2009

# How to select measures for decision support systems - An optimization approach integrating informational and economic objectives

Eugenio Capra
*Politecnico di Milano, Dipartimento di Elettronica e Informazione, Italy*, eugenio.capra@polimi.it

Francesco Merlo
*Politecnico di Milano*, merlo@elet.polimi.it

# GREEN IT: EVERYTHING STARTS FROM THE SOFTWARE

Eugenio Capra, Politecnico di Milano, Dipartimento di Elettronica e Informazione, via Ponzio 34/5, 20133, Milano, Italy, eugenio.capra@polimi.it

Francesco Merlo, Politecnico di Milano, Dipartimento di Elettronica e Informazione, via Ponzio 34/5, 20133, Milano, Italy, merlo@elet.polimi.it

## Abstract

*In this position paper we discuss the importance of Green IT as a new research field that investigates all the environmental and energy issues related to IT and information systems in general. In particular we focus on the energy consumption of software applications, which is amplified by all the above IT layers in a data center and thus is worth a greater attention. By adopting a top-down approach, we address the problem from a logical perspective and try to identify the original cause that leads to energy consumption, i.e. the elaboration of information. We propose a research roadmap to identify a set of software complexity and quality metrics that can be used to estimate energy consumption and to compare specific software applications.*

*Keywords: Green IT; energy efficiency.*

# 1 INTRODUCTION

“Green IT” is an expression that indicates a new research field that investigates all the environmental and energy issues related to IT (Murugesan, 2008). More specifically, we think that Green IT may refer to three different research areas:

1. Energy efficiency of IT;
2. Eco-compatible management of the lifecycle of IT;
3. IT as an enabler of green governance.

The first research area aims at designing energy-efficient IT architectures and data centers, covering also all the effects that utilizations practices have on energy consumption. As we will explain in this paper, energy consumption impact on operating cost has grown in the last years and is now very significant. The second research area proposes to study new methodologies and technologies for eco-compatible manufacturing of IT components, to optimize packaging, and to minimize the environmental impacts of the whole lifecycle of IT. This includes eco-labeling and eco-compatible management and storage of waste and dismissed IT components. Finally, the third research area aims at leveraging IT as a means for measuring and monitoring the green parameters (e.g., energy consumption, temperature, toxic waste produced) related to all business processes, not limited to the IT area. This includes the design of monitoring devices as well as decisional support systems and dashboards to store, analyze, and compare green KPIs.

In this position paper we focus on the first research area and propose a research plan to analyze software energy efficiency. We illustrate as quantum physics theory offers an overall interpretation for the energy consumption of IT: elaborating information requires a minimum energy related to the quantum nature of the world. Actual consumptions are by far higher than the minimal theoretical level because current IT systems introduce a number of inefficiencies related to many different layers, which go from logical gates level to servers architecture and data centers infrastructure level. From another standpoint, this means that there are many possibilities for increasing the energy efficiency of IT, and this is a great challenge that the scientific community needs to face.

Software plays a crucial role in this scenario. Although software does not directly consume energy, it deeply affects the consumption of hardware equipment. Software applications, ranging from operating systems and drivers for hardware devices to decision support systems and ERP suites, indicate how information should be elaborated and to some extent guide the functioning of hardware. Consequently, they are indirectly responsible of energy consumption. In this paper we propose a research plan to measure and compare energy consumption of different applications and to correlate these data with traditional software design quality metrics.

The paper is organized as follows. Section 2 describes why Green IT is gaining more and more importance due to the devastating effects of energy consumption on the environment, on operative costs, and on scalability. Section 3 describes how energy consumption is distributed in a data center and illustrates how the energy actually used for chip-level computation is amplified by all the above IT layers. Section 4 presents the theoretical hypotheses at the base of our research, and Section 5 proposes our research plan. Finally, Section 6 concludes the work.

# 2 WHY GREEN IT IS IMPORTANT

Green IT is attracting more and more attention both in the scientific and business communities. In the past decades research and innovation have focused on increasing clock frequency and on miniaturization (Schaller, 1997), with only a marginal focus on power consumption, mainly associated with battery autonomy of laptop devices. This has resulted in extremely fast IT systems, but which consume a lot of energy that is very often inefficiently employed.

Energy consumption has devastating effects on:

1. Equivalent $CO_2$ emissions;
2. Operating costs;
3. Scalability.

As consumptions rise, the attention on Green IT gains momentum.

