Introduction to Pervasive Computing

Our Purpose

- This is an OS course so why talk about pervasive computing?
  - Pervasive computing depends on OS support
  - Pervasive computing is an interesting area and one of growing importance
    - The bulk of all processors manufactured and deployed are now in embedded applications (like pervasive computing)
    - We need to be able to provide OS services which are appropriate to use in pervasive computing scenarios
- This introduction will:
  - Introduce you to pervasive computing ideas
- This introduction will not:
  - Talk about OS specifics related to pervasive computing
    - That will be up to you to do in the papers you present

What is Pervasive Computing?

- Also known as Ubiquitous Computing
  - Though some distinguish between the two
- The key word here is "pervasive" (or "ubiquitous")
  - Definitions of pervasive:
    - *"spread throughout", "Having the quality or tendency to pervade or permeate"*
  - Definitions of ubiquitous:
    - "being present everywhere at once", "constantly encountered"
- So pervasive computing is any scenario where you have computing integrated into the environment
- According to Mark Weiser, 1991:
  - "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it."

What is Pervasive Computing? (cont'd)

- Another important characteristic of pervasive computing is that many of the devices you interact with are embedded
  - Back to Weiser's vision
- Another common idea you will hear when discussing pervasive computing is that it is the "next" computing paradigm
  - Mainframe (with many users per machine) evolved to the PC (with one user per machine) which evolved to Pervasive Computing (many machines for a single user)
  - Not necessarily the best analogy and a fair bit of hype

What is Pervasive Computing? (cont'd)

- Pervasive computing spans many CS disciplines:
  - Mobile computing,
    - Many pervasive devices are mobile
  - Distributed systems,
    - Multiple machines cooperate to solve a single problem
  - Human Computer Interfaces, and
    - Special UI techniques are sometimes needed to deal with the characteristics of the environment
  - Operating Systems
    - OS must provide support for all aspects of pervasive computing
- Pervasive computing also intersects Engineering:
  - Wireless network access
  - Small and special purpose HW devices
  - Sensors and controls

Pervasive Applications

- A good way to start understanding pervasive computing is by looking at some pervasive applications
  - As we do this, think about some of the underlying capabilities that might be required to support such applications
    - This is where the interesting OS issues come in
- Scenario 1: You are giving a conference presentation
  - You walk into the conference room
    - Your laptop establishes itself in the wireless network and identifies the available devices (e.g. projector, sound system, etc.)
  - The time for your presentation arrives
    - Your laptop automatically takes control of the projector, dims the room lighting, etc. and starts the presentation
  - After the presentation
    - Your laptop exchanges contact information with the laptops of other researchers in the room who are interested in your work

Pervasive Applications (cont'd)

- Scenario 2: You are watching TV in your family room
  - Someone arrives at the front door
    - A sensor detects their arrival and activates a streaming "web cam" monitoring the entrance
  - Sensors in the home determine that you are watching TV in the family room and may not wish to go to the door
    - An on-screen window (PiP) appears showing the output from entrance
    - You decide it is a salesperson and choose to watch TV
    - You use your remote to indicate disinterest causing the PiP window to disappear and playing a recording at the door asking the person to leave materials in the mailbox

Pervasive Applications (cont'd)

- Scenario 3: You are visiting the mall with friends
  - You make a purchase of a winter coat at store X
    - Your location and purchase are recorded by the store
  - Later you pass store Y, an affiliate of store X
    - Your mobile device (e.g. Blackberry, PDA, etc.) announces a special sale for you alone on winter boots that will match your new coat
- Scenario 4: You are on vacation travelling by car
  - In the mountains you must use your brakes frequently
    - Sensors detect a drop in pressure in the brake lines of your vehicle and diagnose a slow leak
    - Your travel itinerary indicates your next stop is Banff and an appointment is made for you to have your brakes serviced that night
      - The necessary parts are pre-ordered for the service vendor
    - Your in-car display informs you to drive carefully and get service

Pervasive Applications (cont'd)

