Abstractions to Support Interactions between Self-Managed Cells

Alberto Schaeffer-Filho and Emil Lupu

Department of Computing, Imperial College London
180 Queen's Gate, SW7 2AZ, London, England
{aschaeff, e.c.lupu}@doc.ic.ac.uk

Abstract. Management of pervasive systems cannot rely on human intervention nor centralised decision-making functions due to their complex and intrinsically mobile nature. In previous work, we proposed the concept of a self-managed cell (SMC) as an architectural pattern for building ubiquitous applications. A SMC consists of hardware and software components that form an autonomous administrative domain. SMCs may be realised at different scales, from body-area networks, to an entire room or larger settings. However, to scale to larger systems it is necessary for SMCs to collaborate with each other, to federate or compose in larger SMC structures. We describe here the main abstractions we have defined and explore future directions towards this goal.

1 Introduction

The complexity of pervasive systems inhibits a centralised or manual management approach. Such systems are saturated with technological capabilities that need to be integrated and work seamlessly. Typical pervasive environments consist of mobile devices, which cannot refer to a centralised management application. In addition, the complex and dynamic nature of such environments prevents any attempt of manual configuration. The feasibility of pervasive systems will depend on their ability to autonomously manage themselves, relying on local decision-making and feedback control-loops. In essence, this is the proposition of autonomic computing [1].

In previous work [2], we introduced the concept of a *Self-Managed Cell (SMC)* as an architectural pattern for building ubiquitous applications. A SMC consists of a set of hardware and software components that form an autonomous domain. SMCs monitor events of interest and perform actions when specific conditions occur, thereby adapting their configuration and operation to changes through a policy-driven feedback control-loop. We have used the SMC pattern in several application areas, such as health monitoring, management of autonomous vehicles, and management of large virtual organisations. This paper focuses on health monitoring applications where a body-area SMC of sensors and actuators monitors the medical condition of the patient and reacts to changes in the patient's condition or context. For example, changes in the patient's blood glucose level may trigger the activation of an insulin pump. Similarly, a cardiac

monitoring subsystem may trigger adaptations in its thresholds based on input from a physical activity monitoring SMC, trigger the activation of an artificial pacemaker, or contact emergency care if it detects an impending heart-attack.

Our main challenge is to define how SMCs can federate and collaborate with each other with little or no user intervention. Interactions between SMCs must be spontaneous, automated and may take the form of *peer-to-peer* collaborations or *compositions* where SMCs can operate and be managed within the context of a containing SMC. SMCs may represent individual devices, personal area networks, or even larger settings such as smart rooms. SMCs must autonomously decide whether and how to interact with discovered SMCs in their surroundings. These interactions are not limited to invocations between SMCs but must also include exchanges of events and policies between the SMCs in order to enable them to react to each other's behaviour. Due to the complexity of smart environments, SMCs need to compose into larger encapsulated structures, exposing their resources (including internal SMCs) only when they are relevant to surrounding SMCs.

This paper presents the first steps of this research, discussing the main abstractions we have defined to facilitate collaboration between SMCs. Ultimately, the collaboration between SMCs will allow services provided in the environment to be combined in order to achieve higher level goals. This will require goal refinement and planning-based techniques.

Although several studies have proposed frameworks for pervasive spaces [3, 4], they tend to share two limitations: they focus on pervasive spaces of a relatively fixed size (e.g. a room) and they fail to cater for dynamic interactions between pervasive spaces. In contrast, we consider the SMC as an architectural pattern applicable at different levels of scale, ranging from small body-area networks, to large-scale virtual organisations. SMCs are expected to dynamically discover and collaborate with other SMCs, whilst most other projects focus on a single-size, single-instance perspective.

2 Self-Managed Cells and their Interactions

A SMC comprises a dynamic set of management services that are integrated through a common publish/subscribe *event bus*, which supplies the basic communication infrastructure between the SMC's components (Figure 1.a). Together, the *event bus*, the *policy service* and the *discovery service* provide the core functionality of a SMC, as they are sufficient to implement a policy-driven feedback control-loop (Figure 1.b) [5]. The discovery of new components or changes of state in the current resources are published on the event bus and trigger the execution of *obligation* policies in the form of *event-condition-action* rules. Such policies define the actions that must be performed in response to events, thereby adapting the SMC to context changes.