According to recent researches (Murugesan, 2008, Brown and Lee, 2007, Kumar, 2007), IT is responsible of more than 2% of global $CO_2$ emissions, and its environmental footprint is comparable to that of the aeronautic industry. The average amount of energy consumed by a PC in 1 year corresponds to the emission of 1 ton of $CO_2$, and a server has roughly the same annual carbon footprint as an SUV doing 5 miles-per-gallon (Restorik, 2007)0. In addition to that, 70% of the landfills of lead, cadmium and mercury derives from the IT industry (Brown and Lee, 2007).

From an economical perspective, whereas the cost of hardware has only slightly grown in the last 12 years, the cost of power and cooling has grown four times. Figure 1 shows data on the global spending for servers in the last  years and estimates for next years.

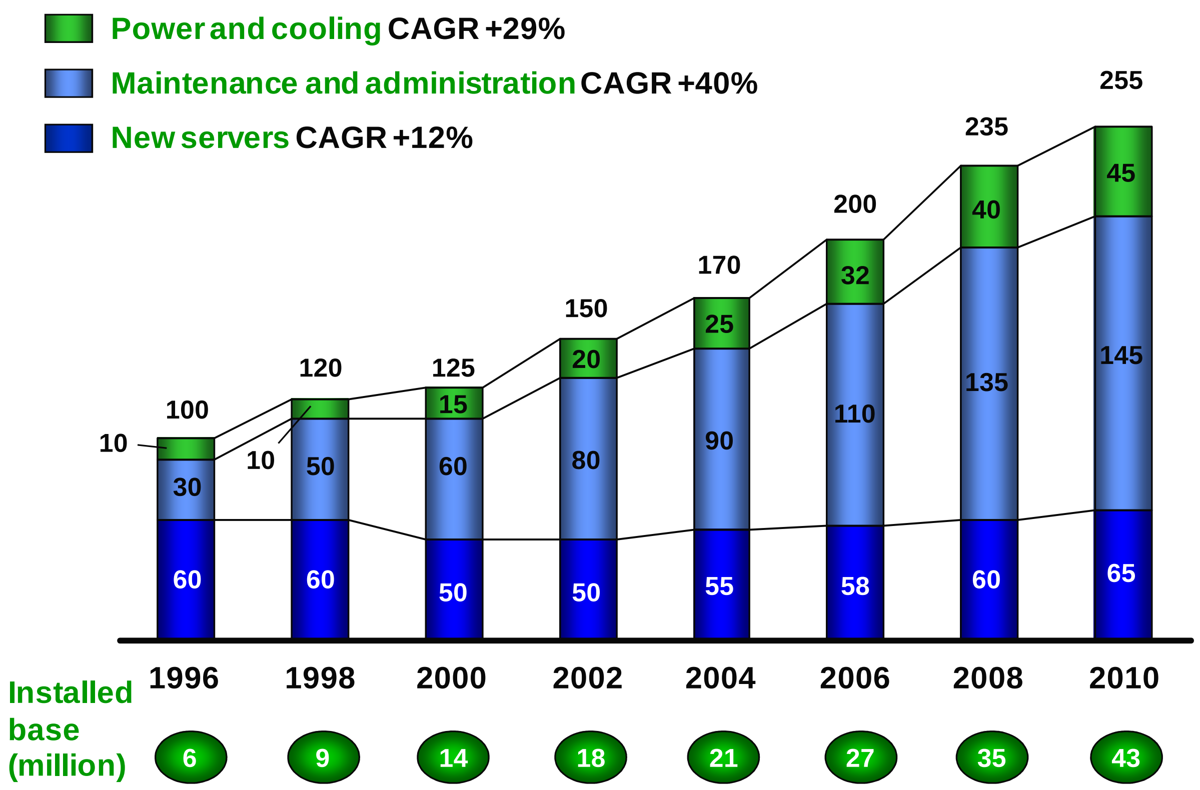


*Figure 1 – Global spending for server (Billion dollar, Source: Josselyin et al., 2006).*

Nowadays, power and cooling operating costs represent 60% of the total spending for new infrastructures, and consequently have a great impact on TCO (see Figure 2). This proportion is expected to rise even more, also because of the continuous growth of energy unit cost. As in most companies energy costs are not charged to the IT budget, the importance of this phenomenon is not yet fully perceived, but it is likely that accounting rules will change as the impact of energy costs on overall IT costs rises more and more.

