ICT-04-2015: Customised and low power computing

M2DC
"Modular Microserver Data Centre"

D4.1 – First report about resource and thermal management

Due date of deliverable: 30th September 2016
Actual submission date: 30th September 2016
Submission date of revised version: 30th October 2017

Grant agreement number: 688201
Start date of project: 1 January 2016
Revision 1.2

Lead contractor: Poznań Supercomputing and Networking Center (PSNC)
Duration: 36 months

Project co-funded by the European Commission within the EU Framework Programme for Research and Innovation HORIZON 2020.	
Dissemination Level	
PU = Public, fully open	X
CO = Confidential, restricted under conditions set out in the Grant Agreement	
CI = Classified, information as referred to in Commission Decision 2001/844/EC.	

D4.1

First report about resource and thermal management

Editor

Wojciech Piatek (PSNC)

Contributors

Wojciech Piatek, Ariel Oleksiak, Mateusz Jarus, Michał Kierzynka (PSNC), Daniel Schlitt, Christian Pieper (OFF), Stefan Krupop (CHR), Holm Rauchfuss (HUA), Paweł Stefanski, Bartosz Misiaszek (BEYOND), Senechal Emmanuel, Raphael Bendel (RxC), William Fornaciari, Federico Terraneo (POLIMI)

Reviewers

Gunnar Billung-Meyer (CHR), Loïc Cudennec (CEA), Thierry Goubier (CEA)

30th October 2017

Revision 1.2

The work described in this document has been conducted within the project M2DC, started in January 2016. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688201.

Document History

Version	Date	Author(s)	Description/Comments
0.1	24.06.2016	Wojciech Piatek (PSNC)	Outline
0.11	18.07.2016	Wojciech Piatek (PSNC)	More detailed outline + first comments
0.2.1	12.08.2016	Christian Pieper, Daniel Schlitt (OFFIS)	Chapters 4 and 5 switched; Filled Section 2.2; Filled Section 5.1
0.2.2	16.08.2016	Wojciech Piatek (PSNC)	Input to section 2 and an initial diagram for section 5
0.2.3	17.08.2016	Wojciech Piatek (PSNC)	Merging versions 0.2.1 and 0.2.2, reorganization of document structure (Subsection 5.1 extracted to separate Section 3)
0.2.4	13.09.2016	Senechal Emmanuel, Raphael Bendel (RxC)	Chapter 4: Power and Thermal Model (Draft version)
0.2.5	13.09.2016	Stefan Krupop (CHR)	Chapter 5.2: Support on hardware layer
0.3	14.09.2016	Wojciech Piatek (PSNC)	Filled Section 2, merged previous contributions
0.4	14.09.2016	Daniel Schlitt (OFFIS)	Chapter 3 - description of basic interaction between management tools
0.41	16.09.2016	Wojciech Piatek (PSNC)	Merging Christmann's corrections to Section 5.2. Rearranging Section 4 and 5.
0.41	16.09.2016	Christian Pieper (OFFIS)	Section 4.1 Replaced text and images (Sorry I overwrote v0.41)
0.42	19.09.2016	Paweł Stefanski, Bartosz Misiaszek (BEY)	Section 1. Introduction
0.43	19.09.2016	Wojciech Piatek (PSNC)	Improvements to Section 2
0.44	19.09.2016	Christian Pieper (OFFIS)	Swapped Section 4.1 and Section 4.2, improved Section about Software Components Added comments to Introduction and replaced image annotation
0.5	20.09.2016	Wojciech Piatek (PSNC)	Filled section 6,
0.51	20.09.2016	Wojciech Piatek (PSNC)	Refactored image and tables captions. Updated bibliography and citation list
0.6	20.09.2016	Mateusz Jarus (PSNC)	Added the description to Section 5.2 Power models
0.61	20.09.2016	Wojciech Piatek (PSNC)	Overall refactoring and formatting

0.62	20.09.2016	Mateusz Jarus (PSNC)	Changed the description of Section 5.2 Power models
0.63	20.09.2016	Wojciech Piatek (PSNC)	Added description of thermal models for CPU to Section 5.3. Reorganization of Section 2
0.64	20.09.2016	Christian Pieper (OFFIS)	Suggestions for improvements of the introduction
0.65	21.09.2016	Ariel Oleksiak (PSNC)	Abstract added. Section "5.2 Power models" modified
0.66	21.09.2016	Paweł Stefański (BEYOND)	Merged 0.64 changes and modified Introduction section.
0.67	21.09.2016	William Fornaciari Federico Terraneo (POLIMI)	Extended the SoTA @2.12. Waiting for final polishing (in the case other contributions will appear)
0.68	22.09.2016	Christian Pieper (OFFIS)	Suggested changes to Chapter 2 - 4
0.69	22.09.2016	William Fornaciari Federico Terraneo (POLIMI)	Section 6 extended
0.70	22.09.2016	Michał Kierzynka (PSNC)	Power model description of GPU modules
0.71	22.09.2016	Mateusz Jarus (PSNC)	Figures that present the classification of power-aware, thermal-aware and fans management techniques
0.73	22.09.2016	Ariel Oleksiak (PSNC)	Conclusions and modifications to Section 6 added
0.74	22.09.2016	William Fornaciari	Section 6 revision. Final check of references
0.8	22.09.2016	Ariel Oleksiak (PSNC)	Version for internal review
0.9	28.09.2016	Paweł Stefański (BEYOND)	Incorporated changes from review.
0.91	28.09.2016	Wojciech Piatek (PSNC)	Added missing Section 6. Addressed review comments in Sections 2.1, 5.3.2, 6.1, 6.2, and 6.3.
0.92	29.09.2016	Emmanuel Senechal (RxC)	Section 5 updated
0.93	29.09.2016	Daniel Schlitt (OFFIS)	Feedback incorporated
0.94	30.09.2016	Holm Rauchfuss (HUA)	Small changes
1.0	30.09.2016	Wojciech Piatek (PSNC)	Merged previous versions, merged Polimi's contribution, updated figures and formatting, corrected spelling, updated references, final version
1.1	10.02.2017	Wojciech Piatek (PSNC)	New version of the document
1.2	30.10.2017	Wojciech Piatek (PSNC)	Glossary added

Rev-0.1	13.12.2016	Wojciech Piatek (PSNC)	Update Section 5.3, version for 1st internal review
Rev-0.2	19.12.2016	Wojciech Piatek (PSNC)	Addressing reviewer's comment
Rev-0.3	06.02.2017	Wojciech Piatek (PSNC)	Minor modifications to Section 5.3 (updated figure, typos corrections), version for 2nd internal review
Rev-0.4	10.02.2017	Wojciech Piatek (PSNC)	Addressing reviewer's comment, added Executive Summary
Rev-0.5	14.02.2017	Wojciech Piatek, Daniel Schlitt	Final improvements

Executive summary

This document presents results of first analysis performed within Task T4.4: Intelligent Resource and Thermal Management. Based on the studies of the current state of the art and existing solutions in off-the-shelf servers, the document introduces the concept of resource and thermal management policies within the M2DC system.

In general, the Resource and Thermal Management module consists of three functional components: Fan Manager, Energy Saver Manager and Power Capping Manager. The Fan Manager is responsible for adjusting the fan speeds in order to keep all components within the desired range of temperature and to optimize their power usage. The Energy Saver Manager is in charge of taking actions that reduce the power consumed by RECS®|Box components. These actions include management of their power states, dynamic addition/removal of cores and exploitation of dynamic voltage and frequency scaling (DVFS) – if available – to optimize speed of processing units while meeting thermal constraints. Finally, the Power Capping Manager allows users to provide a limit for the maximum power drawn by the system. It will utilize both the Fan Manager and the Energy Saver Manager, considering fan management as well as the management of power states and DVFS in order to reduce the power.

One of the main challenges for the managemnet of the RECS®|Box is the interaction between workload and resource management policies. As both modules may want to affect the power state of computing nodes, some coordination between them is required. To this end, they will exchange dedicated management data concerning thermal constraints and workload forecasts for the compute nodes. In this way, the Workload Management module will be able to make the scheduling decisions with respect to the current thermal state of the system. On the other hand, the Resource and Thermal Management module will consider future system utilization while changing the power states of the computing system. However, in case of two modules working together the recommended approach assumes that node power management decisions are made using the Workload Management module and the Resource and Thermal Management module performs fan management, DVFS and power capping actions.

To fully support both management subsystems (Intelligent Management), RECS®|Box will be equipped with dedicated hardware and software interfaces for exchanging data between M2DC components. Hardware and corresponding firmware of RECS®|Box provide IPMI and a restful API supporting controlling and monitoring of the hardware subcomponents. Additionally, the Intelligent Management will interact with OpenStack services (in particular with Ceilometer and Nova) to retrieve monitoring data and manipulate the schedules. Both management subsystems will also utilize power and thermal models created for the microservers that can be placed within RECS®|Box. These profiles will support analysis of their energy and thermal behaviour. Based on the models (predicting future trends) and analysing collected data, the Resource and Thermal Management module will perform energy optimization actions.

The revision (Version 1.1) of this deliverable comes with the revised version of Section 5.3. It focuses on the models that are crucial from the perspective of RECS®|Box. Thus, image, description of considered temperature values and particular model's parameters were modified and adjusted to better reflect the M2DC approach.

Table of Contents

1 Introduction 3
1.1 Motivation 3
1.2 Document Structure 7
2 State of the art 8
2.1 Resource management policies 8
2.1.1 Power-aware resource management 8
2.1.2 Thermal-aware resource management 9
2.1.3 Fans management 12
2.1.4 Resource management in off-the-shelf servers 13
2.2 Workload management policies 15
3 Interaction between resource and workload management policies 16
4 M2DC power and thermal management interfaces 19
4.1 Support on hardware layer 19
4.2 Components and Interfaces on Software Layer 20
5 Power and thermal models 22
5.1 RECS®|Box concept 22
5.2 Power models 25
5.2.1 Power model description of the x86 CPU modules 25
5.2.2 Power model description of ARM microservers 29
5.2.3 Power model description of GPU modules 29
5.2.4 Power model description of HP-FPGA-based microserver 31
5.3 Thermal models 33
5.3.1 Thermal model of a component 33
5.3.2 Thermal model of the CPU module 33
5.3.3 Thermal model of the HP-FPGA-based microserver 36
5.3.4 Thermal model of the GPU module 38
5.3.5 Thermal model of the low power ARM board 40
6 Plans towards resource and thermal management policies 44
6.1 Energy Saver Manager 44
6.2 Fan Manager 44
6.3 Power Capping Manager 45
6.4 Dynamic Thermal Manager 45
7 Conclusion 46
8 Glossary 47
9 References 49

Table of Figures

Figure 1: Annual spending for DC operation - source - 451 Research (CoolEmAll project [36]) 3
Figure 2: Temperature limit references 4
Figure 3: Airflow visualisation made by SVD Toolkit from CoolEmAll project 5
Figure 4: Thermal picture of 2U server rear in Beyond DC1 (courtesy of Beyond.pl) 6
Figure 5: Example screen of modern DCIM software (courtesy of Intel Corp.) 6
Figure 6: ILO server monitoring with 3d graph temp visualisation (courtesy of HPE company) 7
Figure 7 Classification of power-aware resource management techniques 9
Figure 8 Classification of thermal-aware resource management techniques 12
Figure 9 Classification of fans management techniques 13
Figure 10: Interaction between workload and resource & thermal management 16
Figure 11: Node power management accessed by resource and workload management leads to conflicting optimization interests 17
Figure 12: Workflow for parallel usage of thermal and workload management 18
Figure 13: Interfaces between Intelligent Management and other components 20
Figure 14: Interfaces within Intelligent Management 21
Figure 15: small chassis with 3 microserver baseboard slots 23
Figure 16: standard chassis with 9 microserver baseboard slots 24
Figure 17: Dependency between power usage and CPU load for different clock rates on Intel i7 3615QE processor 26
Figure 18: Dependency between power usage and clock rate for different CPU loads on Intel i7 3615QE processor 27
Figure 19: Dependency between power usage and CPU load for different clock rates on Intel i7 2715QE processor 28
Figure 20: Dependency between the power usage and the clock rate for different CPU loads on Intel i7 2715QE processor 28
Figure 21: Heating components picture on the FPGA board 32
Figure 22: Thermal path for a single component 33
Figure 23: Parameters for CPU thermal model 34
Figure 24: Thermal model of the FPGA board 36
Figure 25: Thermal simulation for the devices sharing the heatsink 37
Figure 26: Thermal simulation for the LP ARMV8 board 39
Figure 27: Thermal simulation for the Zynq microserver 41
Figure 28: Thermal simulation for the TK1 microserver 41
Figure 29: Thermal simulation for the EXYNOS microserver 42
Figure 30: Thermal simulation for the T30 microserver 42

Table 4-1: Monitoring/control items and available APIs to influence them 20
Table 5-1: HP-FPGA-Microserver / Estimated thermal and power inputs 37
Table 5-2: Junction temperatures computed for the devices sharing the heatsink 38
Table 3: LP-ARMV8-Microserver / Estimated thermal and power inputs 38
Table 4: Main devices per module 40
Table 5: APALIS based Microservers / Estimated thermal and power inputs 40
Table 6: Temperatures computed for the devices and the heatsink 43

1 Introduction

Appropriate resource, power and thermal management of servers is essential to ensure high performance, reliability and energy efficiency of complex IT systems that are basis of clouds, high performance computing infrastructure, and other advanced and demanding applications. These functions allow lowering the operational costs (OPEX) of data centres, which is especially important as these costs are large fraction of total costs often being even higher than capital costs (CAPEX), cf. Figure 1. Reduction of operational costs can be achieved by lowering energy consumption of the whole data centre, maintenance costs related to diagnosis and repair/replacements of failed hardware, and number of breaks. This should lead to minimization of Total Cost of Ownership (TCO), which is one of main objectives, along with gains in energy efficiency, of the project.

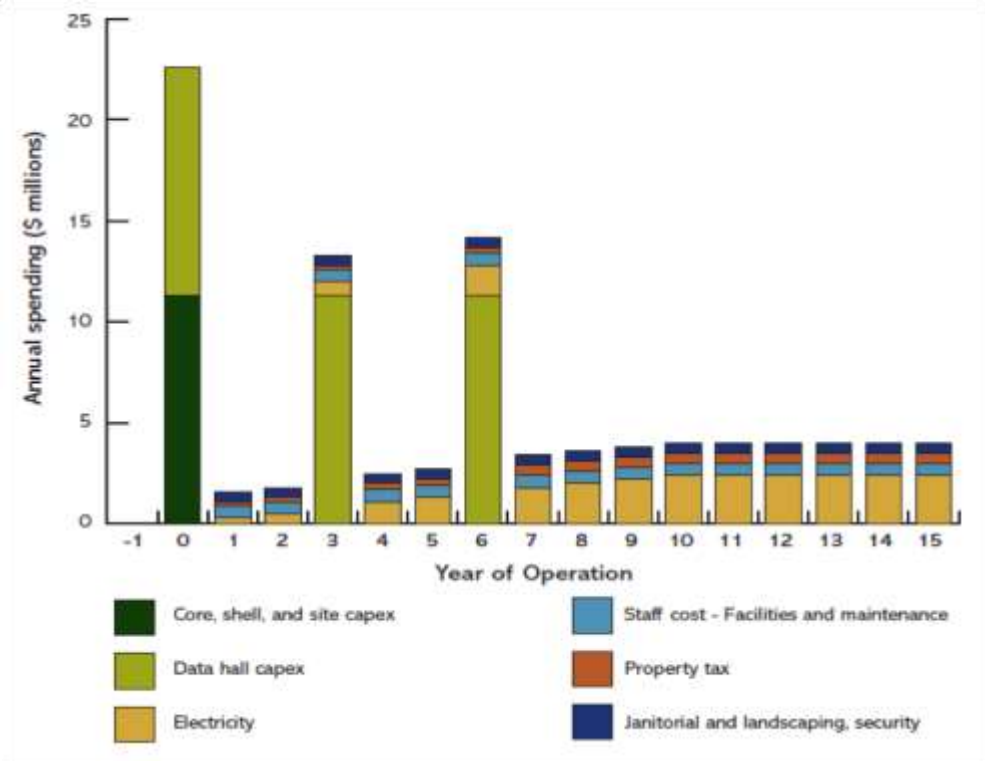

Figure 1: Annual spending for DC operation - source - 451 Research (CoolEmAll project [36])

Therefore, in this document we study possible power and thermal management policies and analyse which of them can be applied for M2DC Appliances. We distinguish resource (power) management and thermal (for both computing nodes and fans) policies.

1.1 *Motivation*

Thermal resource management is a key operational challenge in data centres. Modern facilities integrate high performance infrastructure, demanding a high capacity of cooling, but offering the ability to react very quickly on any environmental changes. The thermal capacity of a standard server, chassis, cabinets and the rest of the equipment is very low, which means any power consumption changes occur rapidly and need to be compensated by effective cooling right away. Data centres are designed with focus on thermal distribution, airflow and environmental metrics inside chambers and cabinets (see [1], Section 6.2). Changes in available space, usability and space efficiency affect those designs concerning the balance between thermal optimizations and customer expectations. Except for technical and design issues, we face a strong economic challenge to have systems as power efficient as possible. This also includes deploying applications, which require low CPU resources, running on low power microservers. This approach does not only reduce the primary energy for running the equipment but also the secondary energy for cooling the data centre. Thus, beside the maintenance costs the Total Cost of Ownership (TCO) is mainly driven by power (this means energy)

prices. If we can lower the thermal margin of our installations - in terms of a lower TCO, lower data centre service costs and consequently – all customers would benefit from lower prices and more competitive offers.

Data centres sustain specific environmental metrics on a declared level to comply with regulations and hardware vendor's specifications. Most data centre operators declare that they comply with given guidelines from the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE), e. g. concerning the temperature and humidity ranges from 2011 (see [1], Section 6.3). The ASHRAE A1 range defines a narrow window of allowed temperatures for most of the operation time as 18-27℃, as depicted in Figure 2, and humidity as 42-60 % RH [2]. A slightly wider window is allowed as a strict limitation because maximal and minimal allowed temperatures are defined as not allowed to exceed. Data centres keep within the defined ranges, constantly trying not to exceed those values, in order to prevent risks to IT equipment reliability. ASHRAE defines several points that define boundary levels of recommended and allowed temperature and humidity ranges. The detailed description of data centres thermal management is covered in M2DC document - D7.2 – Market Assessment, section 6.3 – Cooling.

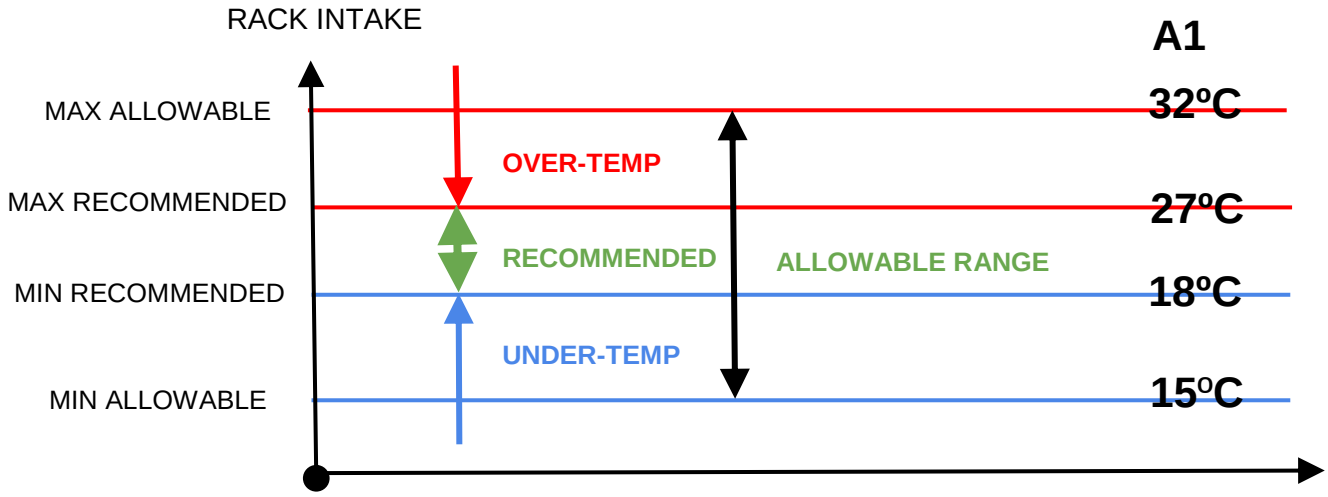

Figure 2: Temperature limit references

The DC operator should concentrate the cooling efforts on air inlet to the equipment, while air exhaust from the device is not so important. The complexity of the airflow inside a typical DC chamber is so big that dedicated simulation software is required to make a model of the cooling efficiency and predict temperatures in different segments of chamber aisles. A good example of this type of tool is the SVD simulation toolkit, shown in Figure 3, made within CoolEmAll project under auspices of the ICT/EU.

