GaSMaC

Great all-purpose
Self Managed Cell

Kevin Twidle
Nov 2005

What is a Self Managed Cell?

- A set of hardware or software components forming an administrative domain that is able to function autonomously and thus capable of self-management. (AMUSE)
- Management services interact with each other through asynchronous events propagated through the system.
- Able to interact with other SMCs and Web Services

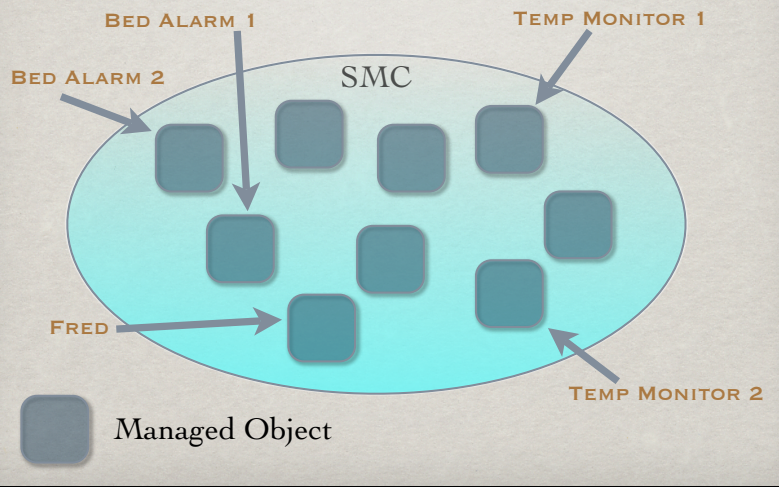
SMCs do what?

- General Purpose Object Management Environment
- SMC could represent
  - the resources available in a PDA
  - a body area network of physiological sensors and controllers
  - application components relating to a set of collaborating partners forming a virtual (e-Health) organisation spanning multiple countries.

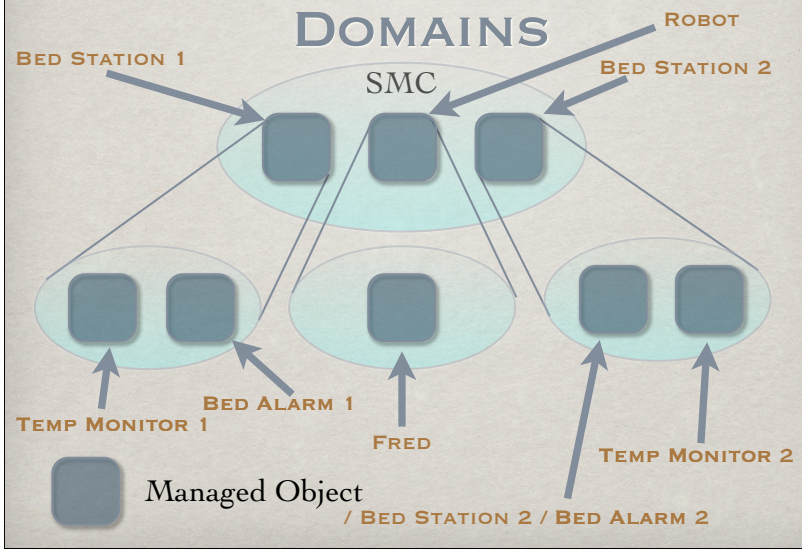
Managed Objects

- A Managed Object is anything that conforms to the SMC interface rules
- Four built-in types of Managed Objects
  - Domains, Policies, Templates, External
- Managed Objects can accept commands from other Managed Objects

SMC Design: MOs have names



SMC Design: Domains



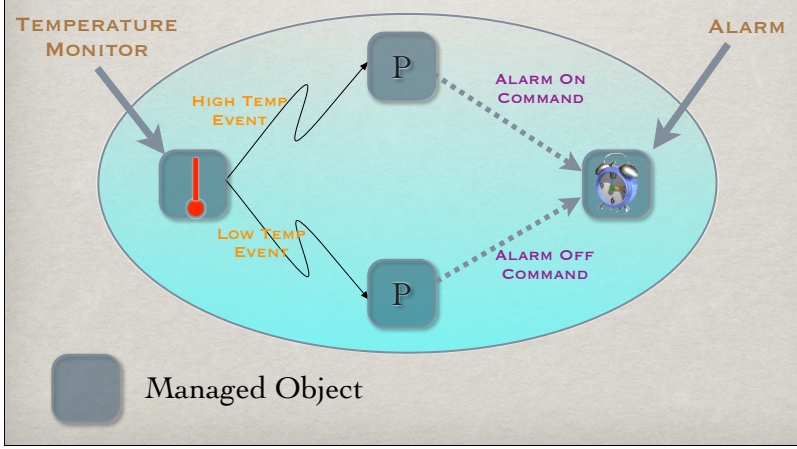
Events

- Event: Notification with named values
- Managed Objects create Events
- Events are picked up by Policies