In addition to that, energy consumption is a limit to the scalability of data centers. New IT equipment requires an extremely high quantity of energy per square meter (e.g., a rack with 5 blade servers of 8 units consume more than 20KW, as much as an apartment complex) and also the energy required by personal computers rises at a rate of 8-10% per year. When data centers are located in areas with high population density, as it often happens in Europe, it may be difficult for power distributors to bring the required energy in the same building. As power infrastructure modifications are difficult and expensive, data centers that are not energy efficient cannot expand their capabilities. According to Forrester Research (Brown and Lee, 2007), in the next few years 60% of data centers will be limited by power consumption, cooling, and space issues.

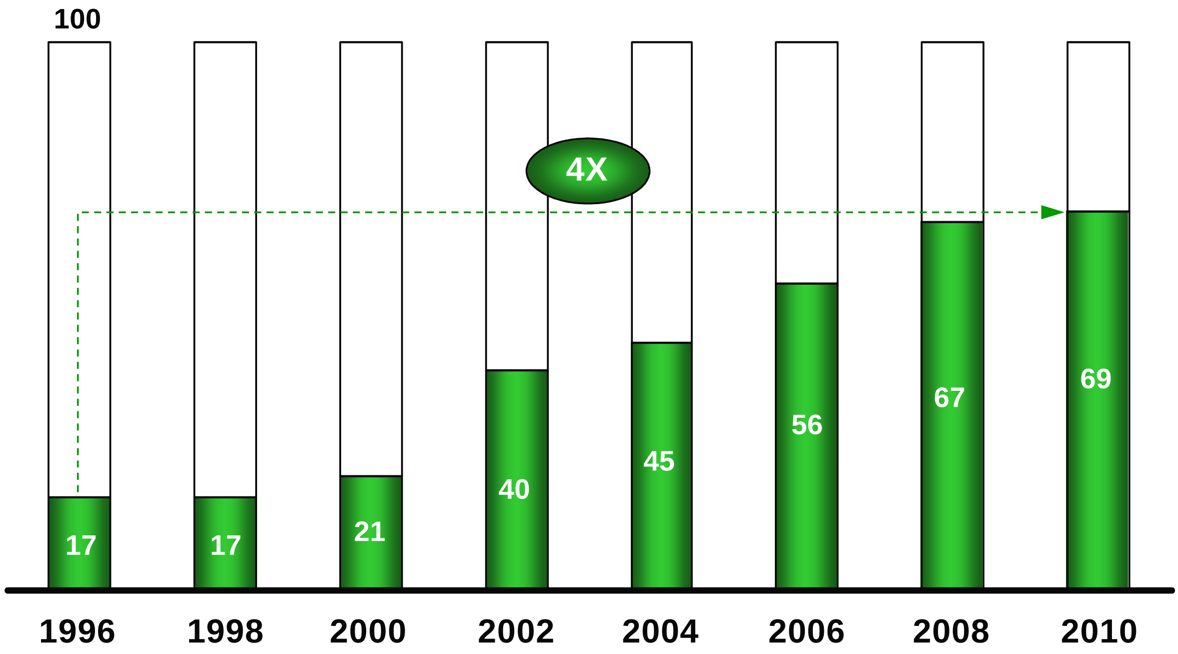


*Figure 2 – Spending for energy and cooling/ spending for new servers (Percent, Source: Josselyin et al., 2006).*

Recent surveys show that there are growing concerns about Green IT in corporate contexts: according to Forrester Research (Forrester Research, 2007)0, 33% of North American and 48% of European IT procurement and operations professionals think that environmental and energy-related issues are very important in planning their company's IT operations, whereas only 15% in North America and 6% in Europe think that Green IT is not a problem at all.

## 3 A MULTILAYER APPROACH

An average data center usually consumes at least 300 KW, whereas a large data center may consume more than 10MW. However, it is important to note that this significant quantity of energy is consumed at different *layers*, i.e. from different parts of the data center with different logical functions.

According to (Renzi, 2007), 40% of the energy consumed by a data center is absorbed by HVAC (cooling) and UPS (back-up batteries) systems, and another 42% is absorbed by fans, AD/DC transformers, and storage, whereas only 18% is consumed by the processors. In addition to that, as some processors stay idle for some time, the energy really used for computation may be as low as 3% of the total. Hence, researches and actions aimed at optimizing the energy consumption of data centers should address all the IT layers involved.

