Self-Managed Cells and their Federation

Joseph Sventek[1], Nagwa Badr[1], Naranker Dulay[2],
Steven Heeps[1], Emil Lupu[2], Morris Sloman[2]

[1]University of Glasgow, Department of Computing Science,
17 Lilybank Gardens, Glasgow G12 8RZ, Scotland
{joe,nagwa,heeps}@dcs.gla.ac.uk
[2]Imperial College London, Department of Computing, 180 Queen's Gate,
South Kensington Campus, London SW7 2AZ, England
{n.dulay,e.c.lupu,m.sloman}@imperial.ac.uk

Abstract. Future e-Health systems will consist of low-power, on-body wireless sensors attached to mobile users that interact with a ubiquitous computing environment. This kind of system needs to be able to configure itself with little or no user input; more importantly, it is required to adapt autonomously to changes such as user movement, device failure, the addition or loss of services, and proximity to other such systems. This extended abstract describes the basic architecture of a Self-Managed Cell (SMC) to address these requirements, and discusses various forms of federation between/among SMCs. This structure is motivated by a typical e-Health scenario.

1 Introduction

Monitoring chronically ill patients as they go about their normal activity enables early release from hospitals and improves the patients' quality of life. Analysis and data mining of the monitored information can be used to predict potential problems (such as a possible heart attack for a specific patient being monitored) and to generate a warning to the patient or medical staff; the information can also be used by medical researchers to understand body changes that take place prior to a specific problem. The technology to enable *healthcare everywhere*, such as programmable body sensor nodes that use wireless communication to PDAs/phones that interact with remote medical centres for logging and patient feedback, is now available [1]. On-body and environmental sensors may also be used in the home for monitoring elderly people to determine problem situations or deterioration of well-being over time [2]. However, configuration of the multiple sensors and software components that form an adaptive body-area network or a home monitoring network is not currently feasible for non-technical patients or medical staff.

Existing network and systems management frameworks do not cater for ubiquitous environments, although specific techniques for monitoring and event correlation, service discovery, quality of service and policy-based management can be used to some degree. Current frameworks are aimed at large-scale corporate environments, telecommunications networks and internet service providers. Their architecture is

based on functional decomposition where the various functions are integrated in centralised network operations centres by human administrators. For self-management in ubiquitous systems to become a reality, it is necessary to define and implement architectures which can scale down to small lightweight structures with local decision making capabilities. The management functionality must be automatically integrated and adapted to the specific application requirements without human intervention, as the management of a body-area network will be much simpler than that for a network service provider. Autonomous, self-managed cells must be composable to form larger cells but also need to collaborate and integrate with each other in peer-to-peer relationships as well as across multiple levels of abstraction relating to hierarchical service relationships.

We are developing autonomic management techniques for self-configuring and self-managing such systems. The systems must be able to add or remove components, cater for failed components and error-prone sensors, automatically detect and adapt to a user's current activity and communication capability as well as catering for interaction with health visitors or other medical staff who attend patients or visit elderly people. User activity determination is critical, for example to distinguish between vigorous brushing of teeth and a possible heart attack. Although we are using e-Health as a driving scenario, our aim is to develop generic techniques applicable to e-Science and other applications.

2 Related Work

This work builds upon related work in four separate areas: e-Health, autonomic computing, policy-based management, and federation.

2.1 e-Health

Future e-Science and e-Health applications will involve mobile users, possibly with on-body sensors interacting with a ubiquitous computing environment which detects their activity, current context and adapts accordingly. Continuing advances in the miniaturisation and bio-compatibility of physiological sensors enables the realisation of future ubiquitous computing environments which can dramatically enhance the healthcare provided in the community to individuals with chronic conditions and to improve their quality of life. However, while devices for insulin delivery, multi-programmable brain stimulators and implantable cardioverter-defibrillators (ICD) have been developed and are soon to be endowed with wireless communication capabilities, the software infrastructure which allows their secure interconnection, configuration and adaptation to current context remains a significant challenge.

As an example, the Ubimon project at Imperial College [1] is producing such sensors while pursuing the following objectives:

- Techniques for portable communicator interactions with implantable sensors and interventional devices.
- Multi-sensor interfacing using a wearable communicator.

- Automated techniques for integrating multi-sensory data leading to an intervention strategy.
- Preliminary clinical evaluation for management of patients with ischaemic and arrhythmic heart disease.