Policies

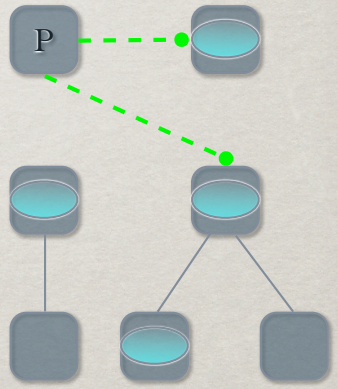
- Policies are Event, Condition, Action rules
- Action: Generate new Event(s) and/or give commands to one or more Managed Objects
- Three basic policy types
  - Access Control Policy
  - Obligation Policy
  - Relationship Policy

Events and Policies

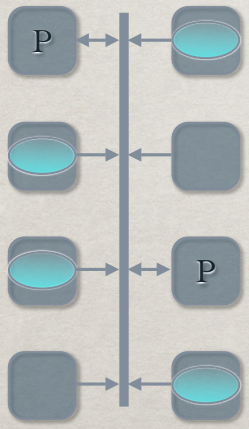


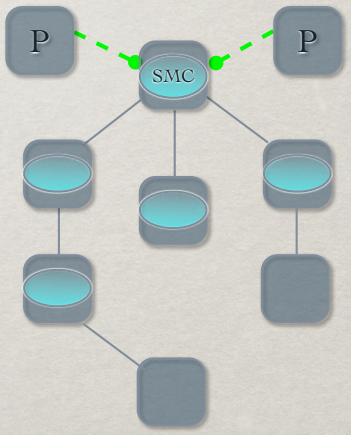
Event Propagation

- Events bubble up the domain hierarchy
- Policies are attached to Managed Objects
- Fine grain control with simple Policies
- Should Events be consumed?



Event Propagation Event Bus Model





Template Managed Object

- Used to create a new Managed Object
- Accepts a "create" command and returns a new instance of a Managed Object.
- Importing new Managed Object code (currently only from a Jar or Java class file) produces a Template Managed Object.

External Managed Object

- Importing a Managed Object from outside the SMC produces an External Managed Object.
- EMO passes all commands to the remote invocation of the Managed Object.
- Results are passed back as though the EMO had executed the command.

The Shell (and a demo!)

- Simple Unix-like* shell interface
- Telnet to port 13570
- Commands include: ls, cd, mkdom, rmdom, ln, dump, restore, read
- XML is terminated by a period. (American for .)

* Unix Version 6 - 1976

XML! Basic Commands

- SMC parses and executes XML
- Few basic commands
- IMPORT
- USE
- Made richer by the commands that Managed Objects obey

Managed Object Commands

- Domain: Add, Link, Remove, List
- Template Object: Create
- Policy Object: Activate, Event, Condition, Action
- Monitor: Threshold, Show, Hide
- TickManager: Tick, Cancel

Syntax for Operations

```
<use name="/pathname/of/managed/object" arg1="value1">
  <operation1 arg1="value1" arg2="value2" argn="...">
    <oparg1 arg1="value1" arg2="value2" argn="...">
      <oparg1arg1 arg1="value1" arg2="...">
        ...
      </oparg1arg1>
      <oparg1arg2 arg1="value1" arg2="...">
        ...
      </oparg1arg2>
    </oparg1>
    <oparg2 arg1="value1" arg2="value2" argn="...">
      ...
    </oparg2>
  </operation1>
  <operation2>
    ...
  </operation2>
</use>
```

Add to a Domain

```
<use name="/common_printers">
  <add name="4thFloorPrinter">
    <use name="/department/printers/hp5040n"/>
  </add>
</use>
```