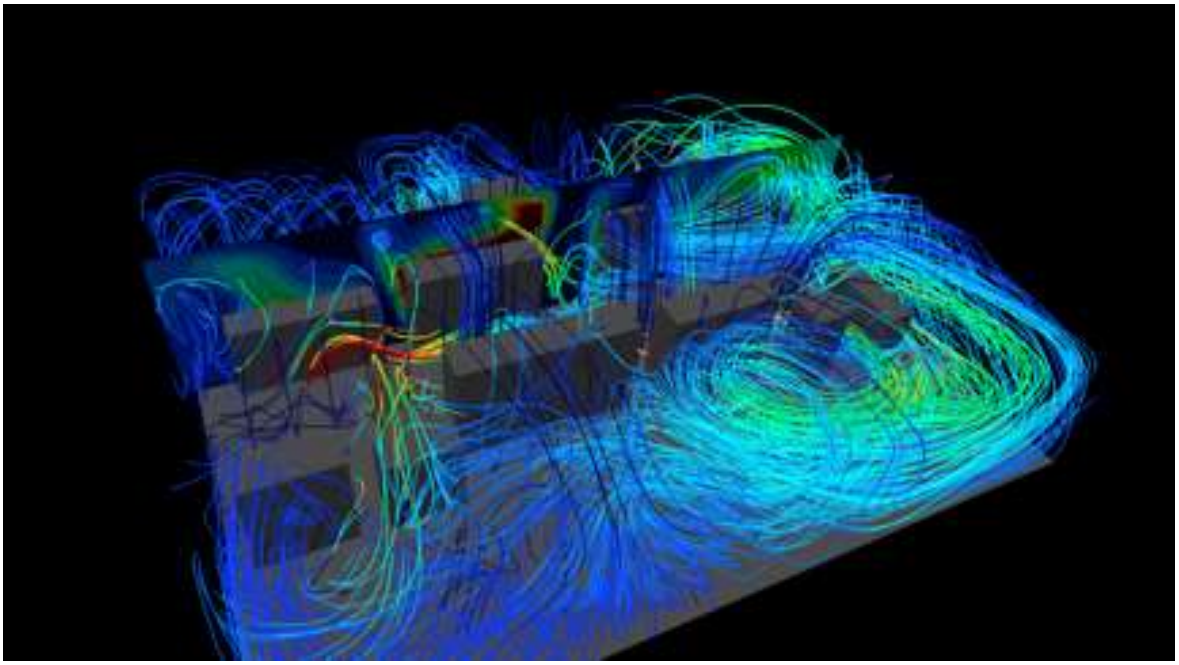

Figure 3: Airflow visualisation made by SVD Toolkit from CoolEmAll project

To successfully operate environmental values and optimize hot assimilation with minimal cooling effort all components need to be optimized. We start with high efficiency, low conversion loss power supplies, optimized airflow inside chassis, well designed and maintained rack cabling, and optimized aisle construction with air curtains for better air isolation.

As a universal metric to measure DC efficiency, the Power Usage Effectiveness (PUE) was introduced. It is the ratio of power used or lost by a data centre facility infrastructure (pumps, lights, fans, conversions, UPS…) to the power used by computing devices. There are many factors that determine the minimal PUE, for example the location, outside temperature, solar activity, type of facility building, cooling technology, use of free cooling, adiabatic processes or condensation. Everything combined creates the minimal PUE which the specific DC is able to achieve (as certain environmental or power usage values).

$$PUE = \frac{IT\ Power\ +\ facility\ power}{IT\ Power}$$

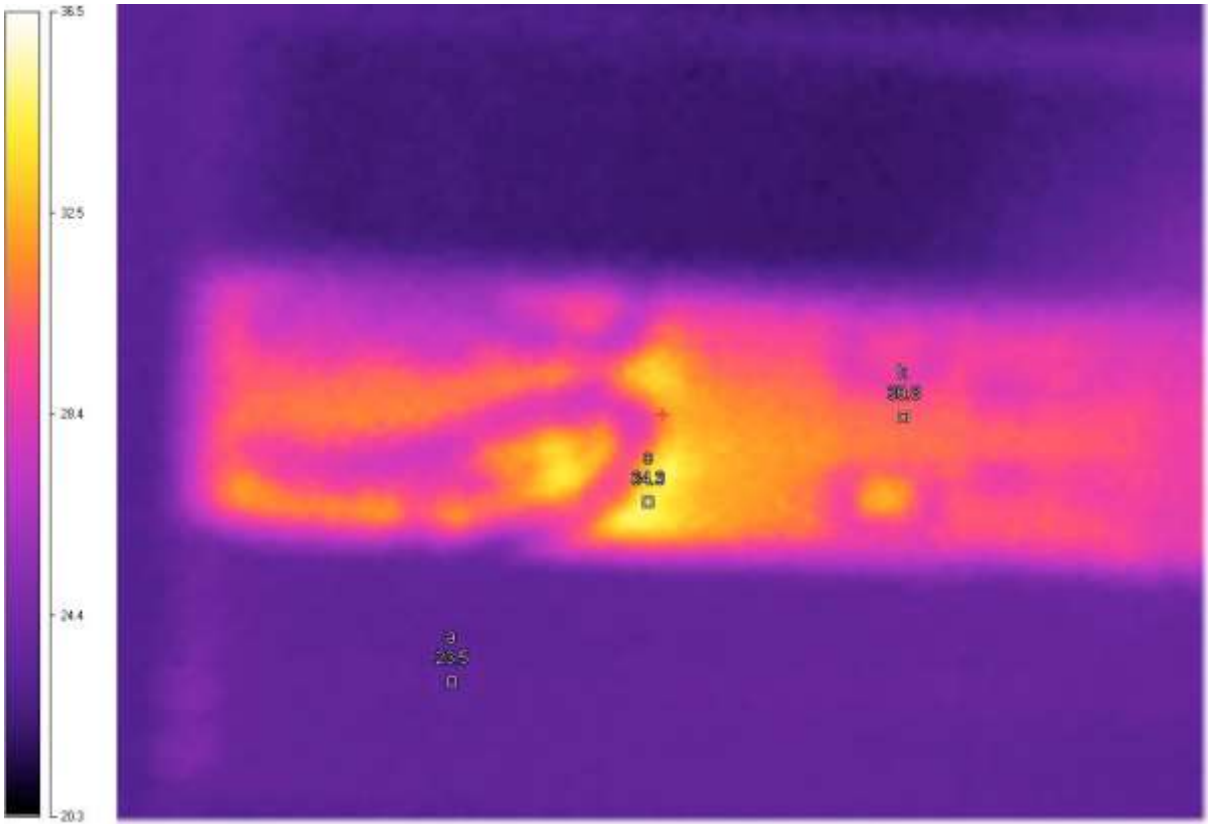

Figure 4: Thermal picture of 2U server rear in Beyond DC1 (courtesy of Beyond.pl)

Data centre infrastructure management software (DCIM) is used to monitor environmental, power and a number of other metrics, see Figure 5 and Figure 6. It's the eye and ear of a data centre operator that keeps track of all values and even reacts when critical values are exceeded. Many different vendors provide complete solution for the whole DC infrastructure. We can expect big changes in this kind of software due to enforcement of the green DC idea and focus on power management options. Nowadays a key factor for DCIM software is the overall power consumption and efficiency factor of the DC infrastructure – the PUE. We see that future DCIM solutions will also integrate a lot more with running devices - like servers and network devices, to cover the full set of data source and work more precisely on DC operation and delivery.

Figure 5: Example screen of modern DCIM software (courtesy of Intel Corp.)

Gaining insights into data collected from end user devices expands the whole situation and gives far more knowledge about DC operation and running conditions. With power capping and enforced limitation for power consumptions we can control and balance the whole DC efficiency and for example limit computations at day to use more free cooling at night.

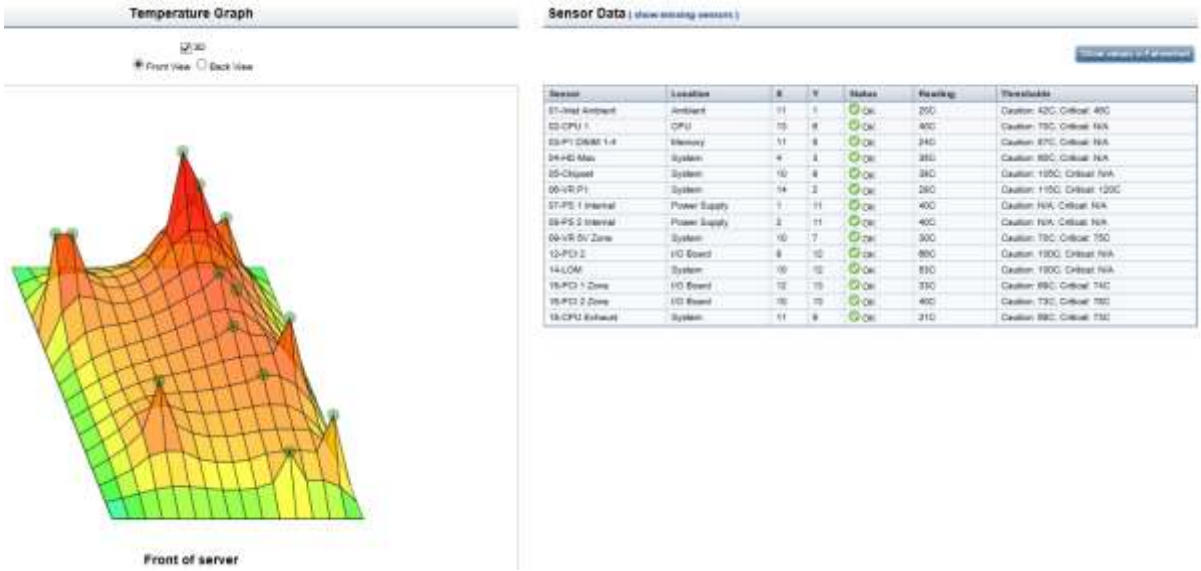

Figure 6: ILO server monitoring with 3d graph temp visualisation (courtesy of HPE company)

Some of the most common challenges in data centres are

- Rapid temperature changes due to low occupancy
- Capacity planning
- Under equipped cabinets
- Low power consumption which causes under-use of cooling facility and instability

1.2 *Document Structure*

As this is a first report from Task T4.4, the document concentrates on the state of the art in the area of power and thermal management. It also introduces the M2DC hardware (the RECS®|Box and potential microservers) within this context. In particular, we summarize information about power and thermal models of specific components that will be (or can be) integrated into the M2DC microserver system. As intelligent resource and thermal management will require development of software components responsible for this task, the designs of software architecture and interfaces to other components are also included in this deliverable. The software description also includes definitions of links to workload management (that will be developed within Task 4.5), which is closely related to resource and thermal management so these two tasks must be synchronised. The document also includes first concepts of policies that will be implemented and tested with the M2DC appliance.

2 State of the art

In recent years, power and thermal-oriented management has become an investigated research area [3]. In general, energy and thermal optimization can be achieved by the means of both proper resource and workload control. Thus, the following section has been divided with respect to these two, sometimes inseparable, techniques. Below the highlights of the conducted studies in that area and their results are given.

2.1 *Resource management policies*

By resource management we understand controlling the states of resources in the manner that it is efficient and effective for the whole computing system. As the issues related to energy-efficiency of computing systems are gaining more and more interest due to the rapid growth of the system scale and related energy costs, reasonable resource management becomes a crucial part of computing system control. For now, lots of effort has been put on managing the IT infrastructure [4] and cooling equipment [5]. With respect to the main purpose behind these actions we decided to classify them as: power-aware resource management (Section 2.1.1), thermal-aware resource management (Section 2.1.2) and fans management (Section 2.1.3). Activities taken towards lowering the power consumption may include dynamic power management (DPM), describing actions like managing the power states of nodes and other system components, but also dynamic voltage and frequency scaling (DVFS) and fetch toggling. Thermal-aware policies also contain dynamic voltage and frequency scaling. Fans management studies refer to optimal control of their speed aiming both on reducing their power consumption as well as keeping the cooled resources within the given temperature range.

2.1.1 Power-aware resource management

There are lots of studies undertaken towards the optimization of power used by IT components. Most of the works concentrate around the aforementioned DVFS or DPM. In terms of DVFS there are several solutions aiming at minimizing the processor power consumption. Hsu and Feng [6] propose an automatic algorithm, leveraging DVFS that adapts processor voltage and frequency settings to reduce its power consumption with minimal impact on performance. The proposed algorithm makes the scheduling decision during the program execution, adjusting the frequency to the optimal level each second. These decisions are based on the given slowdown constraints together with the power usage minimization. Moreover, the algorithm does not require any application-specific information a priori but acquires all the data during application execution. In case of any unavailability of the desired frequency, it is emulated using the neighboring ones. As a result, the solution can save up to 25% energy consumed by the processor, maintaining performance degradation at the 3-5% level. In [7], the authors introduced an intra-task DVFS technique under compiler control that exploits program checkpoints. Checkpoints are generated during compilation and indicate places in the code where the processor speed and voltage should be re-calculated. Moreover, checkpoints also carry user-defined time constraints. The proposed algorithm handles multiple intra-task performance deadlines and modulates power consumption according to a run-time power budget. The authors compared their solution against existing ones gaining 63% more energy savings. Another solutions used the barrier synchronization mechanism to deal with the energy consumed by the faster cores. Liu et al. [8] tracked the idle times spent by a processor waiting for other processors to get to the same point in the program. Then, they used per-core DVFS to manage their frequency, in such a way that both the idle time due to the waiting and energy consumption are reduced. Similarly Li et al. [9] proposed a solution called thrifty barrier that saves energy of faster nodes. However, contrary to previous approach, it places the faster cores into a lower power mode at the barriers (instead of slow downing them) while waiting for the slower cores so that energy can be saved.

DPM actions refer to the selective shutdown of system components that are idle or underutilized. The simplest technique to reduce power consumption is to power down cluster nodes. Kamitsos et al. attempt to find an optimal policy for powering nodes up and down using a Markov decision process [10]. As there is a tradeoff between performance and power consumption, solving the Markov decision process makes it possible to find a Pareto-optimal tradeoff between these two metrics.
In [11], a system-shutdown method to exploit sleep mode operations for energy saving is presented. Each time

the system detects the idle period, it determines whether it should stay in the running state or enter the sleep one. The authors proposed to use an exponential-average approach to predict the upcoming idle period by the accumulative average of the previous idle periods. They also introduced two mechanisms, namely: prediction-miss correction to improve the hit ratio (dealing with impulse-like idle periods) and prewake-up to reduce the delay overhead. To validate their approach they conducted experiments on four event-driven applications: X-server, Netscape, Telnet, and Tin. The proposed approach turned out to be independent of the target applications and achieved high hit ratios for a wide range of shutdown overheads. The authors claimed that their method can be applied to manage both the CPUs as well as other peripherals. A more "server oriented" approach is presented in [12]. Instead of traditional dynamic power management (where the nodes are turned off immediately) they suggested delaying the time until a server is turned off. Analyses were performed using an M/M/k queuing model via Matrix analytic methods. Comparing three management policies ("always on", "on/off" and "delayed off"), the authors demonstrated that turning servers off less frequently with some delay time mitigate the damage caused by large setup times. Finally, the authors showed that the effectiveness of dynamic power management policies increases with the size of the data centre.

Figure 7 presents the classification of the presented power-aware resource management techniques (P – processor, N – node, C – chassis, DC – data centre).

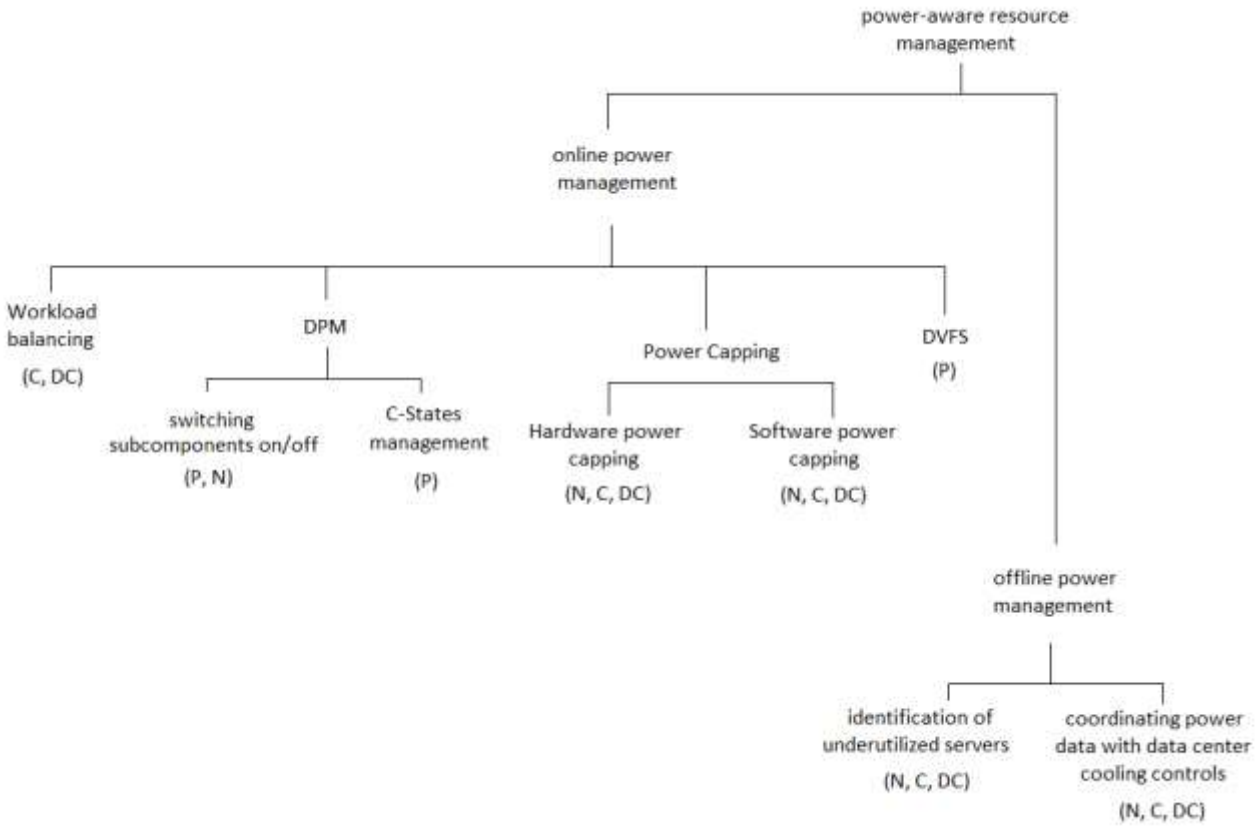

Figure 7 Classification of power-aware resource management techniques

2.1.2 Thermal-aware resource management

There are two main reasons why the temperature of a processor, or in general an integrated circuit has to be kept under control. The first and most obvious reason is to prevent immediate failures such as thermal runaway conditions [13]. This already sets an upper limit to the temperature that an integrated circuit can safely reach. However, many phenomena affecting the reliability of an integrated circuit, such as hot carrier

injection (HCI) [14], electromigration [15], negative bias temperature instability (NBTI) [16], as well as thermal fatigue on the solder joints [17] depend on temperature. It is thus often desirable, or even necessary, to further lower the chip temperature at a value computed from reliability considerations.

Growing importance not only of the power used by IT infrastructure, but also of its thermal condition has made the thermal management almost as crucial as power aspect. More and more solutions have focused on optimizing resource states from the perspective of their thermal efficiency and capabilities. Most of them rely on processor dynamic voltage and frequency scaling. In [18], the authors proposed the combination of load balancing based on task migration and DVFS to reduce cooling energy consumption and prevents hot spot formation. DVFS is used to adjust processor temperature according to the given threshold and, afterwards, tasks are moved in order to address the load imbalance between cores. Wang and Bettati [19] proposed to use reactive processor speed scaling. When the tasks are running and the temperature is below a given threshold, the processor operates at full speed. However, when the threshold is reached, the processor speed is reduced to the level that ensures that temperature will not rise above the predefined threshold. In [20], the authors revealed that using the lowest constant speed might be the optimal method to deal with the temperature. They considered dependency between temperature and the leakage of the processor. They showed that under certain realistic conditions, using the lowest constant processor speed that can guarantee deadlines of all real-time tasks is an optimal method to minimize the maximal temperature for a real-time system. Bansal et al. [21] analyzed policies for setting the speed of the processor to manage its temperature. They assume that a processor cools according to Newton's law of cooling. They provided the approximation of maximum temperature with respect to the maximum energy and gave a new online, cooling-oblivious algorithm for frequency scaling. Finally, they showed that the proposed approach is optimally competitive with respect to the maximum power. Proactive speed scheduling, utilizing DVFS scaling is presented in [22] where the authors proposed two different approaches to meet both, timing and thermal constraints. The timing-optimization approach minimizes response time while thermal-optimization aims to minimize converging initial temperature so that the system can tolerate the thermal constraint to complete more workload. Gained results showed that the proposed proactive approach outperforms existing reactive ones. In [23] an on-line temperature aware DVFS technique is presented. It exploits both static and dynamic slack. The approach consists of two parts: an offline temperature aware optimization step and on-line frequency settings based on temperature sensor readings. The presented approach is aware of the frequency/temperature dependency, by which important additional energy savings are obtained.

