Policy-based Management for Body-Sensor Networks

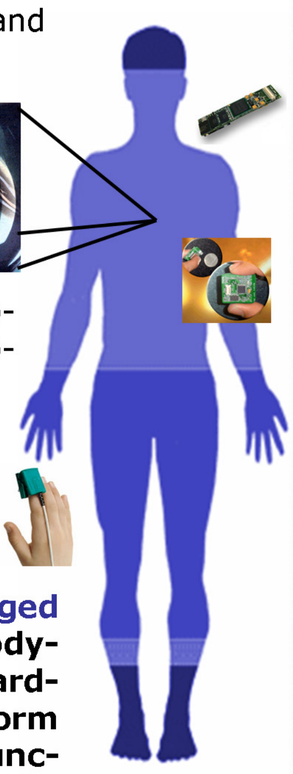
S.L.Keoh[1], K.Twidle[1], N.Pryce[1], A.E.Schaeffer-Filho[1], E.Lupu[1], N.Dulay[1], M.Sloman[1], S.Heeps[2], S.Strowes[2], J.Sventek[2], E.Katsiri[1]

[1]Department of Computing, Imperial College London, UK
[2]Department of Computing Science, University of Glasgow, UK

Introduction

Body sensor networks composed of sensors and actuators can monitor the health of a patient.

This supports the assessment, monitoring and treatment in everyday environments in order to facilitate "health care in the community".



Such environment must be secure, autonomous, and continuously manage their own evolution and configuration changes. For example:

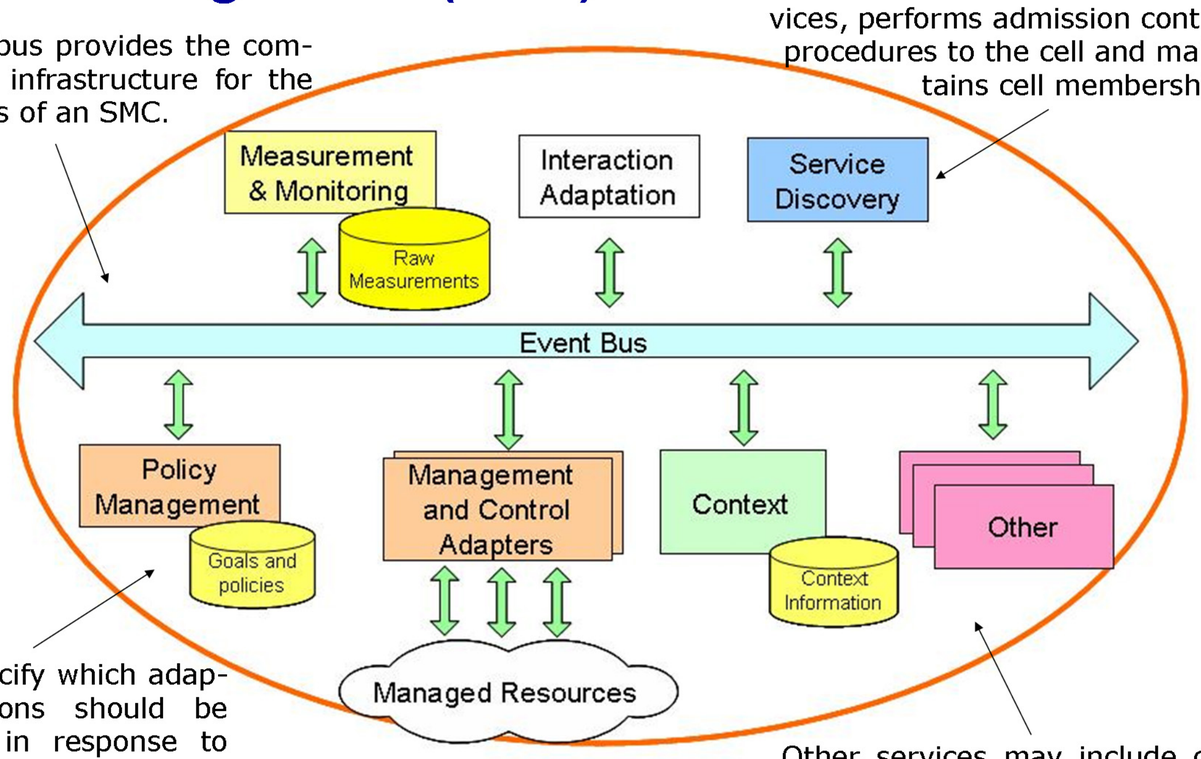
- Adapting the frequency of measurements in order to optimise power consumption.
- Changing the monitoring thresholds depending on the user's context.