Currently, two projects of interest to our work are active – UbiMon focuses on multi-sensory fusion of context information for monitoring and prediction of cardiac problems, while ANS focuses on autonomic configuration of light-weight software components. We are working closely with these projects, using UbiMon as a healthcare demonstrator and making use of the component technology and context sensing developed in ANS.

2.2 Autonomic Computing

IBM has been the prime mover towards autonomic computing. Since Paul Horn, IBM's director of research, presented the grand challenge for the IT industry to address autonomic computing, there has been substantial activity in selected portions of the space [6,7].

According to IBM, there are four fundamental aspects of autonomic computing:

- Self-configuration – systems adapt automatically to dynamically changing environments.
- Self-healing – systems discover, diagnose, and react to disruptions.
- Self-optimizing – systems monitor and tune resources automatically.
- Self-protecting – systems anticipate, detect, identify, and protect themselves from attacks from anywhere.

HP is also addressing similar issues in its provision of on-demand Utility Data Centers [8].

The architecture proposed for self-* systems in these efforts is focused on large computational facilities; as such, the architectures are very heavy-weight, unsuitable to the types of e-Health systems that we are attempting to build. As a result, our architectures differ. Note that one of our goals is that our architecture will be flexible enough to scale from body-area network, e-Health applications to large-scale, health monitoring applications over large geographical regions (e.g. the entirety of the United Kingdom). Therefore, in the limit of large scale, our architecture should approach these autonomous/adaptive computing architectures.

2.3 Policy-based Management

"Policy-based management is an administrative approach that is used to simplify the management of a given endeavor by establishing policies to deal with situations that are likely to occur. Policies are operating rules that can be referred to as a means of maintaining order, security, consistency, or other ways of successfully furthering a goal or mission. In the computing world, policy-based management is used as an administrative tool throughout an enterprise or a network, or on workstations that have multiple users. Policy-based management includes policy-based network management, the use of delineated policies to control access to and priorities for the

use of resources. Policy-based management may be used in systems management, or the creation and operation of an efficient computing environment."[1]

The Policy Research Group at Imperial College has pioneered the development of policy specification languages [3], policy deployment models and infrastructures [4], and security policies [5], all of which are relevant to this research effort.

2.4 Federation

The term federation, usually used in reference to databases, refers to an architecture in which middleware provides uniform access to a number of potentially heterogeneous, managed resources [9]. Each individual resource remains autonomous, and determines the level of exposure of its resources to federated components. Successful federation requires that potential federates agree on a number of concepts/technologies: ontology, abstraction level, security and privacy, and interface.

Application of these concepts to self-managed systems, such as SMCs, is novel. Of special interest are the many and varied types of federation that can take place between SMCs, not all of which are classic peer-peer federation relationships.

3 Basic SMC Architecture

A self-managed cell consists of a set of hardware and software components that form an administrative domain that is able to function autonomously and thus capable of self-management. An SMC could represent the resources available in a PDA, a body area network of physiological sensors and controllers, as well as the application components relating to a set of collaborating partners forming a virtual (e-Health) organisation spanning multiple countries. In each case, SMCs must be automatically configured with the required management services, appropriate to the scale and environment of the cell. These services interact with each other through asynchronous events exchanged over an event bus (see Figure 1). As a minimum, an SMC should contain functionality for measurement and event correlation and support for policy-based control, where policies should specify which adaptation should occur in response to changes of state in the managed resource or changes of context in the environment. In essence, an SMC is a *closed-loop* system where changes of state in the resources trigger adaptation which in turn affects the state of the system. In a ubiquitous environment, the SMC also includes management components that provide contextual information and service discovery (see Figure 1).

[1] Definition from Whatis.com, http://whatis.techtarget.com/definition/0,,sid9_gci537241,00.html, accessed 29 April 2005.

