Version: 0.3 / November 21, 2013

Dr Frank J. Furrer
frank.j.furrer@bluewin.ch

A Tentative Systems-Architecture for Requirements-Controlled Energy Management in a Computing Infrastructure

Summary

Energy consumption of computing systems is a growing concern. Especially for embedded, mobile systems the energy consumption may be a limiting factor for their use. Therefore, a modern research direction focuses on the energy-consumption management and optimization of computing systems.

Energy management/optimization can only be successful if the hardware, the systems software and the applications software cooperate in an intelligent way. To do so, an adequate systems architecture is required. This draft presents some principles of a tentative systems' architecture draft.

Comments, critique and ideas for improvement are very much welcomed by the author. Please address them to:

frank.j.furrer@bluewin.ch

Note: This draft paper can be found at http://st.inf.tu-dresden.de/teaching/fps13

Version: 0.3 / November 21, 2013

Contents

1. Context

In a coffee discussion with Prof. Uwe Assmann the author learned about the interesting TUD research related to energy-aware and energy-optimized computing platforms. One interesting topic under discussion is the systems architecture, especially the steering mechanisms between application software and the *energy management layer* of the computing infrastructure.

Although the author has no special knowledge in energy management, he introduces a concept which has proven very useful in many other areas: *Contracts*. The energy-relevant execution performance parameters can be propagated via contracts at run-time and used during run-time by the energy management layer of the computing infrastructure.

2. Introduction and Terminology

The key idea of this systems architecture draft is that the business processes[1] transmit their performance-relevant operating parameters at run-time to the applications which implement the corresponding functionality. The performance-relevant operating parameters of a business process step are e.g.:

- Maximum duration from start to end (Response time);
- Prescribed window of execution (Earliest start, latest termination).

These performance-relevant operating parameters may change for each execution, depending on context, user or current system-external conditions. If a short response time is requested, more computing power (e.g. in the form of a higher clock rate) must be provided for the execution of the business process step. If a

[1] "*Business process*" in this context means any step of an operational process. In banking it may mean the execution of a payment transaction, in automotive it may mean pushing the brake pedal, in industrial automation it may mean the positioning of a 5-D robot tool. A business process step therefore has a start point (triggered by an event), a duration, an end point and a functionality which is executed.

requested execution window is large enough, the execution can be scheduled to a time where system load is low.

The applications[2] are aware of their consumption of computing resources for the functionality requested by the business process step. The most important parameter is:

- The *worst case execution time* (WCET)[3] for the requested functionality.

Note that the WCET needs to be expressed in some suitable normalized form, e.g. on a specified processor with a specified clock rate[4]. With the parameter WCET the necessary computing power (under normalized conditions) is communicated to the energy management layer – thus allowing to calculate the corresponding computing power on the actual platform.

The applications have now the following information for the *current execution* which they can transmit to the energy management layer of the computing infrastructure:

- Maximum duration from start to end (Response time) or prescribed window of execution (Earliest start, latest termination);
- Their worst case execution times (WCET).

Based on this information, the energy management layer can optimize scheduling, resource assignment[5] and clock rate of the computing elements. The optimization function is (local) energy consumption minimization.

This situation is shown in Figure 1-1.

1. Architecture Proposal

Expanding the ideas in Figure 1-1 leads to a draft systems architecture shown in Figure 1-2. The new elements are the *contracts* on all levels. Note that we only show the contract elements needed for each *run-time execution* (e.g. each call or service request), i.e. the functional elements are agreed upon in *static contracts* and are not included here.

The business process step knows the *cost* of execution in relation to the requested performance (Cost function in the contract[6]). The business process step will therefore only request the minimum timing performance which is needed for this single service request from the underlying application software in order to minimize its cost.