The reduction of power consumption should obviously focus on the optimization of the layers that consume the biggest part of energy, i.e. power and cooling and peripherals systems. This requires research also on non-IT items, such as UPS, air conditioning and other equipment. In addition to that, virtualization can greatly reduce the idle time of processors thus optimizing the energy consumptions. All these researches are specific to particular contexts and typologies of infrastructure.

In this paper we take a different and innovative perspective and we focus on the cause of energy consumption by information systems, independently from their infrastructural implementation. In the next section we illustrate how elaborating information *per se* requires energy, according to recent quantum physics researches. Quantum physics also quantifies the minimum theoretical amount of energy needed to commute a bit of information, which according to the current state of our knowledge could be optimally represented by the spin of an electron. Of course this minimal amount of energy is much lower than current consumptions. This gap between theoretical and actual consumptions is due to all the inefficiencies introduced by the different architectural layers of a computational system, e.g. because we use transistors rather than atoms to store and elaborate bits.

We posit that all these layers amplify the unitary amount of energy required to elaborate information, as all the hardware supporting a processor and the infrastructure in a data center are sized according to the amount of elementary computations required. This hypothesis will be verified during our research.

When a processor spends 1W to elaborate information the total energy consumed by the system may be as much as 28 times higher, due to drivers, memory, cooling, back-up batteries and all the other auxiliary components needed by the processor to work (see Figure 3). Thus, the benefits obtained by optimizing the energy consumed for computation are amplified by the above IT layers and have a great impact on the total consumption.

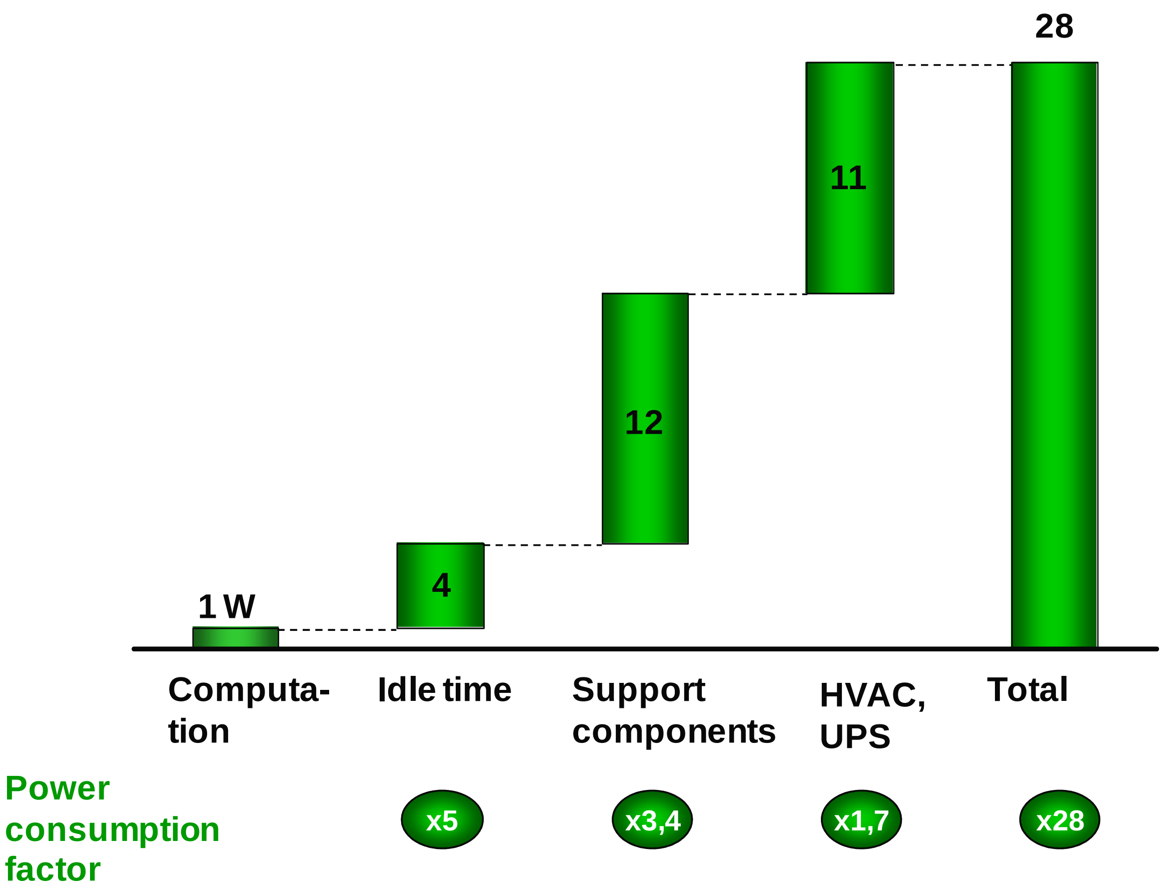


*Figure 3 – Energy absorbed by a data center (Watt, Source: Renzi, 2007)*

Accordingly, our research will focus on analyzing the energy efficiency of software algorithms, i.e. how efficiently information is elaborated, thus laying down the foundations for future optimizations.