Microprocessors are facing a steady power density increase which has eventually lead to the so called "dark silicon" problem [24] [25], i.e., to the impossibility of operating all the processor units at full power without destroying the chip by thermal runaway. For this reason, comprehensive solutions to the thermal management problem require to both design thermal dissipation strategies to be as efficient as possible in removing the heat produced by the computational units, and complement them with optimized dynamic thermal management (DTM) policies to handle power dissipation peaks as well as workloads that overwhelm the selected dissipation strategy. Solutions to maximize performance subject to temperature and power constraints, as well as optimizing the power/performance and thermal/performance tradeoffs, are thus strongly sought.

One of the first DTM policy proposed is the stop and go [26], which halts the processor clock if the operating temperature is higher than a given threshold, and restores normal operation only after it has cooled down. Although this policy yields a considerable performance penalty compared to other techniques [27], it is often used as a secondary policy that acts only in critical conditions.

An improved policy [28] uses a control loop to modulate the instruction fetching of a processor, reducing on purpose the fetch rate if temperature is too high. Contrary to the stop and go policy that uses a binary actuator, this policy can adapt the performance impact to the required control action to keep the temperature under control.

Dynamic voltage and frequency scaling (DVFS) [29], although first introduced to explore the power-performance tradeoff is an effective actuator also for DTM due to its quadratic effect on power consumption, and is being employed as the knob for many DTM policies. One of the first works making use of DVFS in a DTM policy in multicore processors is [27], which proposes a proportional-integral (PI) controller sensing the core

temperature and actuating on its frequency. The scheme is distributed, meaning that each core in the MPSoC has its own controller, and requires per-core DVFS. While the proposed solution achieves remarkable performance, the overhead of the proposed controller is not taken into account in the quoted work.
The heat and run approach [30] proposes a DTM policy based on task migration for MPSoCs. The policy works by moving applications that perform heavy computation away from hot cores. The policy yields a performance improvement compared to stop and go, but suffers from two major drawbacks. First, task migration cannot be performed at a millisecond timescale without incurring in an excessive overhead due to cache lines invalidation. In addition, this solution exacerbates both temporal and spatial thermal gradients, which degrade the overall system reliability [31].
In [32], the authors propose a solution based on convex optimization that assumes the power to be constant and optimizes the workload under steady state temperature constraints. The authors however admit that the solution needs to be complemented with a DTM technique, to account for varying power consumption and ambient temperature.
A Linear Quadratic Regulator (LQR) is presented in [33] to address thermal balancing of a multicore chip based on a model of the system, workload requirements - to be obtained from the OS scheduler - and temperature measurements. The proposed policy is meant to be operated every 100ms, thus not being capable to control temperature in the presence of unexpected power peaks, and its centralized nature, together with the need to interact with the OS scheduler, make it difficult to operate at finer timescales.
Other proposals focus on predictive approaches; a notable example is [34], where a Model Predictive Control (MPC) solution is presented to address thermal management in multi-cores with the objective of smoothing the control actions. The proposed scheme is centralized, and thus its scalability is limited as the number of cores increases. Furthermore, it relies on the strong assumption of knowing the future chip workload.
Analogously, [35] describes a decentralized MPC solution to the thermal management problem, thus solving scalability issues of centralized solutions. The proposal still requires for accurate workload information. The control rule in the quoted work is particularly lightweight for an approach based on MPC, but it has to be applied at a fixed rate, and is therefore limited by the tradeoff between limiting the number of events and achieving good thermal control.
One recent solutions is the sprint computing [36] approach, which consists in relying on the slow thermal dynamics to transiently exceed a chip thermal design power (TDP) without reaching unsafe temperatures. This has already found adoption in commercial applications, in the form of the Turbo boost 2.0 [37], which is a hardware controller in the recent Intel processors capable of operating the cores above their TDP if the temperature state of the chip allows it.

Figure 8 presents the classification of the presented thermal-aware resource management techniques (P – processor, N – node, C – chassis, DC – data centre).

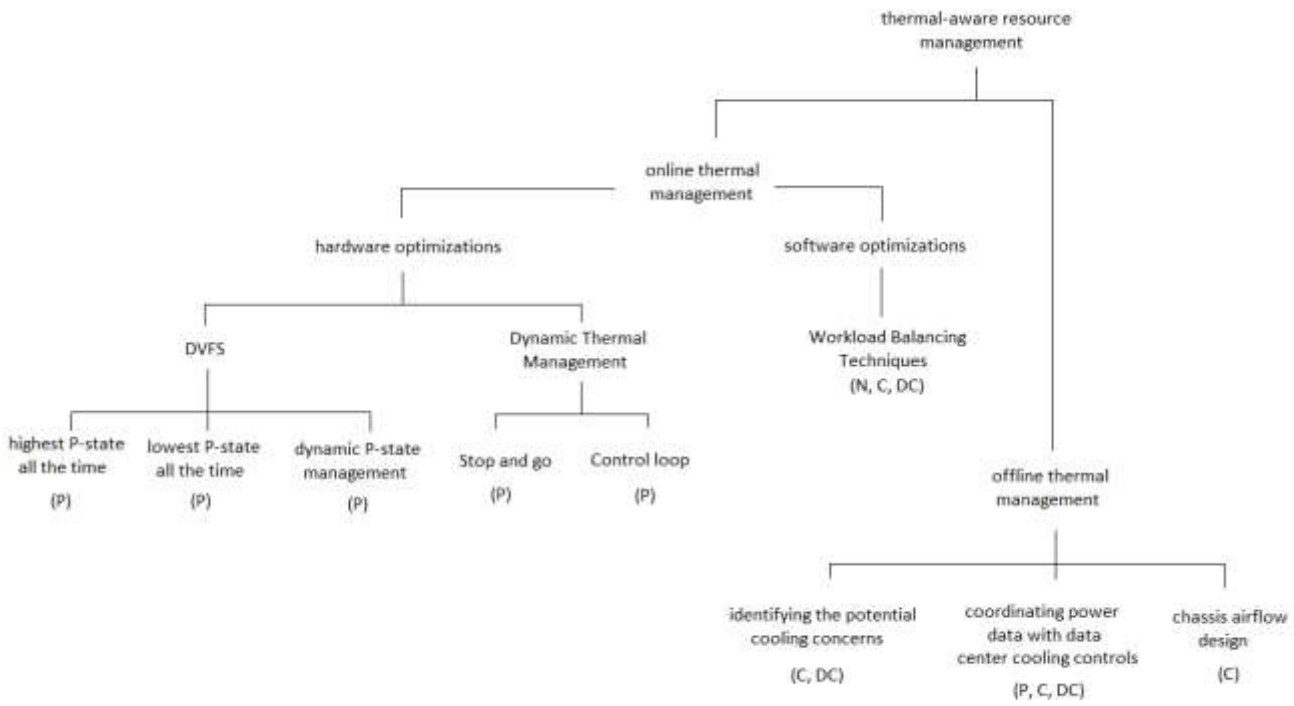

Figure 8 Classification of thermal-aware resource management techniques

2.1.3 Fans management

Recently, studies concerning the impact of fans located within the servers on the system performance and its power efficiency gained in popularity. In [38], the authors presented a fan controller that utilizes thermal models to manipulate the operation of fans. The main motivation behind their solution is to avoid the situation where the operation of a multiplicity of fans is driven by a single hotspot. Thus the controller considers all the fans of the server simultaneously. Taking into account the prediction of server temperatures, the presented solution adjusts the speed of particular fans proactively. The authors compare their results to the reactive fan controller, showing the reduction of fan energy consumption by up to 20%. Similarly, Kim et al. [39] proposed the fan speed control scheme allowing reducing the performance degradation and power usage up to 19%. They present Proportional-Integral-Derivative controller, which are immune to non-ideal temperature effects and then introduce the global scheme, which coordinates actions among multiple local controllers. Their solution is based on the set of sensors located inside a server supported with the power and thermal models used to determine trends in their changes. The authors validate a proposed solution using commercial servers and simulations. In [40], the authors studied the impact of fan speed on the disk throughput and finally on overall system performance. They analysed how vibrations induced by fans affect the hard disk bandwidth, resulting in a corresponding decrease in application performance. In the studies they follow similar thermal models as in the case of two, aforementioned works, however, extending them with the thermal coupling between CPU and memory. As a result, their solution tunes the fan speed with respect to the disk activity, while maintaining the given thermal constraints. Zapater et al. [41] studied the relationship between leakage and temperature of a server and provided the empirical model. To determine the model, the authors studied the behaviour of a server under power usage and varying fan speeds. Based on it, they designed a controller that tunes the fan speed to minimize the energy consumption for a given workload. The proposed approach adjusts the fan speed taking into account different utilization values of the system. The optimum fan speed values are stored in a dedicated table and are utilized by the fan speed controller at runtime. Another method [42] leverages both DVFS and fan management at the same time. Their idea is based on using the thermal resistance of a forced-convection heat-sink as a control variable (in the same manner as in case of voltage and frequency for the microprocessor). The proposed approach tracks the energy-optimal temperature as closely

as possible with a given workload, making the best trade-off between cooling power and temperature-dependent leakage power. Experimental results showed that the method helps in saving up to 17.6% of the total energy. Huang [43] proposed thermal-aware power optimization techniques that can be applied both on data centre and server level. At server level, the authors trade off fan and circuit leakage power by dynamically adjusting the server's thermal setpoint. Their solution tracks the changes in the server power and temperature for different values of the fan rotation speed, adjusting its thermal threshold toward an optimal fan speed that minimizes the overall system power. In this manner, the method lets the system to heat up when this saves more fan power than it costs in terms of leakage power. The total saving oscillates around 5%. Ayoub [44] noticed thermal dependencies between CPU and memory and non-linearity in cooling energy. Thus, he presented a holistic approach that integrates the energy, thermal and cooling management. The author designed a unified thermal and cooling model for CPU, memory and fan subsystems. Finally, the solution consists of a controller, sensors and actuators (on CPU and memory) that are activated depending on the thermal and cooling state of the system.

Contrary to the previous examples, both Moore [45] and Tang [46] skipped the impact of fans in their thermal considerations. Although they noticed the impact of airflow on the outlet temperature, they did not control the fan speed as a mean to improve energy-efficiency. They put their attention on reduction of heat recirculation, and thus improvement of cooling system performance by the opportunity in increasing the temperature of supplied air.

Figure 9 presents the classification of the presented fans management techniques.

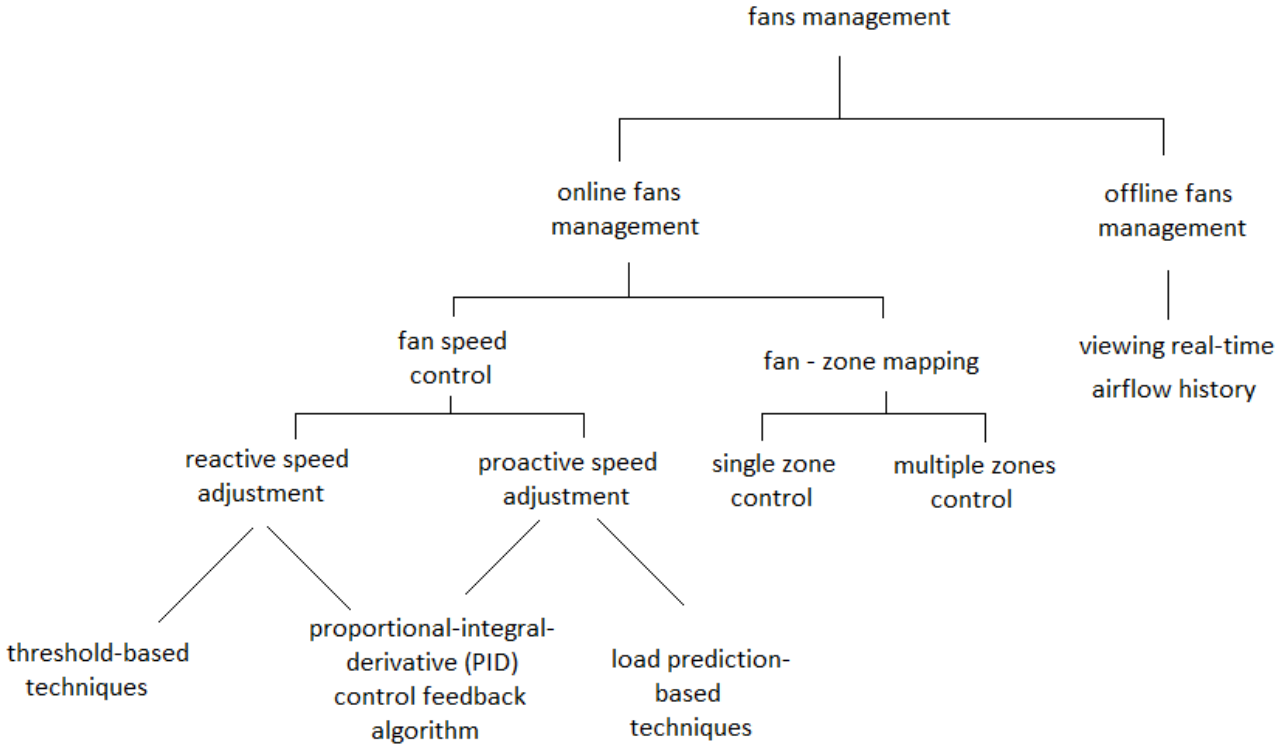

Figure 9 Classification of fans management techniques

2.1.4 Resource management in off-the-shelf servers

Making modern servers as power-efficient as possible is a great challenge for engineers. Current servers use various technologies to increase their overall power efficiency starting from more efficient cooling through

predefined power profiles/modes up to the power capping capabilities. And so HP ProLiant servers [47] come with built-in solutions optimizing internal cooling, managing power consumption and controlling its power usage. A dedicated setup utility allows configuring certain system hardware settings to lower its power consumption. First of all it supports managing processor idle states (C-states). User can choose the lowest power C-State for an individual processor at idle as well as the lowest power C-state for the entire processor when all cores are idle. Another feature, Power Regulator, allows optimizing processor power consumption based on the server activity. Power Regulator monitors processor activity and uses this information to adjust the processor power consumption. Depending on the chosen mode, it can keep processors in their maximum performance state (P-state), in lowest ones all the time or manage them dynamically, trying to reach the lowest level without significantly affecting performance. Moreover, user may select one of the predefined power profiles, which tries to find a trade-off between power consumption and system performance.

HP ProLiant servers support users in limiting the maximum of the server's power consumption to a given level by the means of Dynamic Power Capping technology. The predefined algorithm automatically reduces the server's power consumption in a controlled manner to keep it below the cap and adjusts caps of individual server in real time as workload changes. It can be achieved via pooling and allocating power to those servers who require it at the time due to heavier workloads or adjusting processor performance states in accordance with workload requirements.

In terms of cooling, ProLiant servers use more fans. Each fan in the server may have the outputs from multiple sensors mapped to it. Similarly, a given temperature sensor may also map to multiple fans. The result is a finely tuned system that provides more fan cooling only where it is needed without wasting power where additional cooling is not required. In addition, the fan control system also uses a control feedback algorithm to control the speed for each fan. It enables continuous adjustment of fan speeds to try to maintain a particular set-point temperature, which depends on the currently measured temperature, the history of temperature changes and predicted future temperature values.

The energy efficiency obtained in PowerEdge systems, manufactured by another vendor, DELL [48], [49], is done by the technology named DBPM (Demand Based Power Management). It utilizes DVFS mechanism to deliver power management options. The default policy manages resources with respect to the load running on the server trying to find an optimal trade-off between the power and performance. However, a user can also maximize system utilisation by choosing the performance mode or, on the other hand, disable "Turbo Boost" function to reduce the power consumption. Custom policies are also available allowing users to set and adjust various subsystem levels (CPU, memory and fan) to the desired value. DVFS is also utilized by DELL's power capping mechanism. In a static mode, a power cap is defined at single server level. In a dynamic one, the capping values are adjusted for each server in a group. In that case the management module raises the power caps of individual servers that are busier and need more power, while lowering the caps for the servers that are not using their total allocated power budget.

To improve cooling, fans inside Dell PowerEdge are separated into individual fan zones. Fans power and airflow consumption is saved by mapping these fan zones to specific components and sensors to provide airflow when and where it is needed. This allows only fans coupled to a component to react to component cooling requirements, which allows targeted cooling reducing fan power and system airflow consumption. To determine the appropriate fan response, control algorithm utilizes predictive calculations based on a measured process value compared against a target value. To achieve it, the Proportional-Integral-Derivative (PID) mechanism is applied.

2.2 Workload management policies

There are dozens of solutions for workload management in data centres with the objective of reducing the energy needed in operation. While most of them will be treated in the corresponding Deliverable D5.4 (Intermediate report about energy-aware workload management), some of these approaches also consider the cooling of computing systems and are therefore important preliminary work which has to be reconsidered in the development of the thermal management component. Thus, such solutions will be examined subsequently.

Liu et al. [50] model the data centre energy flows holistically in order to optimize IT workloads, cooling and power supply in conjunction. They integrate parameters such as renewable energy supply, dynamic pricing, cooling supply including chiller and free cooling modes, and IT workloads. Based on these data, forecasts of IT demand and renewable energy supplies are computed, which are then again used to schedule IT workloads and IT resource allocation. The problem is formulated as a convex optimization problem, finding an allocation of workloads to compute nodes minimizing the operational energy costs and maximizing the revenue from performing batch jobs. A runtime workload manager then schedules virtual machine (VM) migrations in order to get to the optimized allocation. While the workload can be allocated such that renewable energy usage increases by 39-63%, alternatively the workload may be scheduled cooling aware, which achieves energy cost reductions by 20-38%. Currently the workload management requires a fully virtualized server environment and is only applicable on standard x86 servers.

Another current example for proactive thermal aware workload management is presented by Lee et al. in [51]. By combining thermodynamic models and real-time measurements (temperature, humidity, air flow and workloads) heat generation as well as heat extraction is modelled as heat imbalance and forecasted in order to enable proactive management. The standard estimation error of the heat imbalance forecast model constitutes 0.866 or 1.6K. The models are then used as input for a heuristic addressing the VM allocation (variable-size multi-dimensional bin packing) problem, which is NP-hard. VMs are sorted by their deadlines/running times and allocated based on a multi-dimensional best-fit algorithm with the aim to pack longer duration VMs together, while considering the relation of estimated energy savings to migration costs and the adherence of thermal specifications as well as SLAs. Currently unused servers may then be shut off in order to save energy. Simulations yielded that the proposed algorithms save 7-10% more energy than the known heuristics Best Fit Decreasing and First Fit Decreasing, respectively, and in regard of thermal-aware algorithms the savings are 12% higher. Analogous to the workload management by Liu et al. [50], the proposed approach is currently only applicable in a fully virtualized x86 server environment.

Villebonnet and Da Costa's cloud middleware centric approach [52] has the target to optimize server utilisation while avoiding hot spots in the server rooms, so that there are no peeks in cooling demand. Like in the aforementioned examples, workload allocation is done by migrating virtual machines. Therefore the cloud management platform Snooze is used to integrate their algorithms. In general, the current temperatures and workloads are monitored and migrations will be triggered when certain thresholds are exceeded. Next to the limitation of relying on fully virtualized server environments, this work also has drawbacks due to being static by using constant thresholds and only measured data without forecasts.

Although there are several approaches to proactive and reactive thermal-aware workload management, none of the work looked into examined management of diverse, alternative compute modules such as ARM based processors or FPGAs. While the pure algorithms may be adaptable, especially the virtualisation based solutions need a strong revision in order to be applicable for such novel compute nodes like in M2DC.

3 Interaction between resource and workload management policies

The main goal for resource management techniques within M2DC is to lower power usage and heat dissipation. However, there is a strong correlation between the resource management strategies and workload management ones. As one may expect, simple fans and cooling control does not influence the workload execution, but the actions taken at the processor or node level may significantly affect the performance of running applications. This section discusses the interaction between resource and workload management policies.