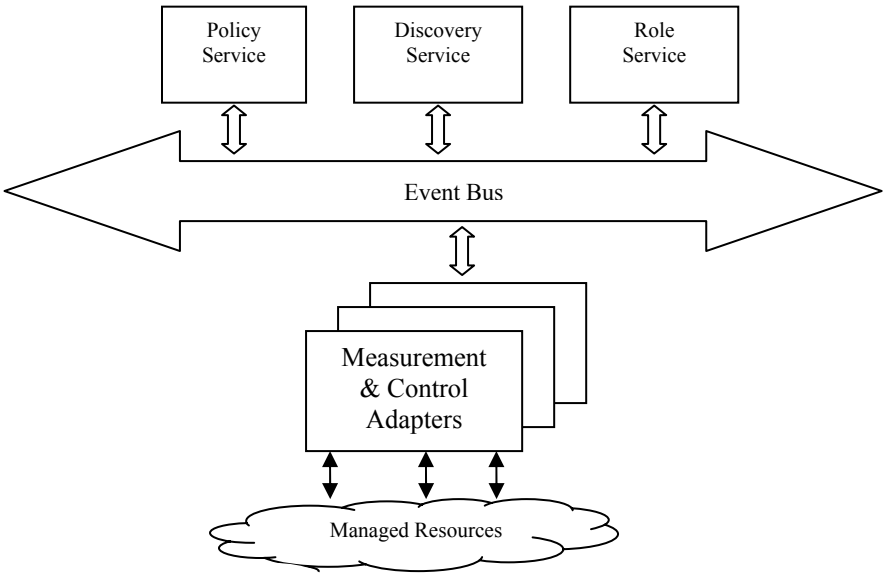

Figure 1. A Basic Self-Managed Cell (SMC)

Note that the list of management services which are shown in Figure 1 is by no means exhaustive. As the SMC closed-loop pattern is applied to larger structures such as organisational networks, other management services such as accounting, resource planning, optimisation and analysis would become essential. In fact, one of our most important challenges is to define the SMC architecture such that it is extensible, and can be specialised to particular environments such as personal area networks, appliance networks, as well as large scale distributed applications. The responsibilities of the core services shown in Figure 1 are described below.

3.1 Event Bus

Adaptive ubiquitous systems are essentially event-driven, as changes of state in resources need to be notified asynchronously to several, potentially unknown, recipients. An event may indicate discovery of a new component, component failure, or change in context or medical condition (e.g. ECG anomaly detected). As described in section 5 below, we implement the event bus as an at-most-once, persistent publish/subscribe delivery service.

3.2 Discovery Service

It is essential to discover nearby components that are capable of being members of the SMC – e.g., intelligent sensors and other cells when they come into communication

range. The discovery service interrogates the new components to establish a profile describing the services they offer and then generates an event for other SMC components to use, as appropriate. The service must distinguish between transient communication failures, which are common in wireless communication systems, and permanent departure from the cell.

3.3 Role Service

Roles provide a means of grouping components that generally corresponds to the role they play within the cell – e.g. temperature sensors, heart sensors, or context sensors. Roles can be considered active components that receive discovery events and decide whether to assign components to a role which will govern what cell resources can be accessed and how the component interacts with other cell components.

3.4 Policy Service

Policies provide the means of specifying the adaptation strategy for autonomic management. Based on the lessons learnt from previous policy specification work [3,4,5], we are developing a new, light-weight policy service appropriate for limited-resource devices. Authorisation policies specify what resources the components assigned to a role can access, and obligation policies (event-condition-action rules) specify how roles react to events and interact with other roles. When a device is assigned to a role, the appropriate polices are deployed to it. Policies can be added, removed, enabled and disabled to change the behaviour of cell components without reprogramming them. Policies also govern the behaviour of the discovery service, the role service, and the policy service itself, enabling these to be tailored to specific situations.

3.5 Management and Control Adapters

Management and Control Adapters are proxies that map devices/components into the SMC. They enable events raised by such devices/components to be signalled across the Event Bus for consumption by interested services; they also provide a signalling channel to enable obligation policies to manipulate devices/components in response to a particular management event.

4 SMC Federation

We are just beginning to study the federation of SMCs. Since SMCs are event-driven, the ontology that determines the domain of discourse between potential federates is the language of events. Policies are normally specified with regards to a triggering event, conditions that must be true when the event is detected, and a set of actions that are to be applied to the system (event-condition-action rules). Federation

of two SMCs will most likely entail the partial sharing of events – i.e. a component in one SMC can register interest in an event generated by a component of the other SMC.

There are many situations in the e-Health environment in which the two SMCs are not peers (e.g. the health visitor/patient scenario described in section 6 below). Therefore, there will be an asymmetry in the level of sharing from each SMC, such that the dominant partner will have access to more of the subservient partner. This indicates that the federation negotiation between two SMCs can be heavily dependent upon the role played by the owner of each SMC.