Recent researches (Bruschi, 2007) have shown that current energy efficiency of algorithms and applications is on average 20%, whereas energy efficiency related to data quality (low quality data requires more operations) is no higher than 60% (Restorick, 2007).

# 4 THEORETICAL FOUNDATIONS

This section introduces the theoretical foundations of our research roadmap. We will assume both a physical and logical perspective. The former is essential to understand *how* energy is actually consumed by the technological infrastructure that is at the base of an information system; the latter will help us understand *why* energy is required to manage information.

From a *physical perspective* it is well known that the average power consumed by a microprocessor while running an application is $P = I \cdot V_{CC}$, where $I$ is the average current and $V_{CC}$ is the supply voltage. Since power is the rate at which energy is consumed, the energy consumption of a given application is the integral of the power consumption $P$ over time $t$, that is:

$$EC_{physical} = \int_t I \cdot V_{CC} \cdot dt \qquad (1)$$

Measuring energy consumption of an application by means of Expression (1) requires to measure $I$ and $V_{CC}$ on the hardware system actually used. Consequently, this kind of measures always refer to a specific microprocessor architecture. Moreover, from Expression (1) it is impossible to analyze *why* energy is consumed.

In order to solve this problem, we need to link the physical domain (i.e., electric energy consumption) to the logical domain. Energy consumption can be assessed by analyzing *why* a given application requires a certain amount of energy to produce the desired output. The following paragraphs introduce some theoretical definitions that are required to consider the problem of energy consumption from a *logical perspective*.

The Margolus-Levitin theorem (Margolus and Levitin, 1998) posits that the maximum frequency for the status commutation of a physical system is directly proportional to the total energy of the system itself. As a consequence, the minimum *commutation energy* required by a system to operate at a given frequency can be computed as:

$$E_{min}(f) = f \cdot h / 4 \tag{2}$$

where $f$ is the frequency, and $h$ is the Planck's constant. For example, if we represent a bit by means of the direction of an electron's spin, the commutation energy required at the frequency of 1 GHz (thus comparable to that of current desktop computers, considered that, for an average particle like an electron, the maximum commutation frequency is $f_e \cong 3 \cdot 10^{13}$ Hz) is $E_e \cong 5 \cdot 10^{-21}$ J.

The *thermodynamic depth* (Lloyd, 2006) is a property of each physical system: it is essentially a measure of the information required to describe, and consequently build, the system itself. This metric is related to the concept of entropy. It is well known that entropy is the measure of the level of disorder in a given system (Haddad et al., 2005). Assuming that a system can always be described by a string of bits (e.g., by describing initial speed and position of all its atoms), entropy is the number of bits of the system that are disordered and unavailable to produce work. Conversely, *negentropy* is the measure that quantifies the number of bits that are ordered and structured. For example, a human being has an high degree of negentropy, whereas a balloon full of helium is completely lacking of negentropy. If we want to describe a table we need a certain number of negentropic bits, but we do not need to describe the positions of all the billions of atoms of the table: these bits can stay entropic without affecting our description.

Based on these definitions, the thermodynamic depth is defined as the number of negentropic bits that have been used to build the system.

The *logical depth* (Lloyd, 2006) of a generic string of bits, that can be interpreted as the representation of a generic system (as well as the output of a computer application), is defined as the computational complexity of the most efficient program that is able to produce that output. In other words, it is the smallest number of elementary logical operations required to perform the computation that produces the desired string of bits.

A software application executes a certain number of computations on a defined number of bits in order to obtain a result. Applying the theoretical definitions discussed above, the energy consumption of a software application can be estimated from a logical perspective as:

$$EC_{logical}(f) = E(f) \cdot C_c \cdot T_d \tag{3}$$

where $E(f)$ is the energy required by a single bit status commutation at frequency $f$, $C_c$ is the computational complexity of the application that is executed and $T_d$ is the thermodynamic depth of the computation that is performed onto the problem representation. In other words, Expression (3) estimates the energy consumption by considering how much energy is required by a single bit status commutation ($E(f)$) that is applied on a given number of bits ($T_d$) for a given number of operations ($C_c$). Expression (3) allows to analyze the causes that lead an application to consume energy because it is elaborating information, without focusing on the physical and electrical mechanisms of consumptions.