In general, resource and thermal management policies will take into account sensor readings and the information coming from the server monitoring tools. These data will be investigated and applied to power and thermal models at the same time. Additionally, external user requirements such as power capping or outlet temperature/flow constraints will be taken into account. Resource and thermal management, as depicted in Figure 10, consist of three functional modules: Fan Manager, Power Capping Manager and Energy Saver Manager. The Fan Manager is responsible for adjusting fan rotation speed, while the Energy Saver Manager controls the state of computing resources to reduce their energy consumption (Power Management). Above them, the Power Capping Manager takes care of keeping the power usage draw below the given threshold. Based on current measurements and predicted future trends, resource management actions will be taken.

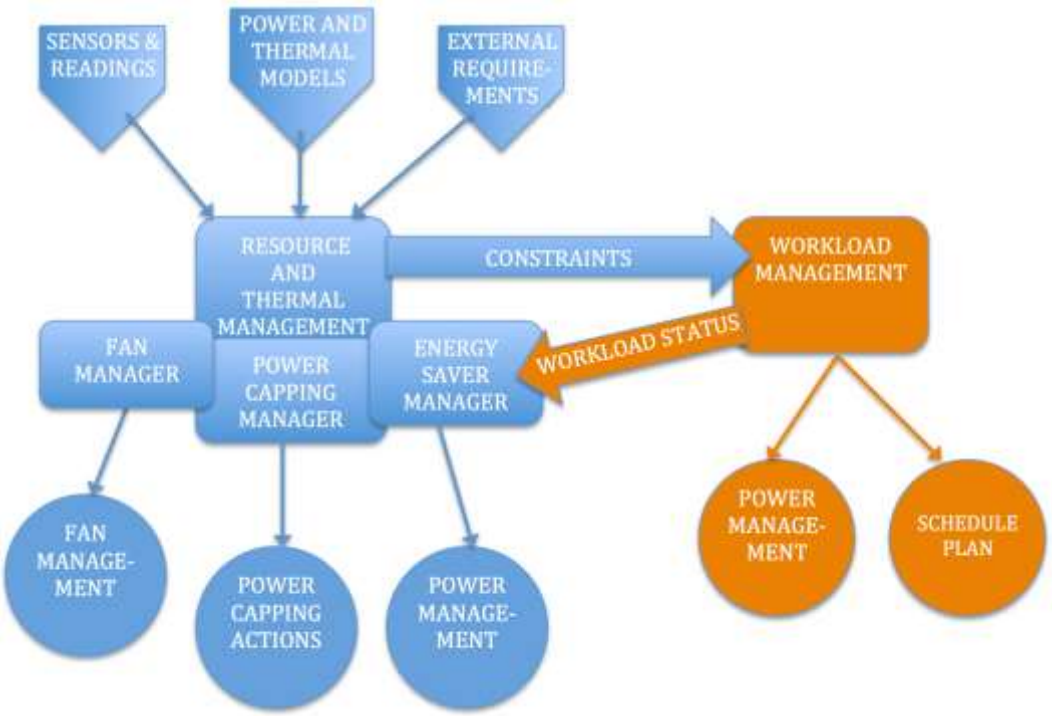

Figure 10: Interaction between workload and resource & thermal management

Independently, these data will be passed as constraints to the workload management module, which also wants to manage/switch hardware nodes (Power Management) and will deploy applications based on thermal constraints as well as current and future workload. As a feedback, the Workload Management will return the status of the nodes in terms of their current and future workload assignment, which might be additionally used by Energy Saver Manager. The Workload Management module decides about the application's assignments and balances the load between nodes according to the given constraints. However, it may also manage power of hardware nodes concurrently and independently to the Resource and Thermal Management. In case of using the Workload Management module to manage the power states of computing resources, a user may disable the Energy Saver Manager relying only on the Workload Management power policies.

Although the resource and thermal management shares its optimization target of reducing the energy demand of IT with the workload management, the control mechanisms are differing from each other with the exception of switching the power states, as depicted in Figure 11. Both management tools want to change the power states of compute nodes like switching nodes on or off, or putting them into power saving states, but the decisions are based on different properties. In the case that both management tools are allowed to control the nodes' power states, you have the problem of potentially conflicting control decisions due to different optimization criteria, i.e. the thermal management wants to switch off nodes the workload management intends to use for application deployment and vice versa.

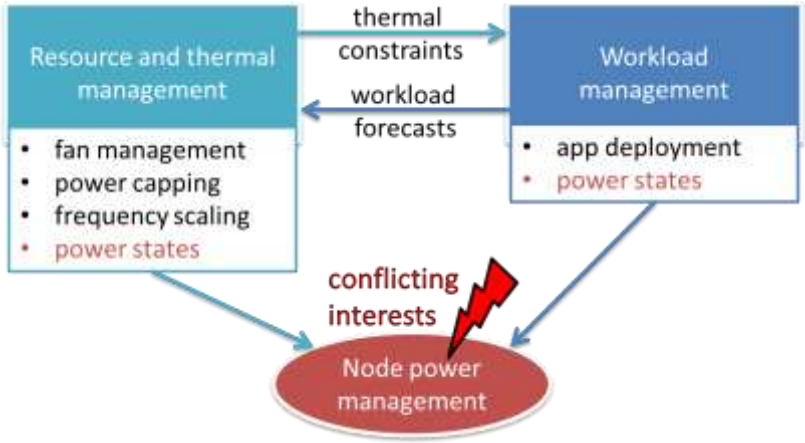

Figure 11: Node power management accessed by resource and workload management leads to conflicting optimization interests

These potential conflicts require some coordination between the management tools. Therefore, an interface between thermal and workload management will be implemented, which enables the exchange of specific management data, i.e. thermal constraints and workload forecasts for each compute node. This way, the allocation algorithm of the workload management is able to consider the thermal environment for its node selection and application scheduling, so that not only the IT is operated energy-efficiently but also the waste heat can be negated efficiently. If the user wants to use thermal and workload management in parallel, the recommended workflow would look like presented in Figure 12. The thermal management would decide about fan management, power capping and frequency scaling independently. However, the switching of node power states will be delegated to the workload management, which then uses provided information about the thermal environment in addition to the workload data. In case, the workload management is not needed or already done by other pre-existing tools, the thermal management is still able to access the node power management module in order to switch nodes on or off based on thermal behaviour.

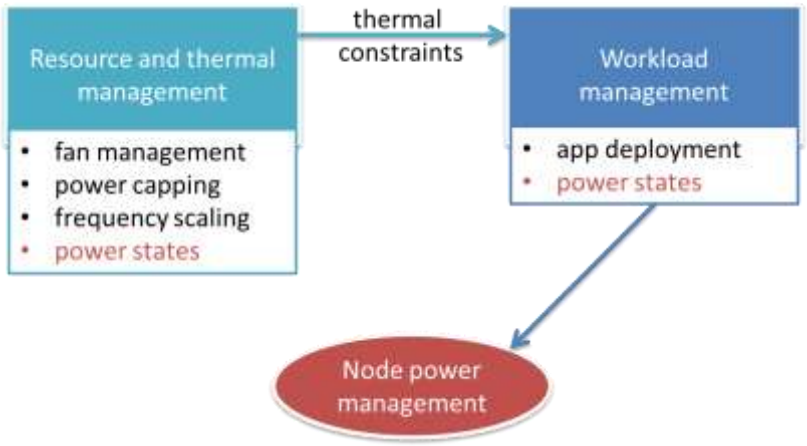

Figure 12: Workflow for parallel usage of thermal and workload management

4 M2DC power and thermal management interfaces

The main goal of M2DC's management is about saving energy. For this purpose it relies on Intelligent Management, consisting of two subsystems: Workload Management as well as Resource and Thermal Management. While the Workload Management deploys workload in a way that compute nodes are used more efficiently, the Resource and Thermal Management takes care about heat dissipation of nodes and corresponding fan management (see Chapter 3). As the management subsystems need to make rational decisions in complex situations, they require a lot of information about their soft- and hardware environments. This chapter describes interfaces used for data exchange between different M2DC components, which are on software layer as well as on hardware layer.

4.1 *Support on hardware layer*

The hardware and corresponding firmware of the RECS®|Box architecture provides various opportunities for resource and thermal management of its components. As described in Section 3.3 of Deliverable 1.3 "Hardware platform and system efficiency enhancement design specification", the RECS®|Box management system consists of multiple microcontrollers distributed over the baseboards and backplanes that make up the system. These microcontrollers are connected to a master controller on a chassis controller board, which manages the whole unit and provides interfaces to other software layers and external third-party software.

For this, the chassis controller will provide an IPMI and a Nagios NRPE interface as well as a restful API. In addition to that, a web GUI can be utilised to monitor and control the RECS®|Box manually. This section focuses on the interfaces that can be used programmatically and therefore lets the web GUI aside. The most complete interface for external control will probably be the restful API with the IPMI interface implementing a subset of the available functionality. A first version of the REST API was already specified in D1.3. The API will later be extended to support more advanced functionality of the M2DC servers.

Components of a RECS®|Box that can be externally controlled are the compute nodes, chassis fans, network infrastructure and the KVM system. The management system inside the RECS®|Box normally autonomously controls the power supplies depending on the power budget of the currently turned on compute nodes and the rest of the infrastructure. However, for development of the middleware, control of the power supplies can also be added to the restful API. The power supplies inside the RECS®|Box have their own fans which are controlled by each PSU on its own. These fans can only be monitored, but not controlled. In contrast, the chassis fans may be individually controlled by the restful API with speeds between 0 and 100 % of their maximum RPM. Compute nodes can be turned on or off, can be reset and selected for KVM.

The monitoring part provides readings from temperature sensors spread over baseboards and backplanes, calculates power consumption of nodes and infrastructure, and monitors the speed of the installed fans. The COM Express baseboards will also monitor some of the status signals from the CPU modules, e.g. system power state (S0-S5). The power supplies can be monitored as well and provide their current load, voltage, temperature, fan speed and other health indicators.

In addition to the sensor data provided by the hardware, a RECSDaemon can be installed in the OS running on a node to gain access to all sensor values that are provided by the OS such as CPU and memory usage. The RECSDaemon is also able to execute commands sent by the management system on the node (e.g. for shutting down the OS gracefully). The RECSDaemon is extensible by a plugin system to add further sensors or different commands to be executed. The daemon might be used to influence power capping capabilities of the installed OS.

Monitoring/Control item	**Controlled by**	**Available APIs**
Chassis fan speed (control)	RECS_Master	REST API
Chassis fan speed (monitor)	RECS_Master	REST API, IPMI
PSU on/off	RECS_Master	(REST API)
PSU load	RECS_Master	REST API

PSU fan speed (control)	PSU firmware	-
PSU fan speed (monitor)	PSU Firmware, RECS_Master	REST API, IPMI
Node on/off/reset	Baseboard firmware, RECS_Master	REST API, IPMI
Node KVM select	Baseboard firmware, backplane firmware, RECS_Master	REST API, IPMI
Node power consumption	Baseboard firmware, RECS_Master	REST API, IPMI
Node power state (S0-S5)	Baseboard firmware, RECS_Master	REST API, IPMI
Temperatures	Baseboard firmware, backplane firmware, RECS_Master	REST API, IPMI
Voltages	Baseboard firmware, backplane firmware, RECS_Master	REST API, IPMI
OS sensors	RECSDaemon, RECS_Master	REST API
OS commands	RECSDaemon, RECS_Master	(REST API)
Switch configuration	Baseboard firmware, backplane firmware, RECS_Master	REST API

Table 4-1: Monitoring/control items and available APIs to influence them

4.2 *Components and Interfaces on Software Layer*

Decisions of the Intelligent Management are based on several factors, collected in different parts within the M2DC environment. The hardware provides the most important part of information through a restful API of the RECS_Master, including temperature values, fan speed, hardware usage or power consumption (compare Deliverable D1.3 “Hardware Platform and System Efficiency Enhancement Design Specification”). It is also possible to extend the RECSDaemon in order to control power capping functions implemented on the microserver (see Section 4.1). Besides, the running use case applications have to be taken into account too, e. g. regarding the length of task queues or task priority. Figure 13 gives an overview about components and interfaces required by the Intelligent Management, consisting of the Workload Management on the one hand, and the Resource and Thermal Management on the other hand. In addition to the aforementioned applications and the RECS_Master, there are connections to the OpenStack services Nova and Ceilometer. Nova is used to manipulate the schedule of used nodes or containers, while Ceilometer supports on getting further information about the system environment.

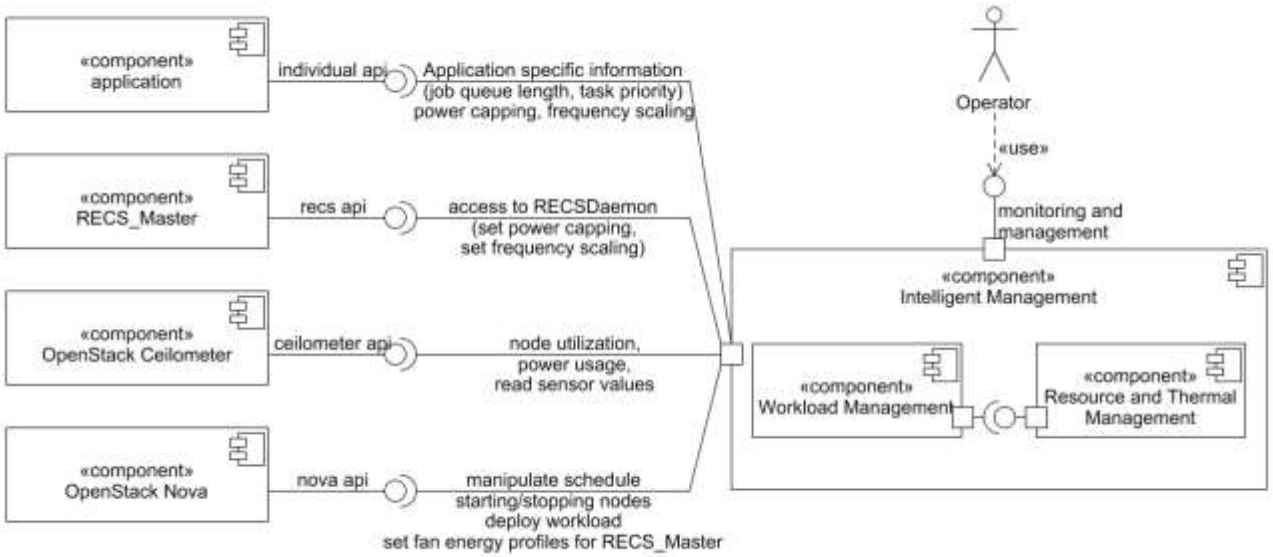

Figure 13: Interfaces between Intelligent Management and other components

Figure 14 explains the Intelligent Management composition in more detail. Currently, the Workload Management and the Resource and Thermal Management are two independent systems from different partners. As they will be combined within the M2DC project, jointly used components were identified in a first step. A data collector will fetch all necessary information from the appropriate sources and make them persistently available in a database. Furthermore, Power Models and Thermal models will be made available. Both the Workload Management and the Resource and Thermal Management analyze the collected data and calculate energy optimizations, for example turning compute nodes on or off. As they may work on the same data source, but pursuing different goals, it is important to prevent both components from executing their suggestions against each other. An interface between both components will enable an exchange of any constraints, preventing opposing behavior.

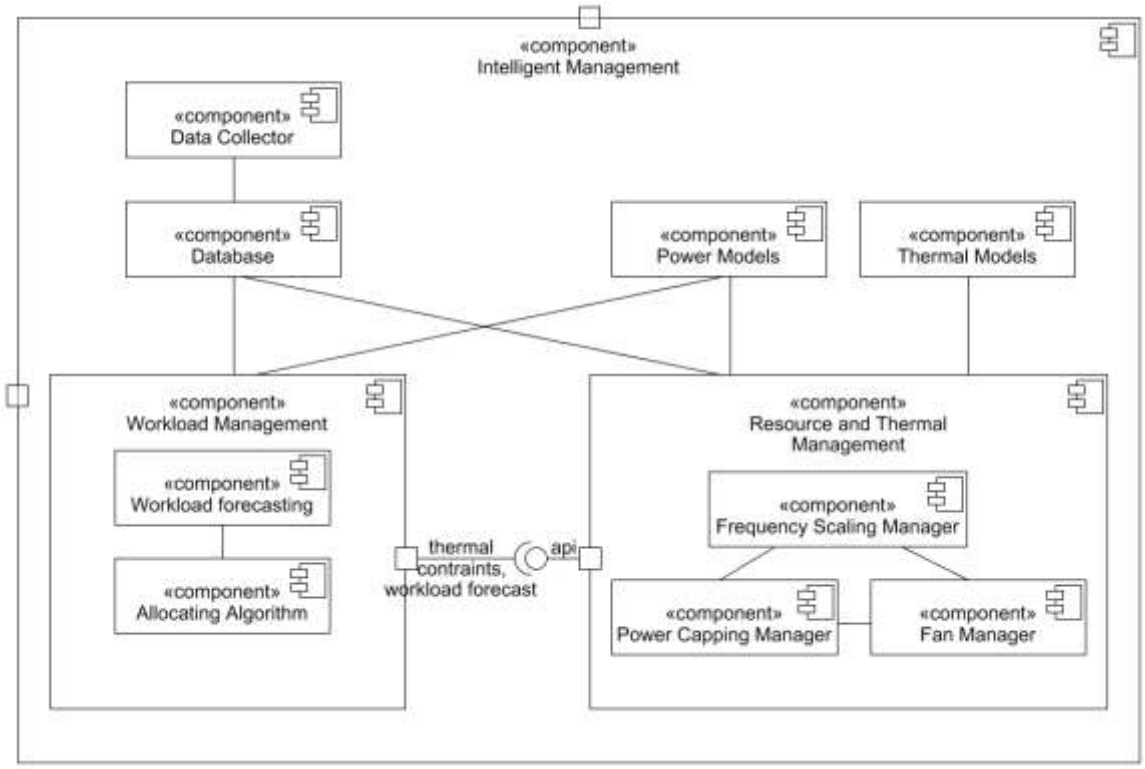

Figure 14: Interfaces within Intelligent Management

Fetching all important data and sensor values for the Intelligent Management would require several interfaces across different system layers. To keep the total amount of connections between components rather low, M2DC's approach is to exchange as much information as possible through the central OpenStack layer. That means, OpenStack reads e.g. sensor values from the RECS_Master API and provides all information through appropriate APIs to the Intelligent Management. The overall approach for integrating the middleware architecture including the hardware layer, OpenStack and the management components is discussed and presented in Deliverable D1.4 "Middleware specification".

5 Power and thermal models

This section presents the overall description of RECS®|Box together with corresponding power and thermal models of microservers that can be placed within it.

5.1 *RECS®|Box concept*

Description of the RECS®|Box **concept:**
The M2DC server is an efficient resource and highly scalable heterogeneous platform that combines a large number of microserver modules in a single chassis. The main target of MD2DC box is to achieve the best performance versus consumption, through the use of thermal and power management associated with resources partitioning and SEE (System Efficiency Enhancement) accelerator. In addition, the M2DC will offer a high level of reliability and maintainability. It will feature hot-plug capabilities (PowerSupply, Fan, Baseboard) and offer some redundancy.

Highly scalable and modular architecture:
As computational resources, this server uses dedicated baseboards and a modular communication backplane. For flexibility, the design of the chassis enables hot-plugging of microserver baseboards. There are two different types of baseboards working within the M2DC server, the High-Performance and the Low-Power Baseboards. The **low power ARMv8 microserver (LPM)** baseboard is composed of two printed circuit boards. Two standards of LPM will be used in the MD2DC server. The first is based on Toradex Apalis [53] small form factor microservers existing in two variations with a SoC integrating an ARM architecture (ex: Tegra SoC) or with an FPGA SoC including a second 1G Ethernet interface (ex: Zynq Soc). The second standard of the LPM carries a NVIDIA's Tegra X1 microservers. The **high performance** microserver baseboard uses one PCB. It can host up to three high performance microservers (COM Express Basic form factor) and features a 10 Gbit Ethernet infrastructure as well as low-latency high-bandwidth communication. Three types of **high-performance** microservers have been selected. Firstly, the **high-performance x86** microservers mainly based on Intel's Skylake architecture, the newest x86 architecture having the Intel's 14 nm process which increases energy efficiency and processing power significantly. Secondly, the **high-performance ARMV8** microservers based upon integration of low power technology and CPU architecture principles from the mobile market. Several processor manufacturers have developed server processors based on that technology (for instance Cavium with the ThunderX based upon own cores CPU cores). And lastly, the **high-performance FPGA based** microservers. Due to their parallel architecture, FPGAs are well suited for energy efficient acceleration of computing tasks. The selected candidate for the M2DC project is the Stratix 10SoC from ALTERA with 1100kLEs integrating a hard 64 bit quad-core ARM Cortex-A53 processor. The server architecture also includes several backplanes which number depends on the chassis type. The backplane routes the communication inside the server. Three baseboards can be connected per backplane. There are 2 types of backplane, the mandatory backplane is used for Ethernet/Management/KVM networks. This backplane has 40 GbE on it to guarantee high bandwidth for internal and external communication. An optional backplane is also available that includes some crosspoints for Low Latency High Bandwidth and PCIe switches. It is optional because it is only used by High Performance base boards. The backplanes are connected together via a proper switch in the backpanel. This board also permits multi rack unit applications. The whole communication infrastructure is **scaled across rack level** by using the connectors located at the backpanel.
The architecture of the server also includes 2 different and independent levels of communication that provide **scalability** in the M2DC server. The first level of communication is the **Ethernet-based network**, with 10 Gigabit Ethernet. Applications can use this 10 Gigabit Ethernet infrastructure for fast communication (DATA). This Ethernet network is connected to a 10 GbE Network switch on the baseboards. The second level is the **Low-Latency High-Bandwidth communication**, which is based on High-Speed Serial Transceivers and PCIe Switching. This communication infrastructure can be used in a flexible manner supporting various use cases (ex: multi-host communication). It allows attaching an accelerator to any node within the server. A run time topology can be reconfigured and accelerators can even be attached together.