5 Architectural Details

In this section we document some of the details of the architecture.

5.1 Event Bus

As asserted in section 3.1 above, we have chosen to implement the event bus as an at-most-once, persistent publish/subscribe delivery service, using a router. The router is content-based – i.e., a subscriber specifies a filter when it registers, and all published events that match the filter are forwarded to that subscriber. The discovery service is concerned with keeping track of SMC membership; if it determines that a particular service has left the SMC, it will raise a "purge subscriber" event, which causes the Router to remove that subscriber's filters (if it had registered for any events), and purge any queued up events for that subscriber. The structure of the event bus is shown in Figure 2.

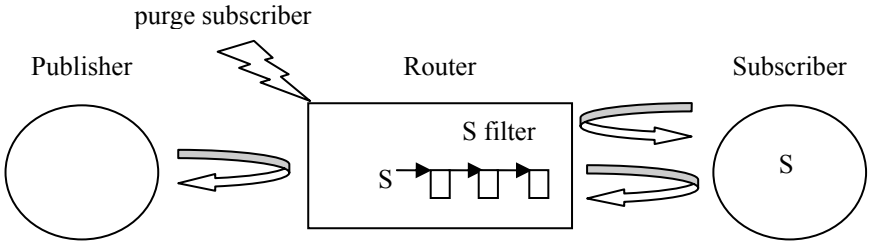

Figure 2. Persistent, best-effort event delivery

Publishers do not need to register with the Router. When a publisher sends an event to the Router, it does so synchronously and reliably; this reliable delivery is shown by the request/response arrow. Successful delivery of the event to the Router transfers responsibility for subsequent delivery to the Router.

The Router attempts to deliver such an event to each subscriber whose filter matches the event. If it is unable to deliver the event to a particular subscriber due to transient communication failure, it queues the event up for redelivery to that

subscriber. The router attempts to deliver queued events until it knows that the subscriber is no longer a member of the SMC.

Each subscriber is guaranteed to receive all events from a particular publisher in the same order as received by the Router. This is required in case there is a causal relationship between events from a particular publisher.

5.2 Discovery Service

The discovery service is responsible for detecting new devices or other SMCs when they come into communication range. It is responsible for vetting a device for membership by obtaining the device's profile, as well as performing any required authentication. If a new device passes the vetting procedure, the discovery service generates a new cell member event.

This service is also responsible for determining when a device permanently leaves the cell. When it determines that this has occurred, it generates a cell member left event.

Note that the discovery protocol used does NOT use the event system; this enables it to act as a veneer for existing discovery services [10], or to implement new discovery services needed for particular applications.

The general functionality of the discovery protocol is represented by the following points:

- the cell is centred around the event bus router
- the device that contains the router broadcasts its identity message at frequency $\omega_{\mathbf{R}}$; this identity message has the form "id; type[; extra]"
- other devices respond to the router identity message with unicast device identity messages
- the router device carries out the vetting protocol with each of these other devices that are not already members
- after the other device knows that it has been granted membership, it unicasts its identity message at frequency $\omega_{\mathbf{D}}$
- if the router device misses $n_{\mathbf{D}}$ successive device identity messages, it declares the device to have forfeited its membership in the cell
- if the other device misses $n_{\mathbf{R}}$ successive router identity messages, it declares that is no longer a member of that cell

6 Typical Scenario

We focus on a body sensor network SMC monitoring a cardiac patient and how it interacts with a nurse who visits the patient at home. Each SMCs discovery service periodically broadcasts a request to discover new components. Any *discoverable* component that is not a member of the cell responds with a message giving its identifier, address, and device type. The device can then be queried to obtain its profile, which describes the services it implements, authentication certificates, etc. Components will have policies specifying how to respond to discovery requests. For

example, a sensor component will only respond if it is not already a member of a cell, whereas a nurse might only respond to a patient whose identifier is in a preloaded patient list. Some example policies include:

1. oblig on compDetect (compName, compRef, compType) =>medicRole.assign(compName, compRef)
   when compType = nurseT and signed (compRef.getCertificate, nursingCouncil_PK)
2. oblig on compDetect (compName, compRef, compType) => temperatureRole.assign (compName, compRef)
   when CompType = tempSensorT

   • • •
3. oblig every mins(1) => discoveryService.findNewMembers()
4. oblig every mins(2) => discoveryService.pollMembers()

This set of policies relates to the device discovery, and specifies that new *nurse type* devices certified by the nursing council should be assigned to the medic role while temperature sensor devices should be assigned to the temperature role. Policy 3 indicates that the frequency of polling for new devices and policy 4 specifies that devices in both roles should be polled to confirm that they are still reachable. pollMembers() is a role-specific action which maintains records of device presence and removes those devices that have been unreachable for some period of time, e.g. 10 minutes.