- Scenario 5: You are a student doing homework
  - Your teacher has assigned a research project
    - The project is entered into your personal agenda and scheduled and you are informed of this
  - You arrive home and boot up your PC
    - Information is transferred to the PC from your mobile and at the scheduled time a browser is started executing a "google" search on the assigned topic and a skeleton report is started in your word processor
- Scenario 6: You are an AD patient living at home
  - At 17:00 you are reminded to prepare supper and given a series of suggested meals
  - As you remove items from the refrigerator, a "stale" ingredient is detected and you are told not to use it
  - Step by step instructions and active safety monitoring are provided

Pervasive Applications (cont'd)

- Scenario 7: You arrive home from work
  - You turn onto your street
    - The lights turn on, the garage door opens and the door to the house unlocks
  - You enter the home
    - The interior lights and temperature adjust to your preferences and the TV turns on tuned to the station you typically watch at this time on this day
- Scenario 8: You experience health difficulties
  - While conducting a normal task, you exhibit unusual behaviour
    - The environment inquires as to your situation
  - You respond in a disoriented fashion
    - The environment directs you to a medical monitoring station
  - Your vital signs are taken and sent to the health authority
    - If necessary, appropriate medical interventions are initiated
      - E.g. ambulance is called, etc.

Pervasive Computing Characteristics

- From the example applications we can observe the following key characteristics:
  - Embedded
    - Don't need special computing devices (ubiquitous, in-built)
      - Doesn't preclude such devices but ...
  - Transparency
    - User is not involved in underlying operations (just use)
  - Context sensitivity
    - Operations are aware of the "context" in which they occur
  - User focus
    - Pervasive computing is very much human-centric
  - Automatic
    - User doesn't necessarily need to ask for work to be done

Embedded Devices

- Many of the "computing" devices that are present in a pervasive computing environment are embedded
  - i.e. not directly visible to the user
  - E.g. sensors and control devices in the home
- Such devices face many of the challenges we discussed for sensor networks
  - E.g. severe resource constraints
- In some cases the challenges may, of course, be different from sensor networks
  - E.g. in-home sensors may run on AC power, also some devices are special purpose

Embedded Devices (cont'd)

- A primary challenge is integrating such devices into a pervasive computing system
  - Communication – what protocols can be used?
    - A thermostat likely doesn't run Java and support RMI over a TCP stack
  - Interoperability – how can different standards be supported?
    - Many different manufacturers and no strong standards (yet)
- Other challenges include:
  - Reliability – can we tolerate failures
    - What else breaks when a device in a pervasive environment fails?
  - Security – once something communicates ...
    - Can someone change your house temperature from outside?
      - Ala theft of wireless Internet service
  - Remote Access
    - E.g. remotely turn up the heat an hour before returning from vacation

Transparency

- Transparency is a familiar concept in Operating Systems research
  - E.g. device transparency in the Unix file system or location transparency with NFS
- In pervasive computing, transparency is paramount
  - This makes sense since many participating devices are embedded and also suits the idea of human-centric computing
- Achieving transparency is key to acceptance since the target audience is non-technical
  - Not you, but your grandmother or child, etc.
  - Users do not want to be aware of the devices they are interacting with
    - Computing should be "natural" (unobtrusive) in pervasive environments

Transparency (cont'd)

- A key challenge to providing transparency is the need for devices to automatically configure themselves to provide useful functions
  - The user cannot (or does not want to) be involved in setting up a pervasive environment
    - Need something along the lines of "autonomic computing" for handling self configuration
  - Further, with mobile devices, pervasive environments are typically too dynamic to permit the use of trained service personnel (i.e. tech support) to configure them
    - Though this would likely be cost prohibitive anyway
  - New ideas and mechanisms are required here
    - We'll talk a bit about this later

Context Sensitivity

- In many cases, the behaviour exhibited by a pervasive environment depends on the situation
  - This is known as being "Context Sensitive"
- There are many different possible types of context sensitivity depending on what affects behaviour:
  - Functional context
    - Based on what you are doing (e.g. context sensitive help)
  - Temporal context
    - Based on current date and/or time
  - Spatial context
    - Based on location – can be geographic (ala GPS) or logical ("at work")
  - User context
    - Based on who is involved (may be multiple persons)

Context Sensitivity (cont'd)

- Context sensitivity helps a pervasive environment react in the right way to different situations
  - E.g. different people have different environmental preferences at different times of the year
- Key challenges in context sensitivity include:
  - Gathering relevant information to base decisions on
    - E.g. data mining applied to historical user behaviour patterns
  - Deciding which context to use in a given situation
    - Sometimes using incorrect context results in poor choices and considering too much context information can decrease performance
  - Dealing with multiple, conflicting contexts concurrently
    - E.g. I like the house warm and my wife likes it cold so how do we pick an appropriate temperature level when we both arrive home together?