Monitoring and control infrastructure (MCI):
From the hardware perspective within a single rack, a large number of sensor values are continuously monitored and transmitted to a dedicated **monitoring and control** infrastructure (MCI). It will provide all relevant parameters coming from thermal, voltage, current sensors, fan speeds, processors and operating systems activities (CPU usage, RAM utilization, …) to the power and management process. In addition to that, the MCI enables several actions like having a controlled system startup and shutdown (i.e. turning any node on or off), setting the fan speed via the use of PWM signals, the configuration of integrated switching structure (ex PCie lanes) and in general to avoid any hotspot in the chassis. To achieve this, several management microcontrollers are implemented in the backplane, baseboards and the chassis controller. This last one controls the complete chassis by running the main management software. In addition, the power supply management bus (PMBUS) is also connected to the chassis controller. The set of microcontrollers will communicate via dedicated buses namely I²C and a higher bandwidth communication link. Hence associated with the SEE acceleration applications, this infrastructure permits a precise level management of power, temperature and performance of the server. More details concerning the usage of microcontrollers and SEEs in terms of M2DC middleware can be found in Deliverable D1.4.

Physical architecture of the chassis:
All versions have common characteristics:
- Compatible to standard 19-inch racks, 482.6 mm wide
- Standard air flow direction from front to rear
- Hot-swap power supplies
- Retractable Baseboards are inserted in a 9 HP slot (45.72 mm), hot-swap
- Chassis controller that will run the main management software to control the chassis.
- Back side: all other ports, hot-swap redundant power supplies

There will be three different chassis available:
- The small chassis version has
  - Dimensions: 1RU height, max 800 mm length
  - 3 microserver baseboards without direct communication between each other.
  - Baseboards are oriented such that it provides the best forced convective efficiency.
  - 2 power supplies including a hot swap redundant element
  - 4 fans (45mm diameter) are used to provide a sufficient horizontal flow rate

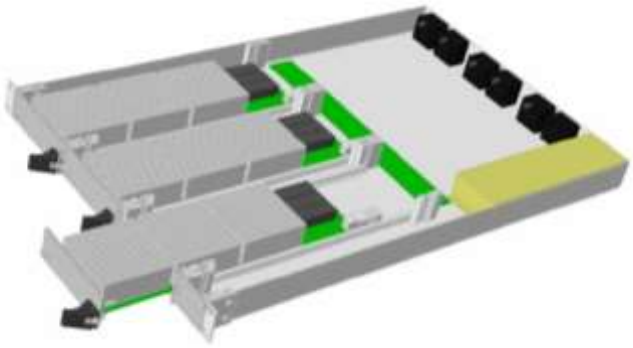

Figure 15: small chassis with 3 microserver baseboard slots

- The mid-Range chassis version has
  - Dimensions: 3RU height, max 800 mm length
  - Front side: 9 microserver baseboards
  - 2 horizontal Backplanes.

- Baseboards and Backplanes are oriented such that it provides the best forced convective efficiency.
- 4 power supplies including a hot swap redundant element
- 3 fans (92mm diameter) are used to provide a sufficient horizontal flow rate.

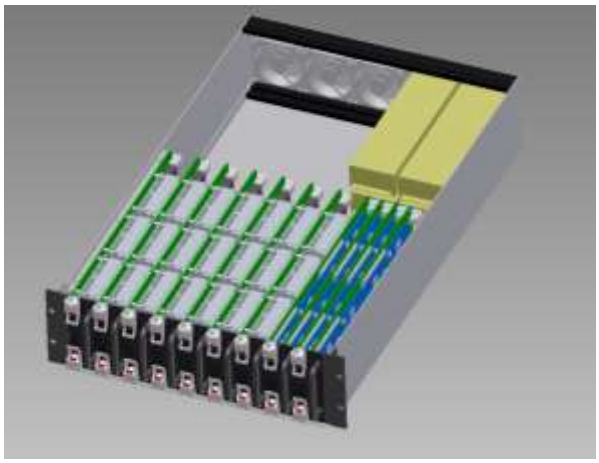

Figure 16: standard chassis with 9 microserver baseboard slots

- Scale-out chassis version has
- Dimensions: 3RU height
- Front side: maximum 9 microserver baseboard slots of 9 HP width
- Back side: maximum 6 microserver baseboard slots of 9 HP width

Cooling system to be defined

5.2 *Power models*

This section presents power models of microservers and other computing units that can be integrated into the RECS®|Box. The RECS®|Box is designed for integration of a variety of microservers compatible with the COM Express interface as well as additional accelerators via PCI Express links. Hence, as the full list of compatible units is long and open, we concentrate on selected examples. The most important ones are FPGA-based microservers, ARM64-based microservers (limited to non-confidential data), GPUs (both microserver versions and large accelerators connected via PCI EXpress), and selected models of Intel CPUs.

5.2.1 Power model description of the x86 CPU modules

In case of the x86 and x86-64 family processors the P-States and C-States are exposed through a standardized interface specified by the Advanced Configuration and Power Interface Specification (ACPI). It allows the software to switch the CPU P-States (ACPI Performance States) based on the amount of workload. Each P-State consists of a frequency and associated CPU core voltage (V_{core}). Modern operating systems have built-in support for frequency and voltage scaling, dynamically adjusting the CPU frequency based on process scheduling decisions. The P-State control is split between the decision making part taken by the OS deciding on a target P-State, and the control algorithms implemented in the processor which are responsible for migrating to the target P-State. The maximum performance P-State is called P_0 and each subsequent state P_n is characterized by less performance and/or smaller frequency than P_{n-1}.

The power consumption of a modern CPU is given by the Ohm's law:

$$P = C * V_{core}^2 * f$$

Where C is the processor switching capacitance, V_{core} the current P-State's core voltage and f is the frequency. Given the above equation and the fact that both f and V_{core} decrease with the increase of the P-State, the power consumption should decrease in a cubic manner with the increase in P-State. However, it is impossible to decrease the V_{core} below the CPUs base operating voltage, thus higher P-States only decrease the frequency.

Additionally, the Intel Turbo Boost Technology implemented in Intel processors allows them to operate at a power level that is higher than the Thermal Power Design (TDP) configuration and data sheet specified power, if power and thermal budget are not exceeded under given conditions.

Next to P-States, processors are also equipped with C-States. A C-State, also called idle or sleep state, is representing the depth of the power conservation in which the CPU is currently residing. The normal operating mode is C_0 – fully turned on, with each subsequent C_n state offering deeper power savings than the previous state $C_n - 1$ at the expense of increased wake-up delays. The most energy-efficient is C_6 – Deep Power Down. It reduces the CPU internal voltage to any value, including 0 V. While P-States can impact the performance of the work performed by the CPU, C-States are entered when the CPU is idle. P-States and C-States may operate independently of each other. C-States are controlled by the OS scheduler which knows exactly when a core's performance is required.

Below, we present power profiles of a few Intel processors, i.e. the power consumption of the processor in relation to the load exerted on it and/or the frequency. As noted earlier, higher P-States only decrease the frequency of the processor. Based on the experiments, we confirmed the almost linear dependency between the power consumption and the processor frequency. We may therefore reformulate the previous equation in the following way:

$$P = a * f$$

Where a is the processor specific constant and f is the frequency.

Modern x86 and x86-64 family processors have different technologies implemented to execute the calculations in the most energy efficient manner. The most prevalent are P-States and C-States. These are also implemented in the M2DC x86 processors available on COM Express Modules: i7-6820EQ, i5-6440EQ, i3-6100E, E3-1505M V5 and E3-1505L V5. A short specification of these processors is the following:

- i7-6820EQ: 45 W TDP, idle power with C-States enabled: 1.4 W, idle power with C-States disabled: 2.6 W, throttling frequency between 2.80 GHz and 3.50 GHz,
- i5-6440EQ: 45 W TDP, idle power with C-States enabled: 1.5 W, idle power with C-States disabled: 2.5 W, throttling frequency between 2.70 GHz and 3.40 GHz,
- i3-6100E: 35 W TDP, idle power consumption 1.2 W, throttling frequency between 2.70 GHz and 3.40 GHz,
- E3-1505M V5: 45 W TDP, idle power consumption 1.5 W, throttling frequency between 2.80 GHz and 3.70 GHz,
- E3-1505L V5: 25 W TDP, idle power consumption 1.3 W, throttling frequency between 2.00 GHz and 2.80 GHz.

Since the processors listed above were not purchased yet, we present below power profiles of Intel processors similar to the ones used in the M2DC project. Due to the similarity of the architecture of those processors, similar conclusions may be drawn in their case.

i7-3615QE power profile

Intel Core i7-3615QE is a quad-core processor with Hyper-Threading Technology enabled. The TDP of this processor is 45 W. The available frequency range is between 1200MHz and 2300MHz, plus the optional increase by the Turbo Boost Technology. In this state the processor frequency dynamically increases, based on current workload, until the upper limit of frequency is reached, which in case of this CPU is 3300MHz. The power consumption of this processor in relation to the throttling frequency is presented in Figure *17*.

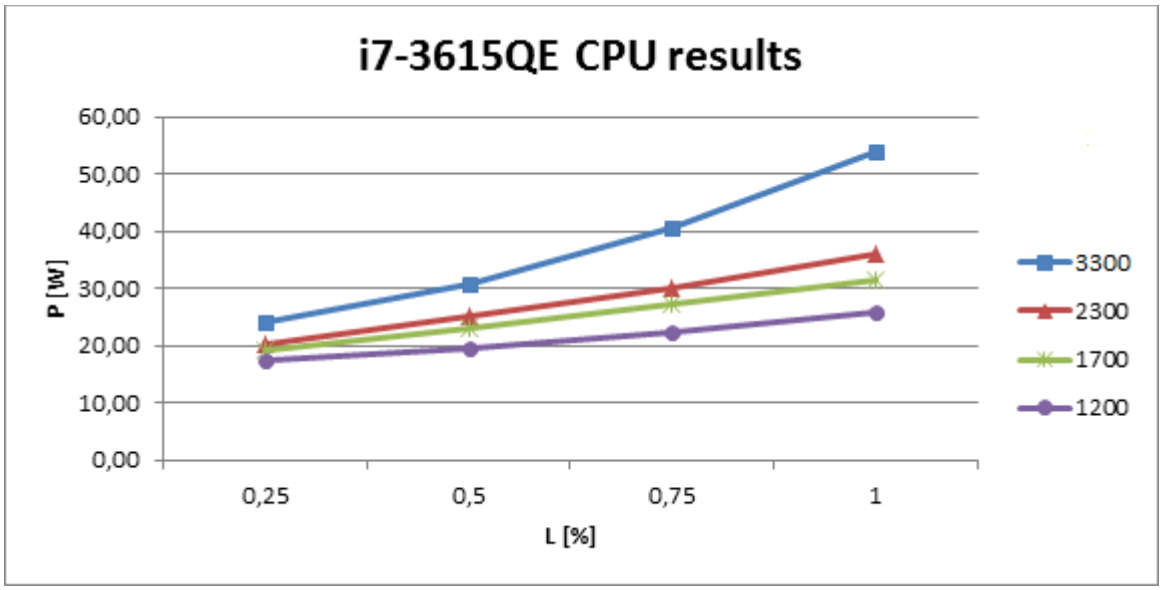

Figure 17: Dependency between power usage and CPU load for different clock rates on Intel i7 3615QE processor

The relation between load and power on this CPU is close to linear. Easily visible is a significant growth of power usage for the highest CPU frequency. It happens because of the Turbo Boost Technology being activated

in this state. Intel Core i7-3615QE reaches a maximum value of about 35W under heavy load without Turbo Boost and more than 50W with the highest available frequency value.

Similarly, Figure *18* presents the relation between CPU frequency and power usage for various loads on this processor. It is also close to linear. Clearly visible is a high peak in case of Turbo Boost mode frequency, but it is important to remember that the real frequency of the CPU is much higher (up to 3300MHz). Unfortunately, it was not possible to dynamically probe its value at runtime.

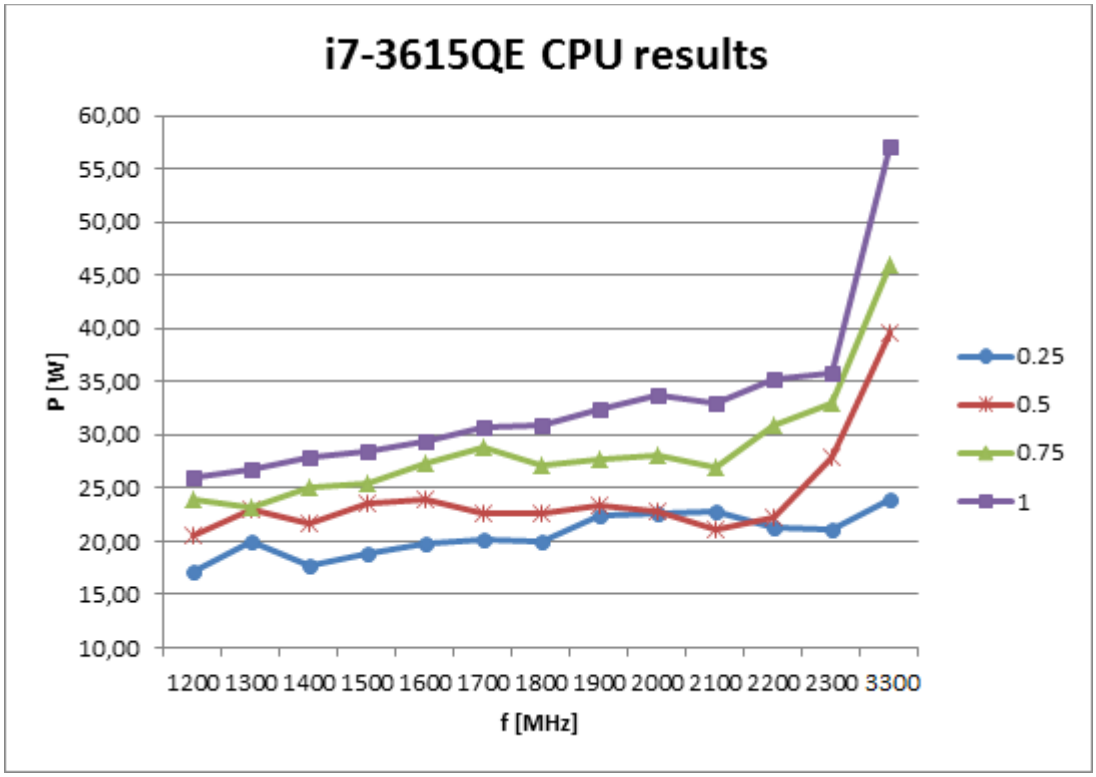

Figure 18: Dependency between power usage and clock rate for different CPU loads on Intel i7 3615QE processor

i7-2715QE power profile

Intel Core i7-2715QE is a similar processor operating at a maximum frequency of 2300MHz (or 3000MHz in case of Turbo Boost Mode). Its TDP is also 45 W. It was built in a 32nm technology, as opposed to 22nm in case of i7-3615QE. The available CPU frequency range is different than in the previous CPU – between 800MHz and 2100MHz, reaching 3000MHz in Turbo Mode).

The dependency between load and power consumption is presented in Figure *19*. The power consumption increases with higher load, although dependency is not linear. Some disturbances appear, probably caused by changes in the ambient temperature (the temperature may also affect the power consumption of the processor). The results are similar to the previous i7 model, although almost all of the results are a bit lower for 2715QE (by up to 4W).

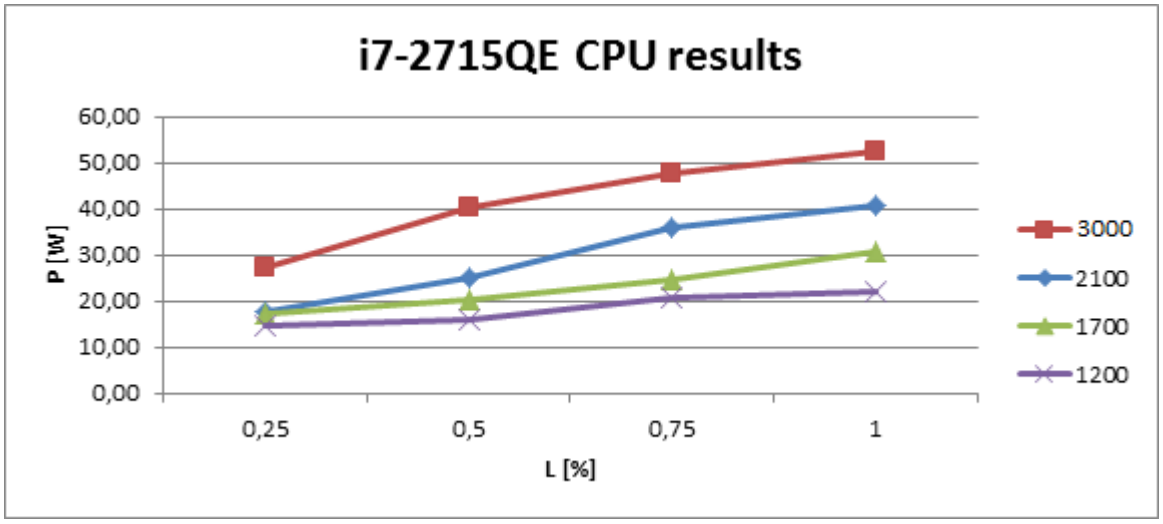

Figure 19: Dependency between power usage and CPU load for different clock rates on Intel i7 2715QE processor

Similarly, Figure *20* presents the dependency between the power consumption and the CPU frequency for different processor loads.

Similar power profiles will be achieved in case of other x86 processors available on COM Express modules: i7-6820EQ, i5-6440EQ, i3-6100E, E3-1505M V5 and E3-1505L V5.

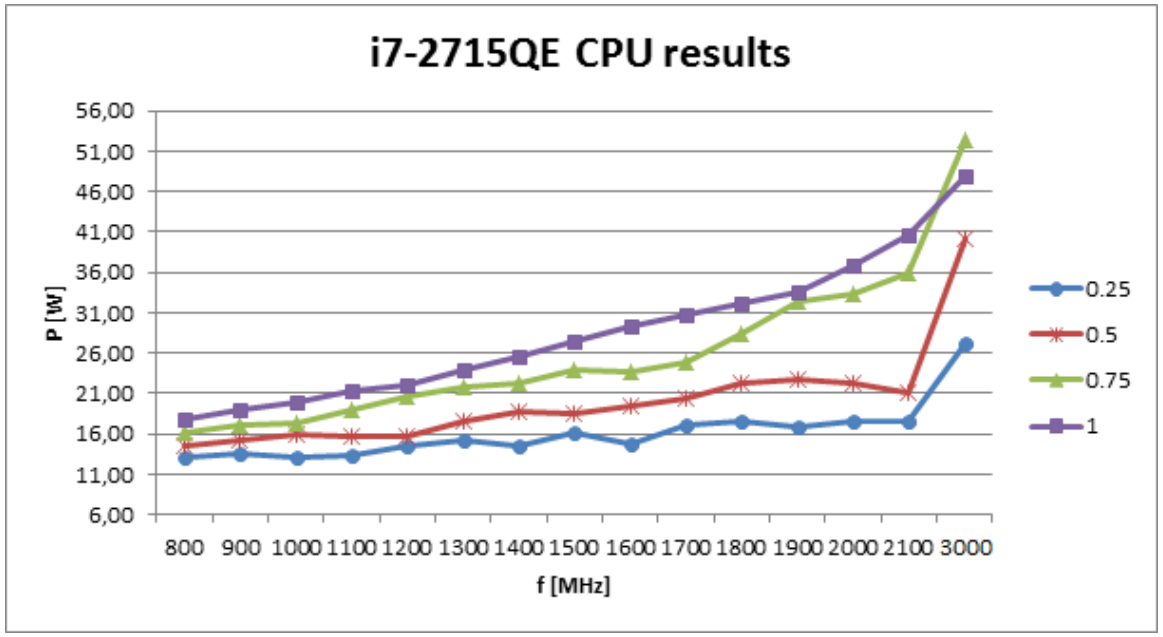

Figure 20: Dependency between the power usage and the clock rate for different CPU loads on Intel i7 2715QE processor

5.2.2 Power model description of ARM microservers

ARM-based CPUs are supporting both *idle management* (switching off pieces of the chip if not needed, ranging from sub-modules to almost the whole device, similar to the C-states in x86) and *dynamic voltage and frequency scaling* (DVFS, nowadays in general not available at the early ARM-64 based server CPUs). Those options are implementation-specific, resp. tied to a certain version of the ARM core.