First of all, we note that there is an unavoidable trade-off between energy and frequency: a faster system requires more energy. The minimization of energy consumption can be achieved by optimizing each of the three terms of Expression (3).

As discussed before, the minimum energy required for the commutation of a bit status at a given frequency is given by the Margolus-Levitin theorem. This is only a theoretical lower bound, which is valid if bits are represented by electrons' spins. As a matter of fact, several attempts of building a computing machine that uses the electron's spin to represent a bit have been made (e.g., Isaac Chuang at MIT has factorized the number 15 with a 7 qubit computer (Vandersypen et al., 20010). It should be considered that the energy $E_{min}$ is by far lower than the actual energy that is consumed to switch a bit in current computers based on transistors (modern architectures require approximately $10^{-14}$ Joules to commute a bit, and research are being carried on to reduce this energy to $10^{-16}$ Joules[1]). However, this is an hardware-related research area, and goes beyond the purposes of this paper.

The remaining two terms, namely computational complexity $C_c$ and thermodynamic depth $T_d$ can instead be faced from an information system perspective.

The minimization of the computational complexity required to produce a desired output can be obtained if the generic application $A$ that is executed has the minimum possible computational complexity, that is, exactly the logical depth $L_d$ of the required output.

The thermodynamic depth can be minimized by adopting the most efficient way of representing the problem and the data required to produce the desired output, that is the minimum thermodynamic depth $T_{d\text{-}min}$.

As a consequence, the lower bound of Expression (3) at a given frequency is given by:

$$EC_{min}(f) = E_{min}(f) \cdot L_d \cdot T_{d\text{-}min} \tag{4}$$

Just as the minimum energy of commutation given by the Margolus-Levitin theorem, also Expression (4) is only an ideal theoretical lower bound. In particular, the problems of writing an application with the minimum computational complexity required to obtain a desired result or stating which is the most efficient representation of a given problem are not trivial problems. For example, there exist problems for which we do not know whether the algorithms used to compute their solution are the most efficient ones (e.g., sorting algorithms). Furthermore, there exist whole classes of problems for which we do not even know if an efficient solution exist (e.g., the NP-complete problem class, if P≠NP). On the contrary, it is possible to design a methodology that allows the comparison of different applications from the efficiency of energy consumption point of view. By considering Expressions (1) and (3), we posit that:

$$EC_{physical}(f) \propto EC_{logical}(f) \tag{5}$$

That is, the energy consumption described from the physical perspective can be considered as a measurement proxy of energy consumption defined from the logical perspective. Accordingly, the comprehension and the optimization of application on the logical level should directly impact on the physical level, i.e. on the actual power absorption.

The following section presents in detail the research roadmap for the definition of such methodology.

[1] http://www.itrs.net.

# 5 A RESEARCH ROADMAP

Our research roadmap focuses on the definition of a methodology that allows the comparison of different applications from the efficiency of energy consumption perspective. Our research roadmap includes the following steps:

1. Comprehension of the problem and study of the state of the art.
2. Identification and operationalization of proxy metrics for computational complexity and thermodynamic depth of specific software applications.
3. Implementation of a tool that measures these metrics.
4. Measurement of actual power consumption.
5. Analysis of data and identification of the most representative proxy metrics.
6. Integration of the results in a software tool to support IT managers in assessing software energy efficiency.
7. Evaluation of the impact of energy costs on the Total Cost of Ownership of an application.

After a first step focusing on getting a more detailed comprehension of the problem, we plan to define benchmarking methodologies to compare different applications and, finally, to propose optimization approaches.

The definition of our methodology requires a thorough analysis of the boundary conditions for the execution of the applications that should be analyzed. Therefore we will perform our analyses on a number of different configurations of hardware infrastructures.

We assume that the commutation energy $E(f)$ of Expression (2) is constant for a given hardware setup (please note that also the frequency $f$ can be made constant for current hardware setup by disabling dynamic frequency adaptation mechanisms such as Intel SpeedStep or AMD PowerNow!). As a consequence, we will focus our analyses on the assessment of energy consumption inefficiencies caused by computational complexity $C_c$ and thermodynamic depth $T_d$.

