processor over an ARM physical machine. Each Virtual machine was named teseus1, teseus2 and so on. They would all we connected with Beowulf.
Besides all that, the whole cluster needed to be controlled by a controller that would be named teseus0. Teseus0 is also included in the cluster as it is another virtual machine that operates with x86 and is connected with the other x86 virtual machine through Beowulf.
Particularly, teseus0 hardware is also x86 and that is because we needed to be able to display all the operations with a screen and we didn't use a Raspberry Pi 2 for that because installing the visualizer would have taken up most of the memory and great part of the power of the SBC.

The whole design and grid configuration has the following structure:



VII. PROBLEMATIC

Several attempts and different solutions have been tested but it is not yet possible to install a virtual machine on the RPi2s in order to run x86 application on the ARM hardware. The Raspberry Pi has an interruption manager incorporated in the software, but in the RPi2 this is a part of the hardware. The problem can be solved, but it requires the time to rebuild the kernel to isolate one or three cores in order to run QEMU, KVM is installed but not supported. Another solution is waiting for new kernel update that enables virtualization extensions out of the box.

It is possible to install the necessary libraries and programs to handle the Beowulf cluster but when installing the same features on the RPi2s we experience crashes once again.

The initial configuration of Teseus has been made. However, when trying to install an x86 program on the ARM boards many obstacles appeared, such as: The Raspberry Pi 2 board is too new when considering installing the necessary OS. Most programs only run on x86 machines, but in the future a great deal more will be made for ARM programs and that is good news for Teseus.

VIII. CONCLUSIONS

The best suited single board computer chosen as nodes in Teseus is the Raspberry Pi 2 due to its high ranking when considering computational power, energy consumption and price. The SBC will have much lower

power consumption than the x86 computers, and by using a compilation of smaller but more and cheaper boards, makes it possible to control the energy consumed even more likely. In order to connect hardware parts, Ethernet was chosen to connect our computer with a switch and not a router.
As far as the software part is concerned, the Beowulf grid configuration was chosen for its availability and simplicity.
However, when it comes to create x86 virtual machines the Raspberry Pi 2 does not seem to be compatible yet. There are several options, such as waiting for new updates or changing the whole kernel configuration but we were given a certain amount of time and the whole team did their best.

The concept of Teseus might be ahead of its time, but not by much, so with a few adjustments in the hardware or the software compatible with RPi2, Teseus will be a powerful grid. Due to the fact that most of the network infrastructure in Vilanova is fiber optics, makes the city a good place to start using the RPi2s as the microcomputer in the Urban Nodes. It is highly likely to build a low cost, scalable data server with existing technology, but the configuration of creating a grid system and pooling the computer resources as well as having task distributed to several nodes at the same time, takes more time to develop than first anticipated.

ACKNOWLEDGMENT

Apart from us, the EPS students, more people have participated in the creation of this project and involve both professors and other students. We students are very thankful to the supervisors and professors that have been inspiring and we have learned so much, and we are forever grateful. They taught us how to organize the project with the scum methodology and gave us guidelines especially at the beginning. We received classes and guidance in programming, and we would not be here without them. We have had several meetings and they were always willing to give constructive criticism and feedback to guide us in the right direction.

In addition we are grateful of Neàpolis for providing an interesting project, lending us their facilities and for Josep Farré, who always were willing to help us with any difficult obstacles that we might have faced during the course of this project. Due to our inexperience in the informatics major, without him probably it would have been impossible to develop the project the way we did it.

We are the most thankful for the help and encouragement from the computer science students, Joan, Josep and Lorena that were always there for any inconvenient or doubt that came up. We truly appreciate their patience, professionalism and kindness. Specially during the last sprint of the project.

Our acknowledgment is also for the maintenance team of the university and mostly Santi, who provided us most of the material and tools necessary to assemble the rack holder.

REFERENCES

Books:

1: Urban Node project: Creating a new urban element to turn Vilanova i la Geltrú into a Smart City (EPS 2013) and Urban Node Project 2014

Websites Front page:

2: www.neapolis.cat/

3: http://en.wikipedia.org/wiki/Single-board_computer

4: https://www.youtube.com/watch?v=_dNvDHhMlgk

5: http://www.androidauthority.com/arm-vs-x86-key-differences-explained-568718/

6: http://comparadorluz.com/faq/precio-kwh-electricidad

7: http://arm.com/products/processors/cortex-a/index.php

8: http://en.wikipedia.org/wiki/Comparison_of_single-board_computers

9:http://www.webopedia.com/DidYouKnow/Hardware_Software/router_switch_hub.asp

10: http://www.mosix.cs.huji.ac.il/

11: http://en.wikipedia.org/wiki/Beowulf_cluster

12: http://wiki.qemu.org/Main_Page

13: http://www.linux-kvm.org/