For example for the power management options for a Cortex-A57 core are described here [54].

As ARM-64 server CPUs are still in the prototyping and integration phase interfacing to the power management is still under development. Goal is to abstract it – similar to x86 – via ACPI, in particular via the optional ACPI methods _PS0, _PS1, _PS2, and _PS3; in ACPI, _PS0 is the method to invoke to turn a device full on, and _PS3 is for turning a device full off. It is the goal to abstract the complexity and platform-specific parts via the firmware. In parallel the Power State Coordination Interface (PSCI) [55] allows core idle management, dynamic addition/removal of cores, big.LITTLE migration and system shutdown/reset.

The concrete power profiles for the high-performance ARM64-based server CPUS are not available, resp. confidential.

5.2.3 Power model description of GPU modules

Like CPUs, modern GPUs also have sophisticated power management. The power consumption and the performance capability of a device are primarily determined by P-States. For typical GPUs, P-States range from P0 to P12, where P0 corresponds to the highest performance/power state, and P12 corresponds to the lowest performance/power state. Note that not all P-States are available on all devices. When a GPU is active with a workload, the NVIDIA driver will continuously adjust the P-State to deliver best performance while matching the performance state to the actual workload. Given that no thermal/power limits are violated, the P-State should reach its highest level (P0) for the most active and heaviest, continuous workloads.

It might however happen that the frequency of clocks is reduced. This usually happens due to the following reasons:

- nothing is running on the GPU and the clocks are dropping to idle state,
- GPU clocks are limited by applications clocks setting,
- SW Power Scaling algorithm is reducing the clocks below requested clocks because the GPU is consuming too much power (SW power cap limit can be adjusted),
- HW Slowdown is engaged because of one of these reasons:
    - temperature is too high,
    - External Power Brake Assertion is triggered,
    - Power draw is too high and Fast Trigger protection is reducing the clocks.

The software power capping is an interesting feature of Kepler-based and newer GPUs. It allows for setting the maximum power a GPU can draw at any time. Using this mechanisms, administrators or server management software may keep the server within a given power budget.

The user may also set manually the desired frequency of graphics (shader) and memory clock. Moreover, an application may request so called Auto Boost mechanisms to increase the performance, in which case the GPU will opportunistically boost to higher clocks when power, thermal and utilization headroom allows.

Additionally, to save power the GPU may be operating in various modes by disabling selected subsystems:

- in "All On" mode everything is enabled and running at full speed,
- the "Compute" mode is designed for running only compute tasks. Graphics operations are not allowed,

- the "Low Double Precision" mode is designed for running graphics applications that do not require high bandwidth double precision.

An important issue is also the GPU cooling. The server GPUs are usually passively cooled which means that chassis fans need to cool down the device whenever needed. This means, in turn, that the server management tools need to monitor the GPU temperature information and react accordingly.

RECS|Box 4.0 also enables integration of GPU modules. Two general options are possible:

- As a microserver deployed on the board using Apalis or NVIDIA Jetson TX1 form factor (embedded GPU modules)
- As a standalone accelerator connected via PCI-Express (typical, large GPU modules)

Example GPU modules that can be integrated with RECS|Box 4.0 within M2DC are given below, starting from the embedded ones.

NVIDIA Jetson TX1

The NVIDIA Jetson TX1 is based on a custom form factor with a 400-pin board-to-board connector which will be supported by M2DC servers. Importantly, the pin-out will be backward-compatible with future versions of the Jetson module, which makes it an interesting platform for future integration.

Jetson TX1 uses 1 watt of power in idle state, around 8-10 watts under typical CUDA load and has the TDP of 15 watts under full load. Under load the ARM A57 cores automatically scale between 102 MHz and 1.9 GHz, the memory controller between 40MHz and 1.6GHz, and the Maxwell GPU between 76 MHz and 998 MHz.

The NVIDIA Jetson TX1 has three basic power states: ON, OFF and Sleep. In the ON state the module is fully functional and will operate normally. Additionally in this state, Jetson TX1 has different features to minimize the power when possible, e.g.:

- Advanced Power Management IC (PMIC)
- On system Power Gating
- Advanced on chip Clock Gating
- Dynamic Voltage and Frequency Scaling (DVFS)
- Always on logic used to wake the system based on either a timer event or an external trigger (e.g., key press)
- Low power DRAM (LPDDR4)

The Sleep state allows the module to promptly resume to an operational state without performing a full boot sequence. This state can be entered by the software, e.g. by the operating system.
Finally, in the OFF state the system is not powered. The module can enter this state as a result of various factors, e.g. pressing the power button, software shutdown, thermal shutdown or voltage brownout.

In addition, the NVIDIA Tegra X1 SoC module has 7 power states: SC0 (Active), SC1-SC3 (IDLE), SC4 (Suspend), SC7 (Deep Sleep) and OFF. Detailed information regarding the characteristics of each state can be obtained in NVIDIA technical documentation [56].

NVIDIA Tegra K1

The NVIDIA Tegra K1 is based on the Apalis form factor, which will be supported by M2DC servers. This form factor has already been supported by previous versions of the RECS|Box systems.

NVIDIA Tegra K1 SoC uses 0.6W to 3W during normal use, and has the TDP of 15 watts under full load (with the CPU, GPU, camera ISP's and codec hardware pushed to their limits). The whole Jetson TK1 board usually uses between 7-11.5 watts under typical CUDA load, and 3-5 watts for a typical CPU load.

The power management techniques of Tegra K1 SoC include:

- Power Management Controller (PMC)
- Power Gating
- Clock Gating
- Dynamic Voltage and Frequency Scaling (DVFS)
- Real Time Clock (RTC)

The system power states of Tegra K1 SoC include: Active, LP1 (Suspend), LP0 (Deep Sleep) and OFF. Detailed information regarding the characteristics of each state can be obtained in NVIDIA technical documentation [57].

NVIDIA Tesla K40

NVIDIA Tesla K40 has a standard PCI-Express 3.0 interface and therefore can be easily integrated with the M2DC servers. NVIDIA Tesla K40 uses 21 watts of power in idle state and has a TDP of 235 watts. Standard power management features described at the beginning of this section apply for this GPU.

NVIDIA Tesla P100

The flagship version of NVIDIA Tesla P100 comes with the NVlink interconnect and therefore will not be integrated within the scope of M2DC. However, its PCI-Express counterpart can be integrated as a regular PCIe accelerator. This card has not been released to the market yet and is only expected in Q4 2016. Therefore, power measurements cannot be done yet. However, it is known that the maximum power consumption (TDP) of the PCIe version will be 250 watts (as compared to 300 watts for its NVlink version). Nevertheless, this accelerator is of particular interest of M2DC due to its computing power and energy efficiency.

5.2.4 Power model description of HP-FPGA-based microserver

The most heating components are, from max to min, the FPGA SoC, the power block and the DDR4 banks. The figure below shows the power distribution with a basic color map.

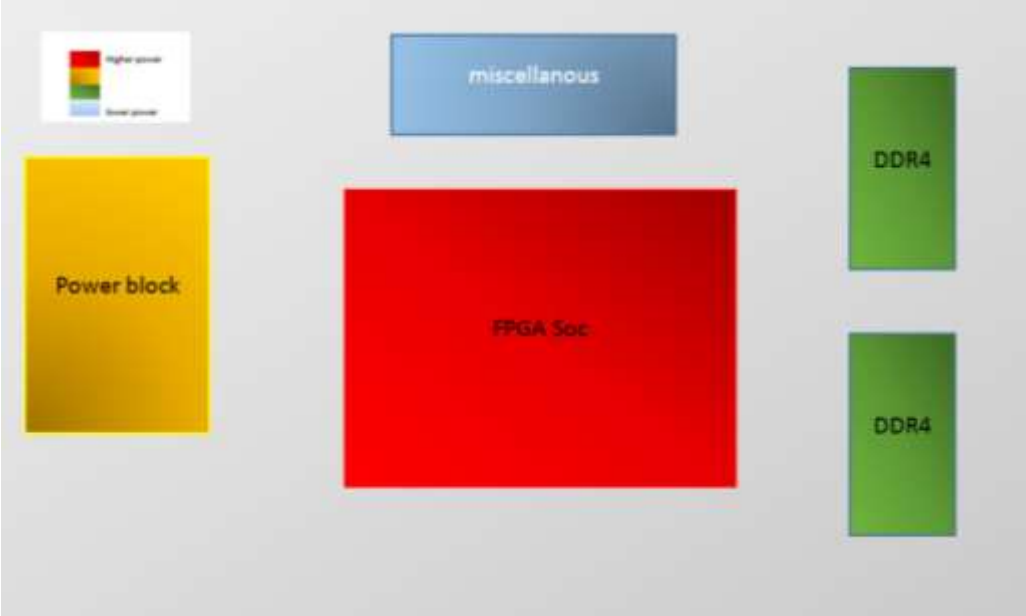

Figure 21: Heating components picture on the FPGA board

Power budget

The maximum input power is **80 W** delivered under 12V (the ComExpress nominal value is 50W):

- The large-matrix FPGA may have an estimated static power of 10W. The dynamic power mainly depends on the application (SEE accelerator type). It means that clock frequencies, DDR4 I/O activity, processor (DSP, HPS) and logic toggle percentage, used for intensive computation tasks, will increase significantly the dynamic power. It may range from 0 to 45W.
- For the power supplies part, the dissipation may be 12 W for 80W consumption (with a typical efficiency of 85%)
- The DDR4 may have a 7W dissipated power.
- A total of 5W for the remaining low-powered components.

Heatsink

- The FPGA, Main DCDC modules (Power Block) will be cooled through a heatsink.

SEE Accelerator power consumption:

- For each SEE accelerator a calibration phase could be done in order to determine its associated power consumption cost. This information could be stored and used by the thermal management system to the workload management system. It will help to select the appropriated nodes, in particular to avoid any hotspot.

Available measurements:

- On-board temperature sensor
- Internal junction temperature of components (FPGA, DCDC modules)
- Consumption and voltages of DCDC modules

Available Power controls

- The possibility to power ON/OFF the DCDC modules on board
- The possibility to power ON/OFF the whole board.
- The possibility to set the voltage value of some DCDC modules
- To dynamically change the configuration of the FPGA resources.
- To dynamically change some clock frequencies

5.3 *Thermal models*

The following subsections describe thermal models and corresponding parameters of modules that can be placed within RECS®|Box.

5.3.1 Thermal model of a component

Processor (and FPGA) based application reliability as well as performance are determined by the operating temperature in parts. As IC process geometries shrink and densities increase, managing power becomes more critical and complex to successful designs. Thermal metrics in the device data sheet provide a first-level approximation of system thermal performance. These parameters associated with a thermal model provide an estimation of the junction temperature of the device. The thermal management should drive to avoid exceeding the maximum temperature allowed of the component junction.
The thermal model being used here is the simple two resistors (2R) model including a junction point. The figure below gives a representation: When using a heat sink, the case (TOP) to ambient resistance can be decomposed as the sum of the case to heat sink resistance with the heat sink to ambient resistance, where

- Rcs represents the case to heatsink thermal resistance
- Rsa the heatsink to ambient thermal performance of the heatsink (depends upon the airflow)
- Rba: Resistance from board to ambient (depends upon the airflow).
- Rjc : Resistance from junction to the top of the package (case)
- Rjb : Resistance from junction to the board

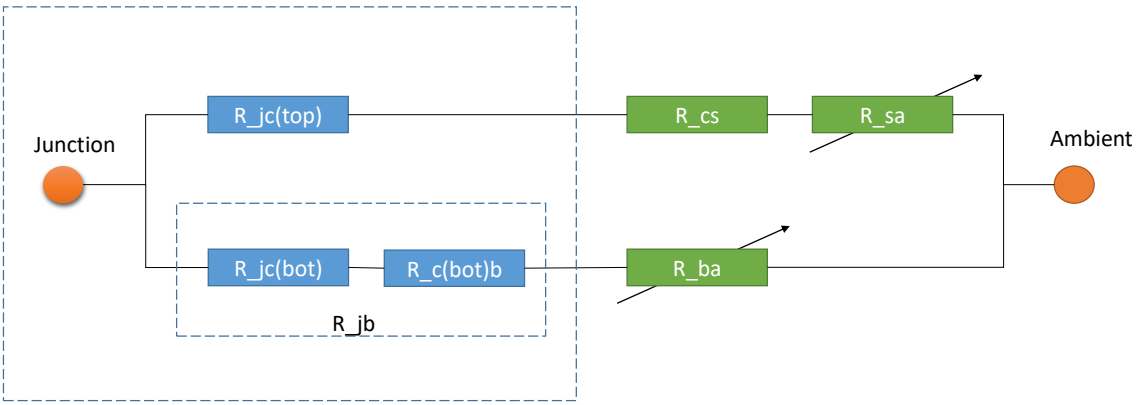

Figure 22: Thermal path for a single component

The junction temperature is obtained from the previous parameters using the following formula:

$$Tj = Ta + \left(\frac{(Rjb + Rba) \times (Rjc + Rcs + Rsa)}{Rjb + Rba + Rjc + Rcs + Rsa}\right) \times Power$$

5.3.2 Thermal model of the CPU module

To describe the thermal behavior of a CPU, we follow the model presented and validated in [58].
Similarly, in case of RECS®|Box we are considering an enclosure capable to host a number of microservers (nodes). Each enclosure has fans attached, responsible for blowing air (described by its volume V) over its internal components. We also consider another temperature point, namely inner temperature, to better reflect and model the nature of the M2DC approach.

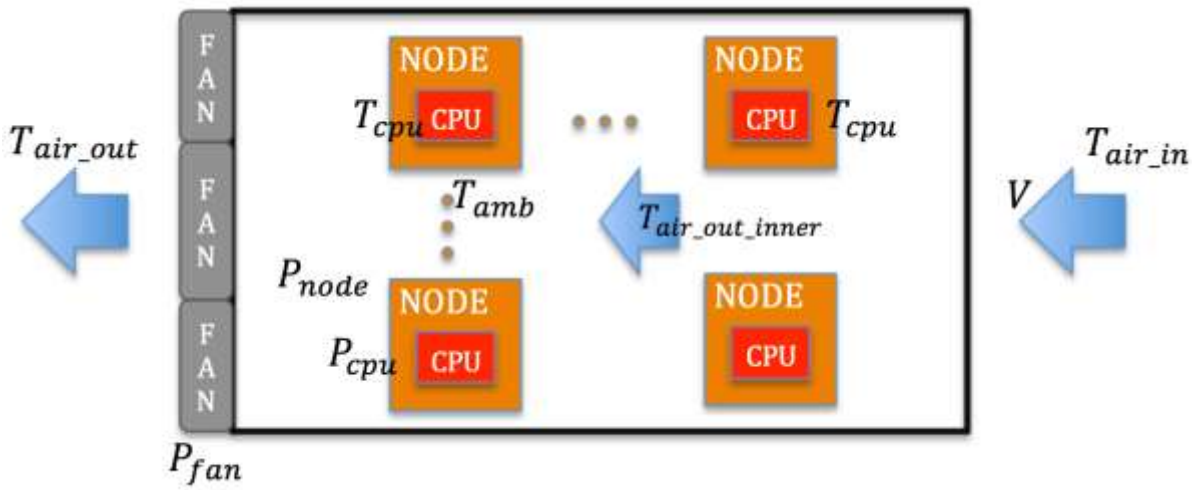

Figure 23: Parameters for CPU thermal model

Figure 23 recalls the general idea behind the proposed model (as defined in [58]), however adapted according to the M2DC project specifics. Presented approach considers five spots as significant for modeling the thermal behavior of the server:

- T_{air_in}: The inlet temperature specifies the temperature at the inlet of the server
- T_{air_out}: Analogously, the outlet temperature represents the corresponding value at its outlet
- $T_{air\_out\_inner}$: The inner temperature is a temperature of air leaving the particular nodes - for the node placed at the outlet of the server it is equal to outlet temperature (T_{air_out}) and at the inlet side (before the air reaches first node) it is equal to T_{air_in}
- T_{cpu} defines a temperature of the processor
- T_{amb} stands for the local ambient temperature surrounding the processor. It should be noted that local ambient temperature varies depending on the location of the processor within an enclosure. Its value may be determined using measurements or approximated using other temperature values. Our previous studies [58] revealed that a quite accurate estimation of its value is a weighted average of processor and inner air temperature expressed as follows:

$$T_{amb} = (2 * T_{cpu} + T_{air\_out\_inner})/3$$

The presented model describing the thermal behavior of the server (as most of the existing ones in literature) benefits from Newton's law of cooling and follows the duality between thermal and electrical phenomena. Here we present only the final formulas – all the necessary transformations can be found in [58]. The following relation describes the temperature changes of the processor:

$$T_{cpu}(t + \Delta t) = T_{cpu}^{\infty}(t + \Delta t) + \left(T_{cpu}(t) - T_{cpu}^{\infty}(t + \Delta t)\right) e^{-\frac{\Delta t}{R(t+\Delta t)C}}$$

with

$$T_{cpu}(t) = P_{cpu}(t)R(t) + T_{amb}(t), R(t) = R_{cond} + \frac{1}{k_v V(t)^n}$$

and

$$V(t) = \sqrt[3]{k_p P_{fan}(t)},$$

where

- $T_{cpu}(t)$ is the temperature of the processor at given time t,
- Δt is a time step,
- T_{amb} is the temperature of ambient air,
- P_{cpu} is the processor power consumption,
- *R* defines thermal resistance, which is often represented by the conductive resistance and the convective resistance,

- V is the airflow volume,
- n is a heatsink-specific factor,
- k_v is the parameter that needs to be determined experimentally as it is typical for a given equipment model,
- P_{fan} is the power consumption of the fan,
- k_p is another parameter that needs to be determined experimentally for the specific hardware configuration and
- C is the thermal capacitance.

More details concerning each of the parameters and origin of particular dependencies can be found in [58].

One should note that the formula above is valid for all the CPUs. However, as mentioned above, ambient temperature used for the calculations may vary depending on their placement within the enclosure. As inlet temperatures for the adjacent CPUs are different, their values need to be recalculated. To this end we can use the formula specifying the outlet temperature, which in our case is a temperature of air leaving the former processor [58]:

$$T_{air\_out\_inner}(t+\Delta t) = T^{\infty}_{air\_out\_inner}(t+\Delta t) + \left(T_{air\_out\_inner}(t) - T^{\infty}_{air\_out\_inner}(t+\Delta t)\right) e^{-\frac{\Delta t}{R(t+\Delta t)C}}$$

with $T_{air\_out\_inner}(t) = \frac{P_{node}(t)+P_{fan}(t)}{K(t)} + T_{air\_in}(t)$, $K(t) = \rho V(t) C_p$ that refers to the heat absorption capacity of air.

In case of several nodes:

$$T_{air\_out\_inner}(t) = \frac{\sum_{i=1}^{n} P_{node}(t) + P_{fan}(t)}{K(t)} + T_{air\_in}(t)$$

As described in Chapter 2, there are multiple thermal management mechanisms applied by CPUs to protect them against excessive heat and to keep temperature below safe thresholds.
In modern CPUs some advanced approaches were applied such as the sprint computing approach, which consists in relying on the slow thermal dynamics to transiently exceed a chip thermal design power (TDP) without reaching unsafe temperatures. Intel CPUs that can be integrated into RECS®|Box have this feature implemented in the form of Turbo boost 2.0, which is a hardware controller in recent Intel processors capable of operating the cores above their TDP, if the temperature state of the chip allows it.

5.3.3 Thermal model of the HP-FPGA-based microserver

Under the following assumptions:

- A unique heatsink is used per microserver
- A uniform temperature all over the heatsink
- A uniform temperature all over the PCB (due its large number of copper and conductive ground copper planes).
- The less dissipating devices are disregarded.
- Power dissipation interactions between components are also disregarded for this first view.
- The back of the PCB itself won't be able to dissipate as the baseboard will block the airflow.