We advocate the concept of self-managed cell (SMC) as a means of managing body-sensor networks. An SMC consists of hardware and software components which form an administrative domain capable of functioning autonomously.

The Self-Managed Cell (SMC)

The event bus provides the communication infrastructure for the components of an SMC.

Discovers new devices and services, performs admission control procedures to the cell and maintains cell membership.



Policies specify which adaptation actions should be performed in response to events and how the system should adapt to changes in the context of the environment.

Other services may include context, accounting, resource planning, optimisation and analysis.

The *Ponder2* Policy Service

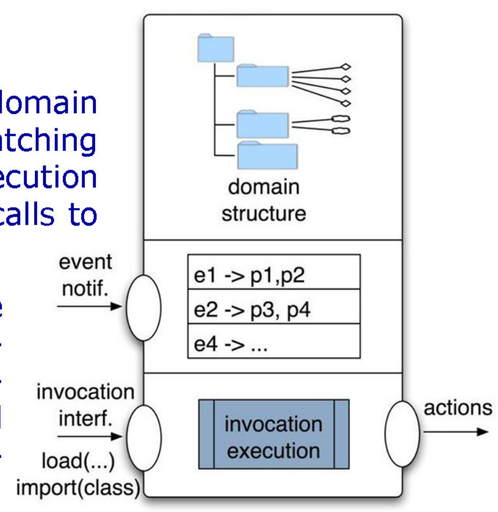
www.ponder2.net

Ponder2 combines a general purpose policy-based management system with a ***domain service***, ***obligation policy interpreter***, and a ***command interpreter***.

- ***Domain Service*** — provides a hierarchical structure for managed objects which represent sensors, actuators, other SMC devices, and remote SMCs
- ***Obligation Policy Interpreter*** — implements Event-Condition-Action (ECA) rules
- ***Command Interpreter —*** accepts commands in XML form via several communication interfaces. Commands invoke operations on managed objects including the domains and the policies themselves.

The *Ponder2* architecture comprises the domain structure, the triggering mechanism matching events to obligation policies and the execution invocation engine which is used to make calls to the objects inside the domain structure.

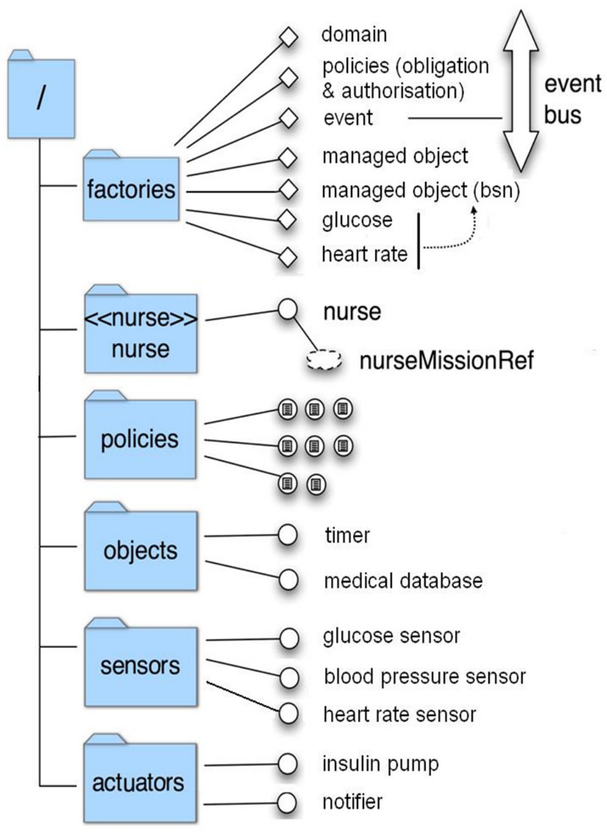
The policy service has an event interface through which event notifications are received from the external event bus, an invocation interface through which external invocations are received and an action interface for invocations on external objects.