---

[2] For many business process steps several applications may be executed

[3] WCET is an interesting concept: It expresses the longest expected execution time – i.e. the longest execution path through a program. Determining WCET bounds is a difficult task and many methods exist in literature

[4] Which can be transformed to any specification of a computing infrastructure

[5] A fully virtualized computing infrastructure is of great help in this situation

[6] This cost function may be static, i.e. long-lived or it may be dynamic, i.e. changing with the current state of the computing infrastructure

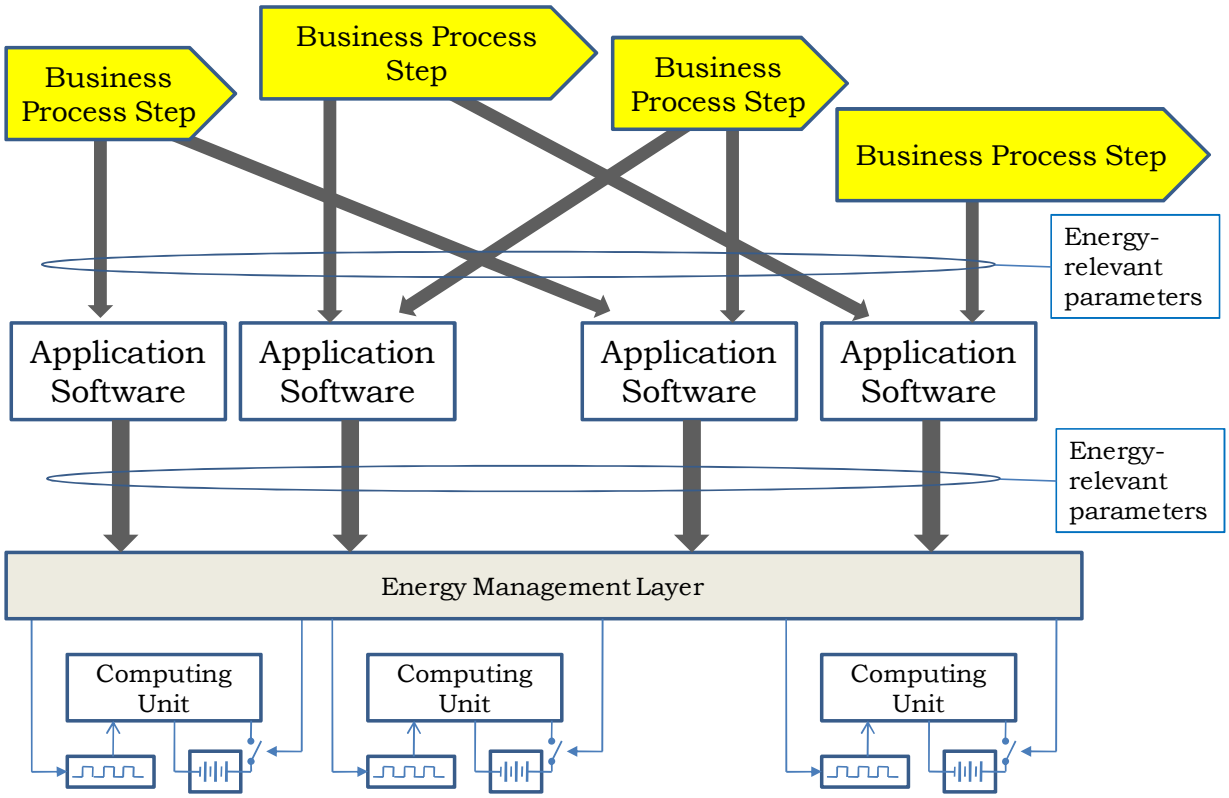

Figure 1-1: Transfer of Energy-Relevant Parameters

The individual applications know their worst case execution time (WCET) for the requested functionality, as well as the cost function in relation to the requested performance (Cost function in the contract). Again, the application will only request the minimum timing performance which is needed for this single service request from the computing infrastructure.

The energy management layer has now sufficient information to assign adequate, but minimum computing resources to guarantee the requested performance of the applications. This leads to a *local optimization* – i.e. an optimization based on individual requests.