The thermal model of the board can be simplified such as:

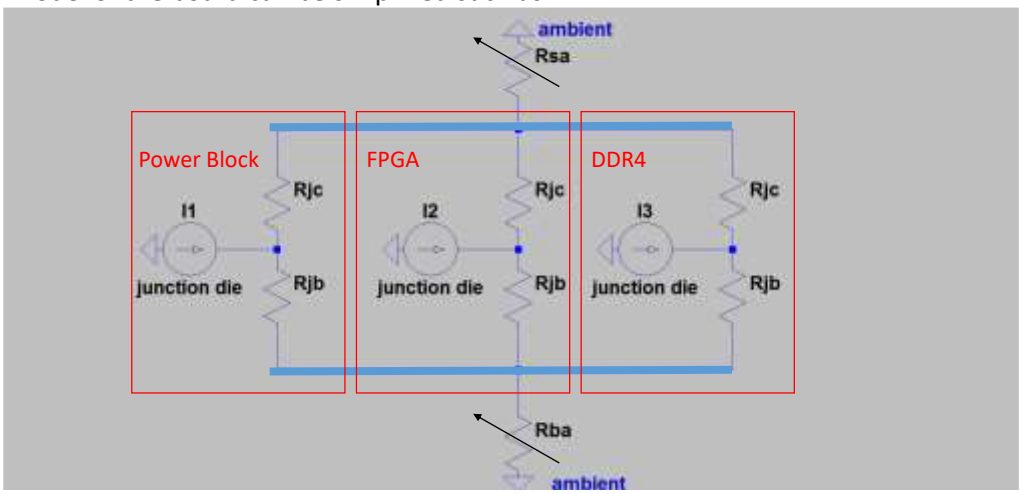

Figure 24: Thermal model of the FPGA board

where the thermal resistances Rsa and Rba are normally subject to change upon the airflow value. However, in the present case, since the air flow does not reach the board to ambient resistance, we assumed Rba much bigger than the other resistance parameters in the model. In consequence, here the contribution of Rba is neglected.

The following approximate calculation based upon the previous model allows selecting the appropriate heat sink by determining its thermal resistance.

Input for calculation (Estimation)

The following data used for calculation are rough estimations, as the board is still under design. The goal here is to show the thermal process calculation. Only the most dissipated components are taken into account. A most accurate and complete calculation should be done with the final design.

- **FPGA:** It is by far the most powerful component.
- **Power** modules: Only power modules used to power the FPGA are included. Others have less dissipation and are not well defined.
- **DDR4** modules: Due to the fact that the design of the board is not defined yet, especially in terms of the number of DDR4 banks, these ones are not included in the first thermal evaluation.

Parameters	FPGA	LT1 (Core)	LT2 (Transceivers)	LT3 (IO)
Estimated dissipated power	56W	3.5W	3.3W	1.4W
Maximum Tj	125°C	125°C	125°C	125°C
Rja	6.6°C/W	10.3°C/W	10.3°C/W	14°C/W
Rjc (top)	0.13°C/W	8.8°C/W	8.8°C/W	20°C/W
Rjb	1.8°C/W	1.3°C/W	1.3°C/W	5°C/W

Table 5-1: HP-FPGA-Microserver / Estimated thermal and power inputs

Calculation results

We assume that only the power modules referenced LT1, LT2, LT3 share the same heatsink with the FPGA. The heatsink evaluated is referenced 3-503807M from Coolinnovations with omnidirectional round pins. It is made of Aluminum with dimensions 125mm (width)*95mm (length) and a height of approximately 21mm, achieving a maximum thermal resistance of **0.24°C/W** under **600LFM** air flow speed (worst case since the information is not provided for a 1100LFM air flow where the thermal resistance is assumed lower).

The thermal resistance to ambient of the PCB is not fully defined (design related) and is set to 20°C/W for the need of the simulation.

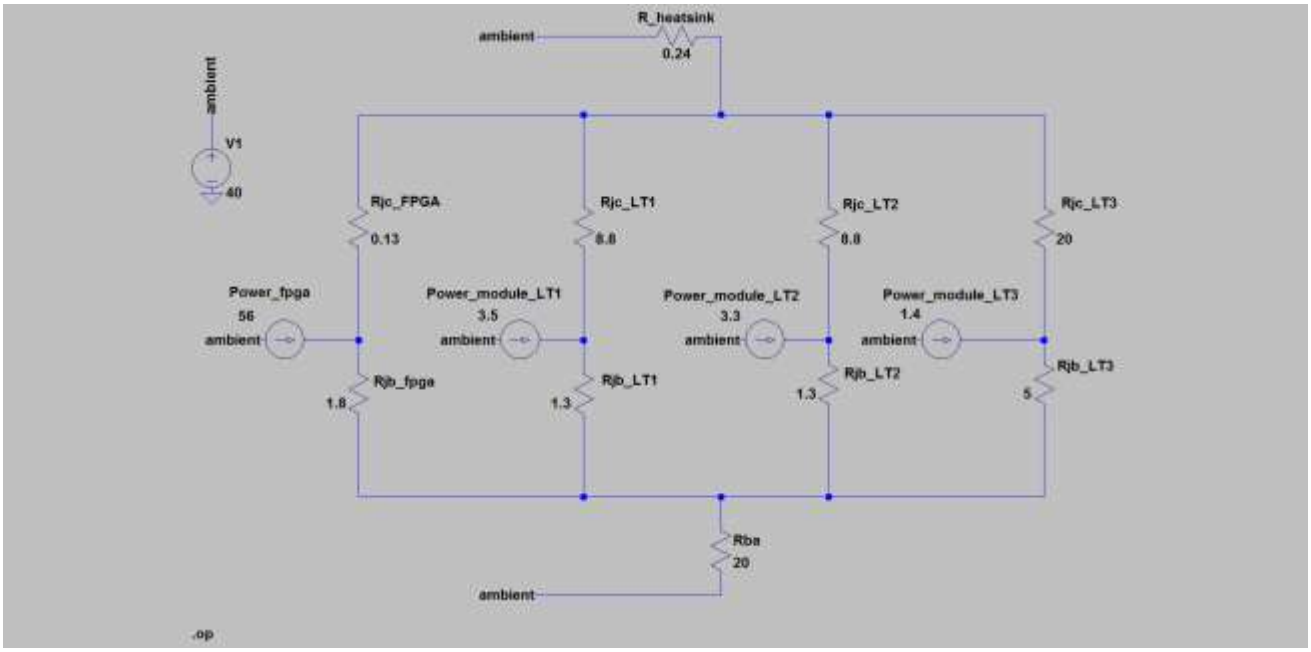

Figure 25: Thermal simulation for the devices sharing the heatsink

Devices	Computed junction temperature °C R_Heatsink = 0.24°C/W (600LFM) Tambient = 40°C
FPGA	62°C
LT1	70°C
LT2	70°C
LT3	71°C
Heatsink	55°C

Table 5-2: Junction temperatures computed for the devices sharing the heatsink

A mathematical model can be provided with the dynamic power consumption of the FPGA as parameter.

In both cases, we find that we can achieve a junction temperature of 80°C for the FPGA, less than the maximum junction temperature of 125°C (The reasonable target is less than 90°C). However, the second heatsink model is recommended.

5.3.4 Thermal model of the GPU module

This section contains description of JETSON Nvidia TX1 module.

Input for calculation (Estimation)

This LP microserver is thermally analyzed as a single entity. It incorporates a thermal transfer plate TTP on which a heatsink has to be mounted. The thermal resistance between the junction and the TTP is referenced as Rjp. The following data used reflect typical load conditions (CPU light, GPU heavy, SOC light and DRAM light).

Parameters	JETSON Nvidia TX1
Estimated dissipated power	15W
Maximum Tj	89°C
Rjp	0.6°C/W
Rjb	3.3°C/W

Table 3: LP-ARMV8-Microserver / Estimated thermal and power inputs

Calculation results:

The thermal resistance to ambient of the PCB is roughly estimated at 23K/W for the need of the simulation. In addition, the thermal resistance of the heatsink is assumed to be 1.9K/W with an airflow value of 1100LFM.

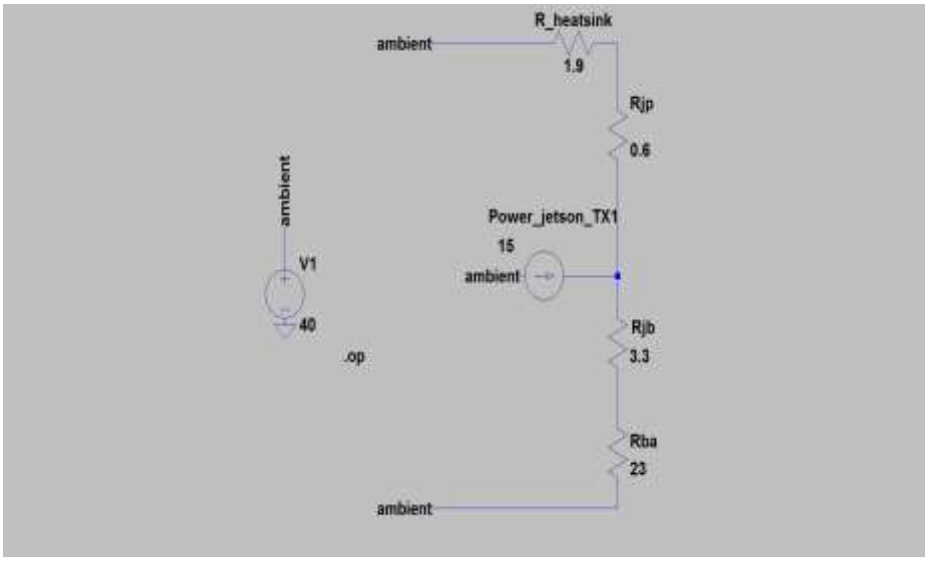

Figure 26: Thermal simulation for the LP ARMV8 board

The junction temperature of the tegra X1 is 75°C, well below 89°C as specified. In addition, the TTP temperature founded is 66°C, below the 80°C specified. The heatsink thermal resistance is sufficient to fulfill the Tegra X1 and TTP temperature specifications.

5.3.5 Thermal model of the low power ARM board

There are 4 APALIS based microservers modules analyzed in this section. The main relevant devices in terms of thermal performances are introduced in the table below for each module. These devices share the same heatsink.

Modules	Main devices
ZYNQ	CPU+FPGA
TK1	CPU+GPU
Exynos	CPU+GPU
T30	CPU

Table 4: Main devices per module

Input for calculation (Estimation)

The following data used reflects typical load conditions and thermal parameters for each module.

Parameters	ZYNQ	TK1	Exynos	T30
Estimated dissipated power	CPU 6W FPGA 8W	CPU 8W GPU 10W	CPU 14W GPU 8W	CPU 9W
Maximum Tj	100°C	100°C	100°C	100°C
Rjc (top)	0.4K/W			
Rjb	4.1K/W			

Table 5: APALIS based Microservers / Estimated thermal and power inputs

Calculation results:

The thermal resistance to the ambient of the PCB is roughly estimated at 16K/W for the need of the simulation. In addition, the thermal resistance of the heatsink is assumed to be 1.8K/W with an airflow value of 1100LFM. The figures below show the simulation model for each Apalis based module.

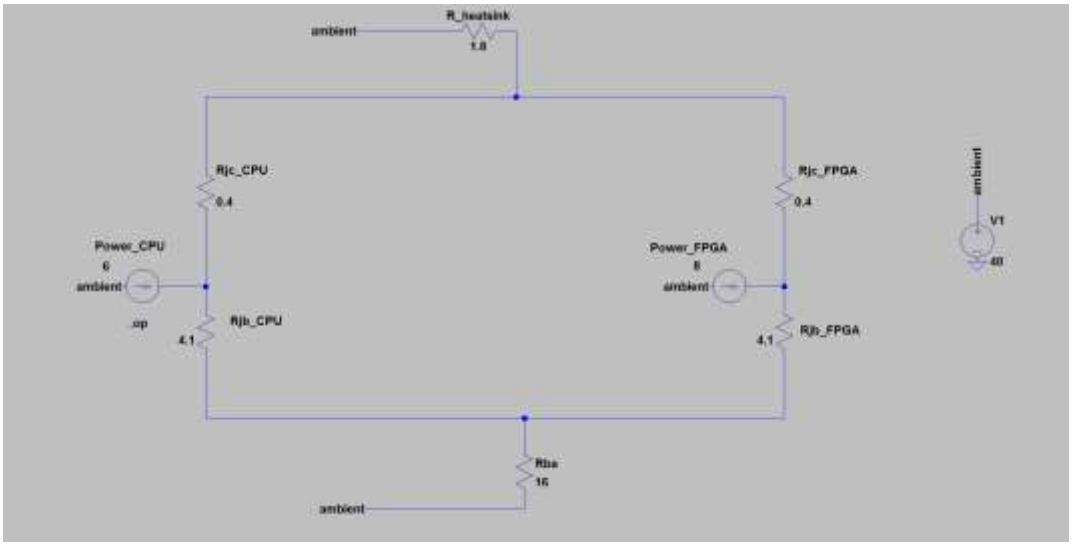

Figure 27: Thermal simulation for the Zynq microserver

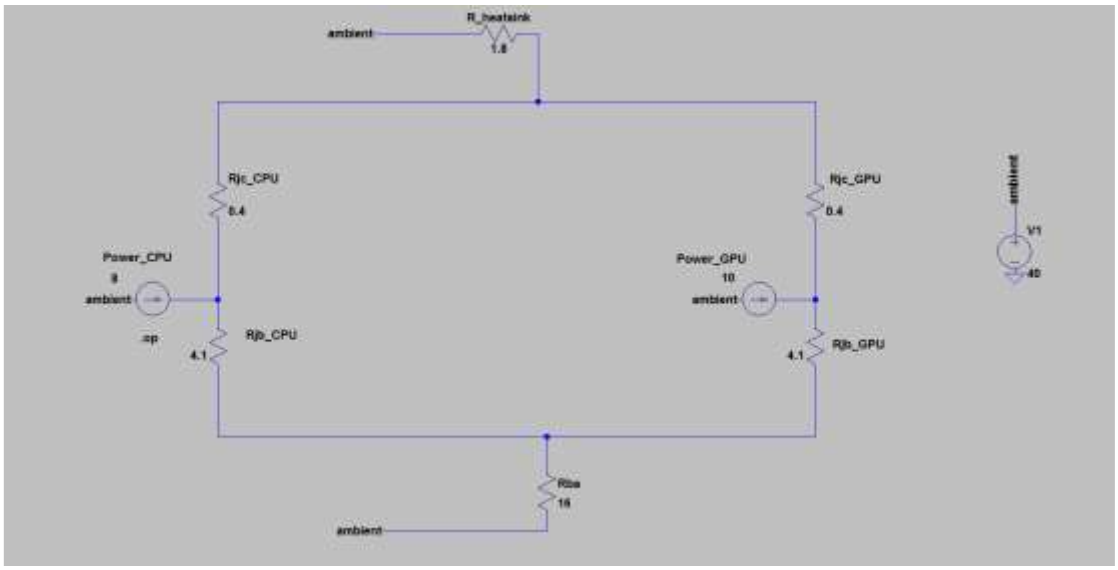

Figure 28: Thermal simulation for the TK1 microserver

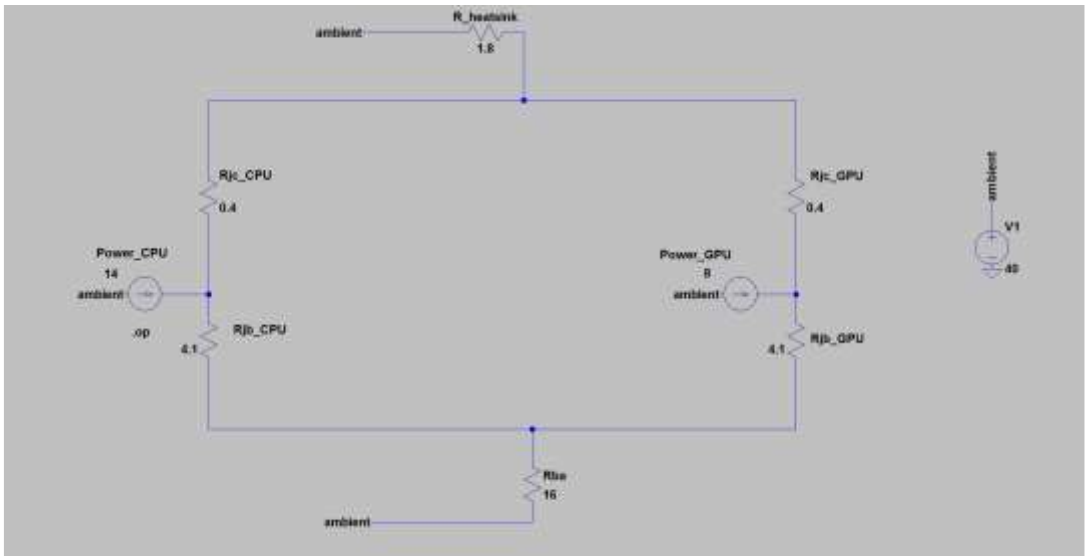

Figure 29: Thermal simulation for the EXYNOS microserver

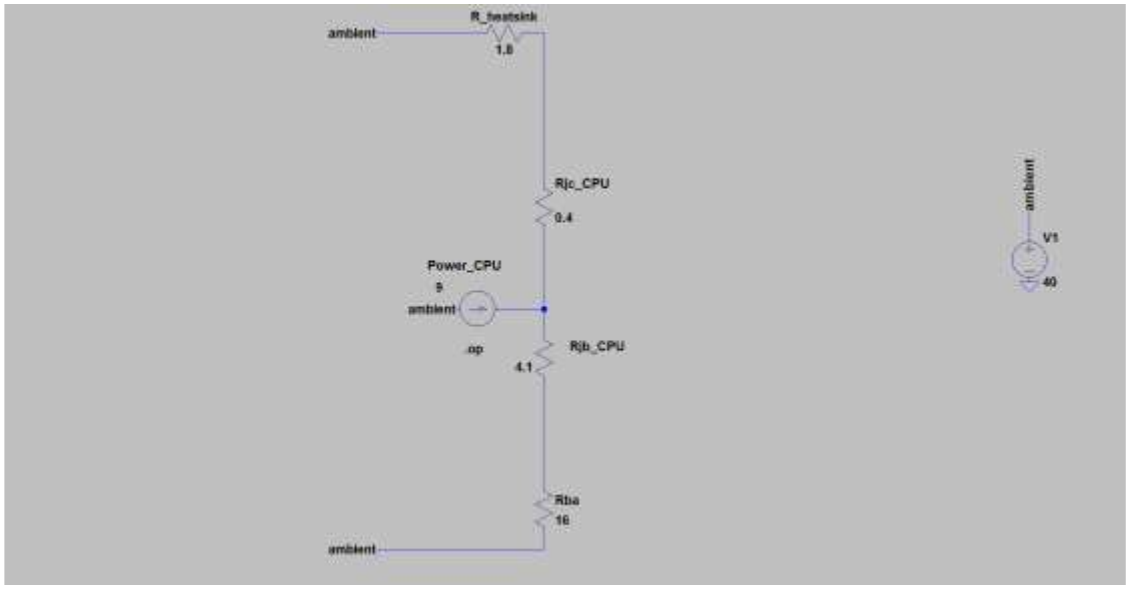

Figure 30: Thermal simulation for the T30 microserver

	Devices	Computed junction temperature °C R_Heatsink = 1.8K/W (1100LFM) Tambient = 40°C
Zynq	CPU	65
	FPGA	66
	Heatsink	63
TK1	CPU	72
	GPU	73
	Heatsink	69
EXYNOS	CPU	81
	GPU	79
	Heatsink	76
T30	CPU	81
	Heatsink	76

Table 6: Temperatures computed for the devices and the heatsink

We find that we can achieve a junction temperature lower than the maximum junction temperature for all the components in the APALIS modules.

6 Plans towards resource and thermal management policies

The RECS®|Box due to its envisioned features such as scalability, density, heterogeneity and use of innovative components will require advanced resource and thermal policies that will allow achieving significant energy savings while maintaining high reliability and usability of the system under various conditions. These methods will be proposed, studied, and, in case of satisfactory results, implemented within the components defined in this section. As mentioned in Section 3, resource and thermal management will provide three functional modules, namely: Energy Saver Manager, Fan Manager and Power Capping Manager. The following section contains description of management strategies that are intended to be included within these modules.

6.1 *Energy Saver Manager*

The Energy Saver Manager is responsible for taking actions aiming at reduction of power consumed by particular system components. These actions will include management of power states of particular system components and exploitation of dynamic voltage and frequency scaling technology to optimize speed of processing units if possible. The Energy Saver Manager will have insight into the current state of the system, sensor readings as well as the power and thermal models of the system components. Based on that, it will look for possibilities to achieve a reduction of the power consumption. First of all, it will consider limiting the power of idle parts of the RECS®|Box. In this case the Energy Saver Manager needs to gain the knowledge about the current and future system computation requirements. These data can be obtained either from the Workload Management or analysing current utilization and predicting future trends based on historical data. Having the candidates, the module will determine whether the given component should be turned off, switch to a certain power state or remain running. The Energy Saver Manger will be in charge of keeping some of the unused resources "ready" for the upcoming load – thus it will perform selective and delayed switching. In a similar way the computation speed of nodes will be adjusted accordingly, by interacting with the the Workload Management or analyzing system load.