***Ponder2* architecture**

Ponder2 can load all the code needed on demand. This includes factories for creating new object adapters (managed objects) for devices and remote services.

The policy service can use multiple communication protocols, such as UDP, Bluetooth, IEEE 802.15.4 and ZigBee.



Example of Domain Structure

An ***event factory*** implements the interface with an external event bus and encapsulates the protocols necessary to communicate with it.

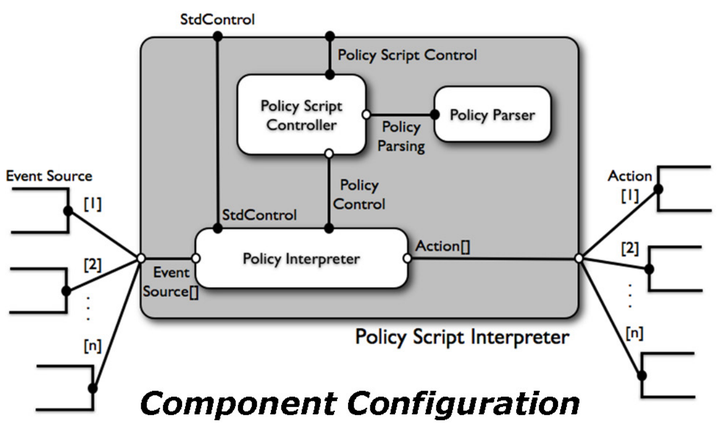
The event factory is used to create new Event Types that issue subscriptions to the external content-based event bus. When an event matching the subscription occurs, the event factory is notified and it raises the corresponding event type in order to trigger the policies.

Authorisation policies — define what actions are permitted under given circumstances.
Obligation policies — define what actions to carry out when specific events occur if the specified condition is true.

The policy becomes active as soon as it is created. Policies can be *enabled, disabled,* and *removed* dynamically.

Ponder2 provides a complete management system for body-sensor networks, but it is not suitable for running on constrained body-sensors.

The Tiny Policy Interpreter



Component Configuration

Provides sensors with programmable local decision capability. It is a simplified policy interpreter running on TinyOS that can be deployed to BSN devices.

The ***policy script controller*** is responsible for loading, adding and removing policies.

The ***policy interpreter*** is invoked to execute the actions according to the deployed obligation policies when event occurs.

Tiny policies are akin to obligation policies in *Ponder2.* They are specified as ECA rules. Events represent samples from sensors, conditions are represented as inclusive ranges. Event sources and actions are represented by NesC interfaces and they are pre-defined as arrays bound to the Tiny Policy Interpreter.

Tiny policy syntax is simpler and compact. Ponder2 policies can be translated into Tiny policies before deployment to the sensors.

Policy Syntax

policy = event condition "->" action

event = uint8 "?"

condition = equals_condition
| in_range_condition
| less_than_condition
| greater_than_condition
| always_condition

in_range_condition = "["min:uint32 ".." max:uint32"]"
always_condition = "always"
greater_than_condition = ">=" min:uint32

action = do_action | set_property_action

do_action = uint8 "!" action_arg?

action_arg = "(" uint32 ")"

For example, the policy 2? [5..9] -> 1! means that if event 2 fires with a parameter between 5 and 9 inclusive, then perform action 1.

AMUSe

Autonomic Management of Ubiquitous Systems for e-Health

www.dcs.gla.ac.uk/amuse