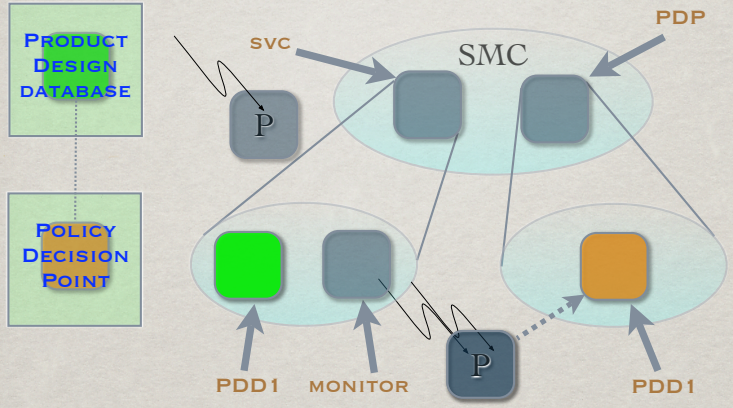
Obligation Policy

```
<use name="/Policy">
 <add name="activateAlarmedPDP">
  <use name="/Template/policy">
   <create type="obligation" event="/Event/repLT50"
active="true">
    <arg name="name"/>
    <arg name="value"/>
    <condition>
     <not>
      <use name="/Policy/alarmedPDP">
       <isactive/>
      </use>
     </not>
    </condition>
    <action>
     <use name="/Policy/alarmedPDP" active="true"/>
     <trace> Policy on event !name; !value; </trace>
    </action>
   </create>
  </use></add></use>
```

TrustCom Demo

- Part of demo given in Brussels
- Shows Policies, Events, External Objects

TrustCom Demo



Bootstrap Demo

- SMC is just an empty Domain
- Import Domain Template
- Create Domain
- Be Happy

To Do

- Deletion semantics
- External references with Dump and Restore
- More external protocols
- Freeze and Restart systems
- General JAVA Swing Managed Object
- Access Control Policy

Tutorial

- Two Tutorials
  - XML Based
  - Java Based
- See tutorial sheet

Programming a Managed Object

- Create
  Creates and initialises a managed object
- Execute Setup
  Reads the setup parameters
- Execute
  Executes operations on the managed object
- Get State
  Dumps current state for later restoration

Programming: Create

```
/**
 * Creates and initialises a NullManagedObject
 *
 * @param xml
 *          the initialisation parameters and commands
 * @param result
 *          the general result structure for errors and complex results
 * @return the new managed object
 */
public static NullManagedObject create(TaggedElement xml, Result result) {
  // May return one of several different sub-types. c.f. Policy
  return new NullManagedObject(xml, result);
}

/**
 * creates an instance of this managed object
 *
 * @param xml
 *          the initialisation parameters and commands
 * @param result
 *          the general result structure for errors and complex results
 */
public NullManagedObject(TaggedElement xml, Result result) {
  super();
  execute(xml, result);
}
```

Programming: Execute Setup

```
/*
 * (non-Javadoc)
 *
 * @see org.trustcom.ManagedObject#executeSetup (com.twicom.qdparser.TaggedElement,
 *      org.trustcom.comms.Result)
 */
@Override
public boolean executeSetup(TaggedElement xml, Result result) {
  // This method is optional
  //
  // Check attributes
  String att = xml.getAttribute("myattribute");
  if (att != null) {
    // Do something
  }
  return true; // if we are happy else return false
}
```

Programming: Execute

```
public boolean execute(TaggedElement xml, TaggedElement command, Result result) {
 // Check the operations and execute them
 /* This example will respond to:
  * <create myattribute="pling"><op1 att1="value1"><op1op/></op1></create>
  * or
  * <use name="/some/name" optionalattribute="a1"><op2 att1="now"/></use>
  */
 String operation = command.getName();
 if (operation.equals("op1")) {
   // Cycle through the sub-elements of the command
   for (Object o : command) {
     if (o instanceof TaggedElement) {
       TaggedElement subop = (TaggedElement)o;
       // do something with subop, e.g. getAttribute
     }
   }
 }
 else if (operation.equals("op2")) {
   // Do something for op2
 }
 else
   // Indicate that we have not recognised the operation
   return false;
 // Indicate that we recognised the operation, no further processing needed
 return true;
}
```

Programming: Get State

```
/*
  * (non-Javadoc)
  *
  * @see org.trustcom.ManagedObject#getState(com.twicom.qdparser.TaggedElement)
  */
 @Override
 protected TaggedElement getState(TaggedElement state) {
   // Code to sucessfully recreate this object at a later time
   /*
    * Any state written out here as attributes or operations MUST be acted upon
    * in executeSetup or execute
    *
    * This example will return
    *
    * <state myattribute="some value"><op2 att1="now"/></state>
    */
   state.setAttribute("myattribute", "some value");
   TaggedElement op2 = new TaggedElement("op2");
   op2.setAttribute("att1", "now");
   state.add(op2);
   return state;
 }
```