From a theoretical perspective, the minimization of the computational complexity term would require to evaluate how far the actual computational complexity $C_c$ of a given application is from its lower bound, that is from the logical depth $L_d$ of the output that the application is intended to produce. However, as noted in Section 4, such solution is really hard to achieve, if not unsolvable at all (at least, given the current state of the art). First, it would require a general methodology for defining the computational complexity of a generic problem. Second, the actual computational complexity $C_c$ of the application should be properly determined. Third, a way to identify the shortest program that solves the problem should be determined (that is, define the logical depth $L_d$ of the problem). Fourth, a comparison between the values of $C_c$ and $L_d$ should be performed in order to evaluate how far the application is from the theoretical optimum. Since phases two and three cannot be completely automated, and would require to identify the minimum logical depth for each possible problem (which is a clearly not satisfiable requirement), we decided to focus on the definition of benchmarking methodologies to compare specific applications.

One of the first issues to be faced is to find suitable proxy metrics for computational complexity and thermodynamic depth of a given application.

Lloyd (2001) suggests a list of 42 different complexity metrics that could be used to characterize the complexity of a system from three different (yet complementary) perspectives: *a)* how hard is it to describe, *b)* how hard is it to create, and *c)* what is its degree of organization. Since our focus is to characterize the complexity of a *software* system, we plan to operationalize and apply such measures of complexity (or a subset of them) to software systems. Along with these new metrics, we are also going to consider classic software quality metrics as validation terms of comparison, such as the McCabe's Cyclomatic Complexity (McCabe, 1976), the Halstead's Software Science (Halstead, 1977)0, and the design quality metrics for object oriented systems proposed by Chidamber and Kemerer (1994) and Brito e Abreu (1995). These metrics are not direct measures of computational complexity, but of software design quality and cohesion. Although there is not yet any empirical proof that these metrics are correlated with computational complexity, an high quality software is usually well structured and its operations follow a logical flow. Software quality metrics may be proxies, or indirect measures, of computational complexity. This hypothesis will be verified during the research.

With regard to the thermodynamic depth term, we acknowledge that different representations of the same computational problem (e.g., the file structure adopted to store data) can be more or less efficient, as well as the fact that different ways of representing single bits can have different commutation energy requirements. However, these issues will be included in our future works. Given the current absence of consolidated metrics for thermodynamic depths, we will perform our analyses on software systems and data sets of comparable dimensions, so to be in a situation of comparable thermodynamic depth.

After the definition of the theoretical framework and the operationalization of variables, we will develop a tool to measure these metrics by analyzing the code of an application.

We will then assess which subset of our metrics best represent computational complexity and thermodynamic depth and thus could be used to operationalize Expression (4). This will require to compare the data gathered by our tool with measures of actual energy consumption.

The measurement of energy consumption ($EC_{physical}$) related to the execution of specific applications will be performed through the use of ammeter clamps. Such methodology is commonly adopted (e.g.,Isci and Martonosi, 2003) since it does not require particularly instrumented hardware, nor the definition of instruction-level energy consumption models for the target microprocessor (as done for example by Tiwari et al., 1994). Figure 4 shows the details of the power measurement setup that will be used to measure the actual power absorbed by a microprocessor.

We will conduct our experiments on a number of different hardware configurations, and for each configuration we will analyze the relationship between actual power consumption and the set of measured metrics. Metrics will be gathered for a sample of Open Source applications similar for domain, functionalities, and language. Different classes of applications will be considered and comparative analyses will be performed within each domain and for each specific hardware setting. Application classes will be selected according to relevance, usage (optimization is convenient only if usage is high), and also availability of a minimum number of Open Source projects. For example, ERP and DBMS systems are likely to be included in the analysis as they both respect all the criteria listed above. For each class specific scripts will be implemented to automatically generate benchmark workload and compare the consumption of the different applications. For example, for DBMS different kind of queries will be considered (e.g., CREATE/ INSERT/ DELETE on a test database with 20 fields and 1.000.000) and for ERP different activities flow will be created (e.g., create a new order, receive the goods in the warehouse, etc.).

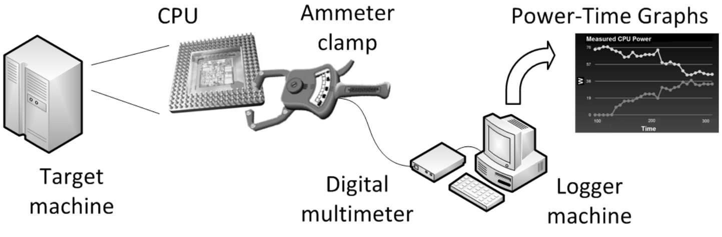


*Figure 4 – Power measurement setup.*

We consider the realization of a software tool that can help IT managers in assessing the power consumption efficiency of software applications the first milestone of our research roadmap.