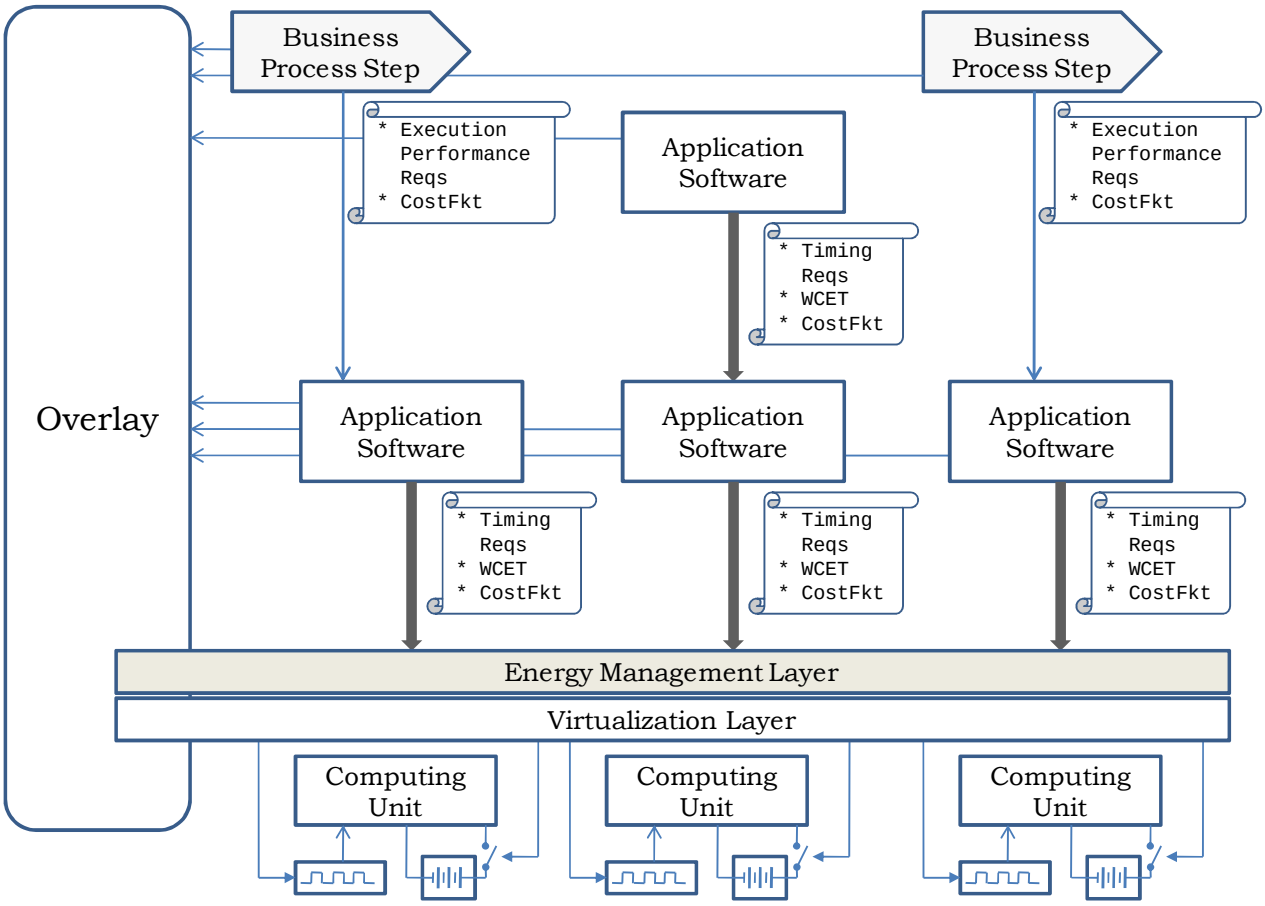

Figure 1-2: Draft Systems Architecture

2. Optimization Overlay

Figure 1-2 introduces another element: The *overlay*. So far, only *local optimization* is possible, i.e. energy optimization based on single, unsynchronized requests. In some cases, there may be a significant additional energy saving potential if the *global information* is available and can be used for scheduling and resource assignment, due to the availability of information from the business processes and the applications.

The overlay can therefore include higher level optimization algorithms, such as global scheduling or scheduling optimization[7]. This is especially useful in a fully virtualized computing infrastructure.

[7] Note that the overlay must under no circumstances become a *single point of failure*, but only an additional, safe optimization support

3. Conclusions and Outlook

This short draft introduced a tentative systems-architecture for requirements-controlled energy management in a computing infrastructure. Cost-aware execution performance requests are propagated via run-time contracts.

The author is aware, that many open questions have been raised by this draft and hopes that it has some merit in introducing the proven concept of contracts into this field.

Note: This draft and more information (e.g. the list of references) can be found at http://st.inf.tu-dresden.de/teaching/fps13