The Energy Saver Manager will come with a set of predefined policies aiming at system optimization:

- High Performance – RECS®|Box components operate at their full speed all the time trying to reach the highest performance
- Low Power – RECS®|Box components operate in the low power mode (for instance the highest P-state in case of process), devoting performance to save energy
- Active - manages resources with respect to the load running on the server trying to find optimal trade-off between the power and performance.

6.2 *Fan Manager*

The Fan Manager is in charge of controlling the fan speed in order to keep all the components within the desired temperature range and optimize its power usage. Fan management will benefit from the possibility of managing each fan separately in a fine-grained manner. Similarly to the solutions adopted by HP and Dell servers, each fan will be responsible for covering the given zone within the RECS®|Box. Due to a large number of sensors within the particular zones, each fan in the box will have the outputs from multiple sensors mapped to it. This will allow providing cooling only where it is needed without wasting power in a situation where the operation of a multiplicity of fans is driven by a single hotspot. Apart from sensor readings, the Fan Manager will utilise power and thermal models introduced in Section 5 to determine trends in their changes. Taking into account the prediction of particular component temperatures, the Fan Manager will continuously adjust the speed of particular fans proactively trying to maintain a particular set-point temperature and preventing the components from exceeding predefined thresholds. The proposed control feedback algorithm will also have to consider the trade-offs between the power consumed by the fans due to the increase in their rotation speed

and the increase in power caused by the increase in the temperature (so called power leakage). In this manner, the method will allow the system to heat up (without exceeding the given limit) when this saves more fan power than it costs in terms of power leakage.

An example of a feedback loop control method applied in modern servers is the PID approach described in Section 2.1.4. It combines a reactive and a proactive approach but concentrates on adjustment to a desired setpoint rather that an overall energy minimisation. Furthermore, PID coefficients need to be set, which can be difficult if the hardware configuration changes. In the case of RECS®|Box these coefficients may depend strongly on a specific configuration (appliance). Hence, an automated intelligent method needs to set these coefficients or another approach should be adopted.

6.3 *Power Capping Manager*

The Power Capping Manager is responsible for limiting the maximum RECS®|Box power consumption to a desired level set by the user. It will take advantage of both the Energy Saver Manager and the Fan Manager. Thus, it will consider management of power states, dynamic voltage and frequency scaling as well as fans management in order to limit the power. As the actions taken by the Power Capping Manager are performed with respect to external requirements provided by the user, they have the highest priority. Thus, eventually, the Power Capping Manager could make its decision without interacting with the Workload Management. The Power Capping Manger needs to know in advance how much energy can be saved while performing the given action (reducing the speed of processor, switching node off, etc). Firstly, the Power Capping Manager will try to reduce the power consumed by unused nodes putting them into low power mode or switching them off and reducing the speed of corresponding fans. Secondly, it will try to adjust the processor's speed to the level in which the power consumption of the total system is below the given level. After all, it will gradually force some of the running nodes to switch off until the desired power level is reached.

6.4 *Dynamic Thermal Manager*

The Dynamic Thermal Manager is responsible for controlling the temperature of the CPU cores in the system. In a hierarchical thermal control strategy, the Dynamic Thermal Manager is the fastest thermal control loop, sensing the silicon die temperature using on-chip temperature sensors and acting only through the fast actuators that affect chip temperature, such as DVFS. Its implementation requires fast and low latency access to both the on-chip temperature sensors and DVFS knobs. For this reason, we will evaluate if there is the need to implement part of the control strategy in the operating system kernel.

Its action allows both to counteract fast temperature variations induced by power peaks and to limit the temperature increases that occur under heavy workload condition. In the case of unpredictable load increases, it keeps temperature under control during the period of time required for the slower actuators, such as fans, to increase thermal dissipation. The Dynamic Thermal Manager is only responsible for controlling the chip temperatures by acting on the DVFS actuators. It assumes the presence of a higher level thermal optimization strategy that is integrated with the Energy Saver Manager and the Fan Manager.

The Dynamic Thermal Manager takes as input the desired temperature set point computed by the Energy Saver Manager and provides a signal that allows higher level and possibly predictive thermal controllers to manage the on board actuators such as the cooling fans.

Our goal is to perform an extensive feasibility check of this solution to verify the possibility of its inclusion in the M2DC computing architecture.

7 Conclusion

This document contains a summary of the work done within first 3 months of Task T4.4 Intelligent resource and thermal management policies. It concentrates on 3 main points: the analysis of the state of the art in resource management of servers, power and thermal models of the RECS®|Box, and management software architecture. The state of the art contains power and thermal management methods both from research and industry – implemented in off-the shelf servers. Thermal models of M2DC microservers and the whole platform are essential to develop methods that are optimised to M2DC Appliances and able to cope with RECS®|Box flexibility, heterogeneity and scale. As the work on hardware design and components benchmarks and integration is ongoing this is a work in progress, which will be detailed within next months and deliverables. In the software architecture part an initial design of interfaces between components involved in resource and thermal management were defined. Particularly important is the relation between resource and workload management (part of Task T4.5) as well as interfaces to specific hardware components and sensors.

Based on this analysis, initial assumptions and plans related to management policies development were defined. As presented in the state of the art section numerous methods of resource and thermal management can be found in literature. Some of them are well established and implemented in off-the-shelf systems. M2DC will adopt existing techniques where possible and take advantage of power and thermal management features in new components that will be integrated into RECS®|Box. However, planned features of RECS®|Box and its efficiency objectives will require important customisation or a development of new techniques. RECS®|Box scale, density, and heterogeneity introduce challenges but on the other hand provide opportunities for better energy efficiency and performance for certain classes of applications. Therefore, we defined methods to be developed and studied such as advanced fan management with independent zones, predictive models for resource and thermal management, software power capping, resource management in the presence of accelerators, and energy efficient modes.

General architecture was defined including definition of several components responsible for workload, resource, and thermal management along with interfaces between them and other parts of the system. Energy Saver Manager, Fan Manager, Power Capping Manager and Dynamic Thermal Manager were proposed to take actions aiming at energy savings by server nodes, control the fan speed, manage power caps and control CPU core temperature, respectively. These components will take their control actions based on monitoring data coming from microservers, power meters and numerous sensors – all through dedicated monitoring system and OpenStack components. Additionally the Workload Management component will be responsible for workload management and prediction along with interfaces to OpenStack and potential application managers in the case of specific appliances. The design of Workload Manager will be described in a first deliverable from T4.5 task.

8 Glossary

API: (Application Programming Interface) is a set of defined methods used for communication between various (SW) modules iv, 19, 20, 21

ARM: (Advanced RISC Machine) describes a family of architectures for computer processors building on reduced instruction set computing (RISC). M2DC makes use of CPUs based on the latest ARMv8 64bit architecture v, 14, 21, 28, 29, 39

CAPEX: (Capital Expenditures) is the cost of buying or improving physical assets 3

Ceilometer: (Telemetry) is the monitoring component of OpenStack collecting and saving data about physical and virtual resources as well as triggering actions if certain conditions are met iv, 20

DVFS: (Dynamic voltage and frequency scaling) is a power management method in computer systems, where the voltage/frequency is adjusted to meet specific requirementes iv, 8, 10, 11, 12, 14, 29, 30, 31, 45

FPGA: (Field Programmable Gate Array) is an integrated circuit which can be configured after manufacturing as it contains an array of programmable logic blocks and a hierarchy of reconfigurable interconnects v, 1, 21, 24, 30, 31, 32, 35, 36, 37, 39, 42

IPMI: (Intelligent Platform Management Interface) is a set of computer interfaces for managing and monitoring computer systems iv, 19, 20

M2DC server: the server platform developed within the M2DC project, which will be sold by Christmann under the selling name RECS®|Box. 22

Nova: (Compute) is the component in the OpenStack cloud manager responsible for provisioning compute instances. Nova also integrates its own scheduler to select target nodes based on different parameters iv, 20

NRPE: (Nagios Remote Plugin Executor) supports remote execution of Nagios plugins 19

OpenStack: a popular cloud management software, which is used as base platform for middleware development in the M2DC project iv, 20, 21, 46

OPEX: (Operating Expenditures) is the cost of running a product or a system 3

PCI: (Peripheral Component Interconnect) is a computer bus used for attaching hardware devices 22, 25, 30, 31

RECS_Master: the embedded controller in the RECS|Box system which is also responsible for monitoring and serves as an interface to the hardware 19, 20, 21

RECS®|Box: is the selling name of microserver chassis sold by Christmann. The fourth revision of RECS|Box is the M2DC server, which is the server platform developed within the M2DC project iv, 7, 19, 22, 25, 33, 35, 44, 45, 46

RECSDaemon: multiplatform daemon used for monitoring within the @RECS®|Box 19, 20

REST: (REpresentational State Transfer) is a definition for implementing web interfaces offering a uniform and predefined set of stateless operations for requesting systems 19, 20

RPM: (Revolutions per minute) is a measure of the rotation 19

SoC: (System On Chip) an integrated circuit with all components of a computer system integrated 22, 30, 31

SVD: (Simulation, Visualization and decision support toolkit) is a set of open source tools used to model and analysed data centres in terms of their energy efficiency 1, 4, 5

TCO: (Total Cost of Ownership) is the cost of acquisition, operating, upgrades and replacement of a product or system 3, 4

TDP: (Thermal Design Power) is the maximum amount of heat produced by a computer component that the cooling system is able to remove under any load 11, 25, 26, 27, 30, 31, 35

UPS: (Uninterruptible Power Supply) is an electrical device providing emergence power in case if power source failure 5

9 References

[1] Mariano Cecowski et al., "D7.2 - Market Assessment," 2016.

[2] [Online]. http://ecoinfo.cnrs.fr/IMG/pdf/ashrae_2011_thermal_guidelines_data_center.pdf

[3] Marcos Dias de Assuncao, Laurent Lefevre Anne-Cecile Orgerie, "A survey on techniques for improving the energy efficiency of large-scale distributed systems," 2013.

[4] H. Viswanathan, E. K. Lee, M. G. and Dario Pompili, and M. Parashar. I. Rodero, *Energy-efficient thermal- aware autonomic management of virtualized hpc cloud infrastructure*.: Journal of Grid Computing, 2012.

[5] A. Oleksiak, W. Piatek, J. Salom, and L. Siso G. D. Costa, "Minimization of costs and energy consumption in a data center by a workload-based capacity management," in *3rd International Workshop on Energy-Efficient Data Centres Co-located with the ACM e-Energy*, 2014.

[6] Wu-chun Feng Chung-hsing Hsu, "A Power-Aware Run-Time System for High-Performance Computing," in *Proceedings of the ACM/IEEE SC 2005 Conference*, 2005.

[7] I. Issenin, R. Cornea, R. Gupta, N. Dutt, A. Veidenbaum, A. Nicolau A. Azevedo, "Profile-based dynamic voltage scheduling using program checkpoints," in *n DATE*, 2002.

[8] A. Sivasubramaniam, M. Kandemir, M. Irwin C. Liu, "Exploiting barriers to optimize power consumption of cmps," in *In Parallel and Distributed Processing Symposium*, 2005.

[9] J. Martinez, M. Huang J. Li, "The thrifty barrier: energyaware synchronization in shared-memory multiprocessor," in *10th International Symposium on High Performance Computer Architecture*, 2004.

[10] L.L.H. Andrew, H. Kim, M. Chiang Y. Kamitsos, "Optimal sleep patterns for serving delay tolerant job," in *ACM eEnergy*, 2010.

[11] Allen C.-H. Wu Chi-Hong Hwan, "A Predictive System Shutdown Method for Energy Saving of Event-Driven Computation," *ACM Transactions on Design Automation of Electronic Systems*, 2000.

[12] Mor Harchol-Balter Anshul Gandhi, "How Data Center Size Impacts the Effectiveness of Dynamic Power Management," in *Allerton Conference on Communication, Control, and Computing*, 2011.

[13] A. Vassighi and M. Sachdev., *Thermal and Power Management of Integrated Circuits.*: Springer, 2006.

[14] G. Van den Bosch, R. Bellens, G. Groeseneken, H.E. Maes. Paul Heremans, "Temperature dependence of the channel hot-carrier degradation of n-channel mosfet," *IEEE Transactions on Electron Devices*, 1990.

[15] K.-Y. Fu, C.J. Varker M.L. Dreyer, "An electromigration model that includes the effects of microstructure and temperature on mass transport," *Journal of Applied Physics*, 1993.

[16] H.J. Mattausch, M. Miyake, T. Iizuka, M. Miura-Mattausch, K. Matsuzawa, S. Yamaguchi, T. Hoshida, M. Imade, R. Koh, T. Arakawa, and J. He C. Ma, "Compact reliability model for degradation of advanced p-mosfets due to nbti and hot-carrier effects in the circuit simulation," in *IEEE International Reliability Physics Symposium (IRPS)*, 2013.

[17] B. Vandeveldeand and E. Beyne, "Improved thermal fatigue reliability for flip chip assemblies using redistribution techniques," *IEEE Transactions on Advanced Packaging*, 2000.

[18] Phil Miller, Ehsan Totoni, Laxmikant V. Kale Osman Sarood, "'Cool' Load Balancing for High Performance Computing Data Centers," *IEEE Transactions on Computers*, vol. 61, pp. 1752-1764, 2012.

[19] R. Bettati S. Wang, "Reactive speed control in temperature-constrained realtime systems," in *Conference on Real-Time Systems (ECRTS)*, 2006.

[20] S. Ren G. Quan, "Leakage-aware real-time scheduling for maximal temperature minimization," *ACM SIGBED*, 2010.

[21] T. Kimbrel, and K. Pruhs N. Bansal, "Speed scaling to manage energy and temperature," *ournal of the ACM*, vol. 54, pp. 1-39, 2007.

[22] S. Wang, and L. Thiele J.-J. Chen, "Proactive speed scheduling for real-time tasks under thermal constraints," in *RTAS*, 2009, pp. 141–150.

[23] A. Andrei, P. Eles, and Z. Peng M. Bao, "On-line thermal aware dynamic voltage scaling for energy optimization with frequency/ temperature dependency consideration," in *Design Automation Conference*, 2009, pp. 490–495.

[24] E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger H. Esmaeilzadeh, "Dark silicon and the end of multicore scaling," *IEEE Micro*, 2012.

[25] M.B. Taylor, "A landscape of the new dark silicon design regime," *IEEE Micro*, 2013.

[26] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *High-Performance Computer Architecture (HPCA)*, 2001.

[27] J Donald and M Martonosi, "Techniques for Multicore Thermal Management: Classification and New Exploration," in *International Symposium on Computer Architecture*, 2006.

[28] T. Abdelzaher, and M.R. Stan K. Skadron, "Control-theoretic techniques and thermal-rc modeling for accurate and localized dynamic thermal management," in *International Symposium on Computer Architecture*, 2002.

[29] G. Magklis, R. Balasubramonian, D.H. Albonesi, S. Dwarkadas, and M.L. Scott G. Semeraro, "Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling," in *International Symposium on High-Performance Computer Architecture*, 2002.

[30] M. Gomaa, and T.N. Vijaykumar M.D. Powell, "Heat-and-run: Leveraging smt and cmp to manage power density through the operating system," in *International conference on Architectural support for programming languages and operating systems (ASPLOS)*, 2004.

[31] H. Amrouch, and J. Henkel T. Ebi, "Cool: Control-based optimization of load-balancing for thermal behavior," in *IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis (CODES+ISS)*, 2012.

[32] M. Kadin and S. Reda, "Frequency and voltage planning for multi-core processors under thermal constraints," in *IEEE International Conference on Computer Design*, 2008.

[33] D. Atienza, and G. De Micheli F. Zanini, "A control theory approach for thermal balancing of MPSoC," in *In Design Automation Conference (DAC)*, 2009.

[34] D. Atienza, L. Benini, and G. De Micheli F. Zanini, "Thermal-aware system-level modeling and management for multi-processor systems-on-chip," in *Design Automation Conference (DAC)*, 2011.

[35] M. Cacciari, A. Tilli, and L. Benini A. Bartolini, "Thermal and energy management of high-performance multicores: Distributed and self-calibrating model-predictive controller," *IEEE Transactions on Parallel and Distributed Systems*, 2013.

[36] L. Yixin, A. Chandawalla, M. Papaefthymiou, K.P. Pipe, T.F. Wenisch, M.M.K. Martin A. Raghavan, "Computational sprinting," in *In High Performance Computer Architecture*, 2012.

[37] A. Naveh, D. Rajwan, A. Ananthakrishnan, and E. Weissmann E. Rotem, "Power management architecture of the 2nd generation intel core microarchitecture, formerly codenamed sandy bridge," in *Hot chips*, 2011.

[38] C. Bash, N. Tolia, M. Marwah, X. Zhu, P. Ranganathan Z. Wang, "Optimal fan speed control for thermal management of servers," in *Proceedings of the ASME*, 2009.

[39] M. M. Sabry, D. Atienza, K. Vaidyanathan, and K. C. Gross J. Kim, "Global fan speed control considering non-ideal temperature measurements in enterprise servers," *DATE*, pp. 1-6, 2014.

[40] Y. Jin, Y.-K. Wu, K. C. Gross, K. Vaidyanathan, and T. S. Rosing C. S. Chan, "Fan-speed-aware scheduling of data intensive jobs," *ISLPED*, pp. 409-414, 2012.

[41] J. L. Ayala, J. M. Moya, K. Vaidyanathan, K. C. Gross, and A. K. Coskun M. Zapater, "Leakage and temperature aware server control for improving energy efficiency in data centers," *DATA*, pp. 266-269, 2013.

[42] J. Kim, N. Chang, J. Choi, S. W. Chung, and E.-Y. Chung D. Shin, "Energy-optimal dynamic thermal management for green computing," in *International Conference on Computer-Aided Design*, 2009, pp. 652–657.

[43] M. Allen-Ware, J. Carter, E. Elnozahy, H. Hamann, T. Keller, C. Lefurgy, J. Li, K. Rajamani, and J. Rubio W. Huang, "Tapo: Thermal-aware power optimization techniques for servers and data centers," in *EEE International Green Computing Conference*, 2011, pp. 1-8.

[44] R. Nath, T. Rosing, R.Z. Ayoub, "JETC: joint energy thermal and cooling management for memory and CPU subsystems in servers," in *Proceedings of HPCA*, 2012, pp. 299–310.

[45] J. Chase, P. Ranganathan, and R. Sharma J.Moore, "Making scheduling "cool": Temperature – aware workload placement in data centers," in *Proceedings of the 2005 USENIX Annual Technical Conference*, 2005.

[46] S. K. S. Gupta, D. Stanzione, and P. Cayton Q. Tang, "Thermal-aware task scheduling to minimize energy usage of blade server based datacenters," 2006.

[47] Power efficiency and power management in HP ProLiant servers. [Online]. http://h10032.www1.hp.com/ctg/Manual/c03161908.pdf

[48] R610™, AND R710™ SERVERS THERMAL DESIGN OF THE DELL™ POWEREDGE™ T610™. [Online]. http://www.dell.com/downloads/global/products/pedge/en/server-poweredge-11g-thermal-design-en.pdf

[49] [Online]. http://en.community.dell.com/techcenter/b/techcenter/archive/2013/06/04/dell-poweredge-powermanagement-options-in-vmware-esxi-environment

[50] Zhenhua Liu et al., Renewable and Cooling Aware Workload Management for Sustainable Data Centers, 2012.

[51] Eun Kyung Lee, Hariharasudhan Viswanathan, and Dario Pompili, "Proactive Thermal-aware Resource Management in Virtualized HPC Cloud Datacenters," *IEEE Transactions on Cloud Computing*, 2016.

[52] Violaine Villebonnet and Georges Da Costa, "Thermal-aware cloud middleware to reduce cooling needs," *Proceedings of 23rd International WETICE Conference*, 2014.

[53] Toradex AG. (2015) Apalis Module Architecture. [Online]. http://developer.toradex.com/hardware-resources/arm-family/apalis-module-architecture

[54] [Online]. https://developer.arm.com/docs/ddi0488/latest/functional-description/power-management

[55] [Online]. http://infocenter.arm.com/help/topic/com.arm.doc.den0022c/DEN0022C_Power_State_Coordination_Interface.pdf

[56] (2016) NVIDIA Jetson TX1System-on-Module Data Sheet.

[57] (2016) NVIDIA Tegra K1 Series Processors with Kepler Mobile GPU for Embedded Applications Data Sheet.

[58] Ariel Oleksiak, Georges Da Costa Wojciech Piatek, "Energy and Thermal Models for Simulation of Workload and Resource Management in Computing Systems," *Simulation Modelling Practice and Theory*, 2015.