However, in ubiquitous environments, where smart entities may range from a body sensor or personal belonging to a room or an entire building, SMCs have to interact and collaborate in different ways. Because such environments are sat-

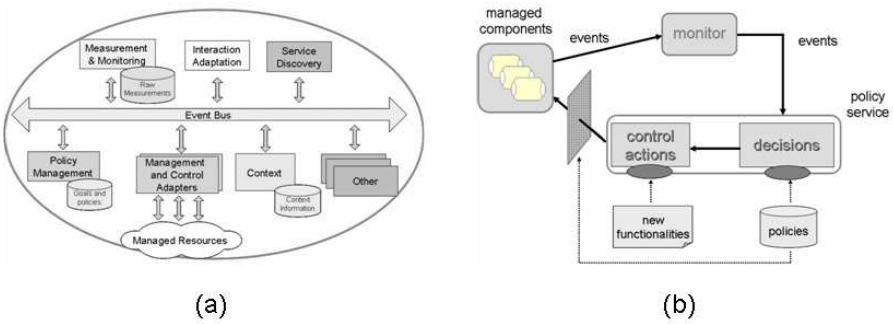

Fig. 1. The SMC core services (a) implement a feedback control-loop (b)

urated with technological capabilities, the ability to encapsulate resources and hide underlying complexity is crucial to their scalability. Therefore, we have investigated how SMCs could *compose* into complex structures whilst preserving the autonomy of their components. A composition interaction encapsulates a SMC (with its own resources) as a managed resource within a containing SMC. Functionality is exposed to external interacting SMCs through *customised interfaces* which are specific to them. For example, a patient SMC would expose access to its sensors to doctors, but hide the sensors from other patients. These customised interfaces also provide the ability to mediate and filter interactions with external SMCs. In pervasive systems, interactions and mediation aspects must be determined at run-time and must change dynamically in order to adapt to new circumstances such as failure of a sensor or discovery of a new one. This presents new challenges when compared to component composition in distributed systems where composition is often statically defined.

In order to perform complex interactions, SMCs can exchange obligation policies with each other. The set of policies that defines the behaviour of an interaction is called a *mission* and it specifies how a remote SMC should behave within the context of the interaction in terms of sending notifications, and reacting to events by invoking actions. For example, upon discovery of a patient SMC, a doctor SMC may load into the former an ECG monitoring mission, containing policies that perform heart beat readings at a specific frequency for a given amount of time, sending partial reports to the doctor's office every six hours and, in the occurrence of abnormal conditions, setting the alarm off in the nurse station. The term mission suggests that collaborations between SMCs can be done in terms of high-level goals. However, the ability to endow SMCs with planning capability on relatively small devices remains part of our future work.

SMCs discover each other at run-time, but policies defining how they should interact with discovered SMCs must be specified beforehand. For example, the doctor should be able to specify the mission in advance, and load it dynamically into the patient SMC when the latter is discovered. We have introduced *roles* as placeholders for remote SMCs yet to be discovered. Roles are associated with the missions and define the set of functions that a SMC of a specific type (e.g.

patient) is expected to provide. Thus, roles provide a scope for specifying the policies of a mission. This is somewhat different from the use of roles in Ponder (where there is no notion of expected behaviour from external components) or IETF PCIM (where roles define capabilities of policy targets). When remote SMCs are discovered they are assigned to their respective roles, and missions specified by the discovering SMC are instantiated on these remote SMCs.

3 Current Status and Future Work

Our prototype covers the implementation of obligation policies in the policy interpreter, exchange of customised interfaces and deployment of missions. Future work will address collaboration definitions whose enforcement is itself distributed and collaborations based on exchanges of high-level goals. A strategy for goal refinement and planning in a SMC-rich scenario will have to be conceived, in order to provide a complete solution for interactions between SMCs.

4 Concluding Remarks

This paper has briefly described a set of key abstractions to facilitate collaborations between SMCs. Allowing SMCs to dynamically compose into more complex structures caters for larger pervasive applications. Customised interfaces allow SMCs to selectively hide their complexity, exposing internal components only when they are relevant to specific partners. Finally, the same event-condition-action rules that provide to the SMC its ability of self-management can be grouped into missions and used across multiple SMCs, extending their local control-loop to involve remote SMCs available in the surroundings. Abstractions such as roles, customised interfaces and policies are not inherently novel, however the ways in which they can be combined and used to support interactions between autonomous SMCs remains a challenging task that requires further work.

References

1. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. IEEE Computer **36**(1) (January 2003) 41–50
2. Lupu, E., Dulay, N., Sloman, M., Sventek, J., Heeps, S., Strowes, S., Twidle, K., Keoh, S.L., Schaeffer-Filho, A.: AMUSE: autonomic management of ubiquitous systems for e-health. Special Issues of the Journal of Concurrency and Computation: Practice and Experience, Wiley (to appear)
3. Roman, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R., Nahrstedt, K.: A middleware infrastructure for active spaces. IEEE Pervasive Computing **1**(4) (Oct.-Dec. 2002) 74–83
4. Johanson, B., Fox, A., Winograd, T.: The interactive workspaces project: experiences with ubiquitous computing rooms. IEEE Pervasive Computing **1**(2) (April-June 2002) 67 – 74
5. Sloman, M.: Monitoring Distributed Systems. In: Network and Distributed Systems Management. Addison-Wesley (1994) 303 – 347