Finally, we will evaluate the impact of energy cost on the Total Cost of Ownership of an application and, more generally, of an information system.

## 6 CONCLUSIONS AND FUTURE WORK

In this paper we have proposed a research roadmap to identify a set of software complexity and quality metrics that may be used to assess the energy efficiency of a specific application. We plan to validate our theoretical framework by measuring the actual power consumption on a number of different hardware systems. Our research will result in a tool able to extract a set of energy-related metrics by analyzing the code of an application. Our tool will allow to compare the energy efficiency of two or more applications with the same functionalities, thus enabling a green-aware choice. Project managers, software developers, and software buyers will greatly benefit from our research as they will be able to assess the differences in power consumption among a set of software applications. An issue that will need investigation is how energy consuming will be the code analyzer tool, as it may turn out that the energy required to analyze and optimize an application outweigh the savings obtained. However, optimization costs occur one time only, whereas savings are repeated every time the application is executed. Moreover, most of the currently available code-based metrics can be easily and quickly computed by parsers and code analyzer tools.

After the first phase of our research, we foresee that we will extensively apply our methodology to Open Source applications so that we will gather a significant quantity of data to analyze. Our research will then focus on optimizing the energy efficiency of applications by identifying development best-practices.

## References

N. Margolus, and L.B. Levitin, “The maximum speed of dynamical evolution”, Phisica D120, pp. 188-195, 1998.

S. Murugesan, “Harnessing Green IT: Principles and Practices”, IT Professional, vol. 10, no. 1, pp. 24-33, 2008.

R.R. Schaller, “Moore’s law: past, present and future2, IEEE Spectrum, vol. 34, no. 6, pp. 52-59, 1997.

E.G. Brown, and C. Lee, “Topic Overview: Green IT”, Forrester Research report, November 2007.

R. Kumar, “Important Power, Cooling and Green IT Concerns”, Gartner report, January 2007.

T. J. McCabe, “A complexity measure”, Proceedings of International Conference on Software Engineering, pp. 407, 1976.

T. Restorick, “ An Inefficient Truth”, Global Action Plan Report, 2007, www.globalactionplan.org.uk/research.aspx

S. L. Josselyin, B. Dillon, M. Nakamura, R. Arora, S. Lorenz, T. Meyer, R. Maceska, and L. Fernandez, “Worldwide and regional server 2006-2010 forecast”, IDC report, November 2006.

“Tapping Buyers’Growing Interest in Green IT”, Forrester Research report, May 2007.

F. Renzi, “Evolution scenarios for data centers technologies and architectures and impacts on energy consumptions”, IBM, Presentation held at the conference “Green ICT”, 22nd November 2007, Milan, Italy.

F. Bruschi, “From Green Grid to Efficient Enterprise”, Schneider Electric, Presentation held at the conference “Green ICT”, 22nd November 2007, Milan, Italy.

S. Lloyd, “Measures of complexity: a non exhaustive list”, IEEE Control Systems Magazine, pp. 7-8, August 2001.
M. H. Halstead, Elements of software science, Elsevier Computer Science Library, 1977.
S.R. Chidamber, and C.F. Kemerer, “A metrics suite for object oriented design”, IEEE Transactions on Software Engineering, vol.20, no. 6, pp. 476–493, 1994.
F. Brito e Abreu, “The MOOD metrics set”, Proceedings of ECOOP Workshop on Metrics, 1995.
W. M. Haddad, V. S. Chellaboina; S. G. Nersesov, Thermodynamics - A Dynamical Systems Approach, Princeton University Press, 2005.
S. Lloyd, Programming the Universe, Knopf, 2006.
L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, “Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance”, Nature, no. 414, pp. 883-887, 2001.
V. Tiwari, S. Malik, and A. Wolfe, “Power analysis of embedded software: A fisrt step towards software power minimization”, IEEE Transactions on Very Large Scale Integration Systems, vol. 2, no. 4, pp. 437-445, 1994.
C. Isci and M. Martonosi, “Runtime power monitoring in high-end processors: methodology and empirical data”, Proceedings of International Symposium on Microarchitecture, 2003.