When a device is assigned to a role, any policies applicable to that role will be loaded onto the device and enabled. Thus a temperature sensor would receive policies instructing it to take a temperature reading every <n> minutes but only raise an event when the temperature exceeds a threshold. The cardiac sensors may send a 10-second ECG sample every minute to the PDA for anomaly detection; however, under some circumstances, it may switch to continuous monitoring with remote transmission via the PDA to a medical centre. We assume that devices capable of wireless communication are also capable of interpreting simple policies. When the nurse is assigned to the medic role, its PDA will receive policies relating to interacting with the patient's devices – e.g. to obtain device readings directly. The Patient SMC will have authorization policies permitting members of the medic role to interact with its devices. The patient's SMC could also be made a member of the Medic SMC. The nurse could then replace, enable, or disable policies in the patient's SMC to modify the medical regime. This type of peer-to-peer relationship is one form of SMC federation.

7 Status and Future Work

We are currently building an implementation of the SMC architecture. The prototype consists of a set of body sensor nodes with Zigbee [11] wireless capability developed in the UbiMon project [1] that communicate by low-power radio with a PDA that hosts the management components and has wireless LAN or GPRS communications. We are also building an SMC simulator which will enable larger SMCs to be tested and more easily simulate repetitive events or devices coming into and out of range. There are a number of issues still to be resolved, such as making sure the protocols we develop optimise the use of battery power; how to make sure a device is 'owned' by a particular SMC and cannot be taken over by another SMC; how SMCs can interact at

a level of abstraction higher than implementable policies etc. There are a number of related projects starting which will investigate the privacy, trust and security issues, the applicability of the concepts to SMCs consisting of groups of unmanned vehicles which need to cooperate to achieve an overall goal; and micro-miniaturised, potentially implantable, sensors being developed in another EPSRC project.

Acknowledgements

The authors wish to thank the UK Engineering and Physical Sciences Research Council for their support of this research through grants GR/S68040/01 and GR/S68033/01.

References

1. Ubiquitous Monitoring Environment for Wearable and Implantable Sensors, http://www.ubimon.org, accessed 29 April 2005.
2. Care in the Community, A virtual research centre of the DTI Next Wave Technologies and Markets Programme, http://www.dticareinthecommunity.com, accessed 29 April 2005.
3. Damianou N., N. Dulay, E. Lupu, M Sloman, "The Ponder Specification Language", Proc. Policy 2001: Workshop on Policies for Distributed Systems and Networks, Bristol, UK, 29-31 Jan. 2001, Springer-Verlag LNCS 1995, pp. 18-39.
4. Dulay N., E. Lupu, M. Sloman, N. Damianou, "A Policy Deployment Model for the Ponder Language", IEEE/IFIP International Symposium on Integrated Network Management VII (IM'2001), Seattle, May 2001, IEEE Press. pp. 529-544.
5. Sloman M., E. Lupu "Security and Management Policy Specification", IEEE Network Special Issue on Policy,16:2, March 2002, pp.10-19.
6. Kephart J, Chess D, The Vision of Autonomic Computing, IEEE Computer, Jan 2003, pp 41-50.
7. Autonomic Computing Special Issue, IBM Systems Journal, Vol. 42, No 1, 2003.
8. "HP Utility Data Center: Enabling Enhanced Datacenter Agility", http://www.hp.com/large/globalsolutions/ae/pdfs/udc_enabling.pdf, May 2003, accessed 29 April 2005.
9. D. Heimbigner and D. McLeod, "A federated architecture for information management", ACM Transactions on Office Information Systems, vol. 3, no. 3, 1985.
10. Universal Plug and Play Device Architecture, http://www.upnp.org/resources/documents.asp, accessed 29 April 2005.
11. ZigBee Alliance, http://www.zigbee.org/en/index.asp, accessed 29 April 2005.