User Centric Behaviour

- Much more than traditional computing systems, pervasive environments are very user centric
  - Everything is focused on catering to the user
- This reflects itself most naturally in the user interface (UI) which often must be tailored to the user
  - Different users will have different abilities/skill-sets and a single, rigid user interface is often inappropriate
  - Further, pervasive environments are dynamic so user interface selection may have to be done "on the fly"
- Additionally, other constraints are also induced
  - E.g. QoS/QoE restrictions such as delay to availability when entering a new environment
    - These can present very hard challenges

Service Automation

- While it is quite possible that users may explicitly request particular services, it is just as likely that they may expect the pervasive environment to provide them automatically on their behalf
  - Even when services are explicitly invoked, they have to be available and this is non-trivial when users move into and out of environments at will
- A key challenge is deciding what to automate and what not to automate
  - E.g. turning the lights on at 6:30 on Saturday morning because someone got up then last Monday through Friday might not be appreciated

Pervasive Systems & Services

- "Services" are a key technology to help enable the construction of pervasive computing systems:
  - Service libraries
  - Service discovery
  - Service composition
- Many of these technologies are provided via middleware packages
  - Interoperability protocols
    - E.g. UPnP, Jini, OSGi
- We'll briefly look at services before enumerating some other challenges in pervasive computing

Service Libraries

- It is inefficient to build anything from scratch
  - This is part of the reason why we have OSs
- A common practice in pervasive environments is to view everything that can be done as a "service"
  - Whether it is implemented in hardware or software
- Not surprisingly then, it is common to see collections ("libraries") of services/components being provided
  - Essentially provide low-level building blocks for developing pervasive applications
  - Provide abstractions for common components
    - Service creation, deployment, advertising, discovery, initialization, invocation, shutdown, deletion, etc.

Service Discovery

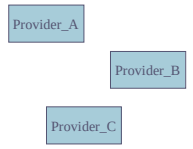
- Before a service can be used, it must be known
- The process of finding a service is commonly referred to as "service discovery"
  - Some entity provides a service that some other entity needs and the latter must "discover" the existence of the service offered by the former
- Two common approaches to service discovery may be used:
  - Broadcast based
  - Directory based

Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders

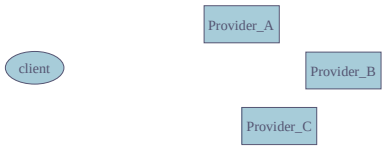
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



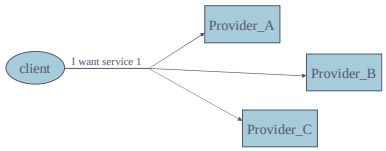
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



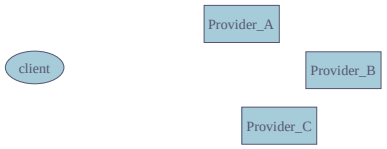
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



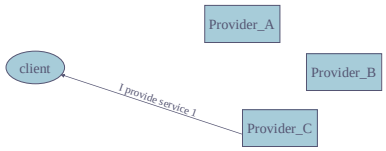
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



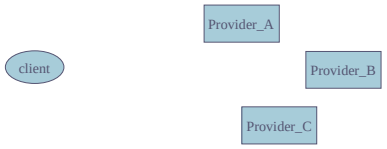
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



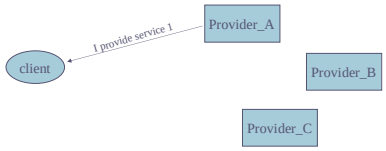
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



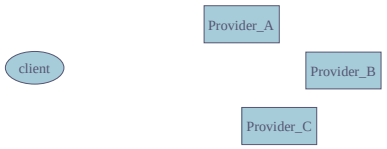
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



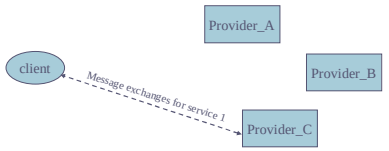
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



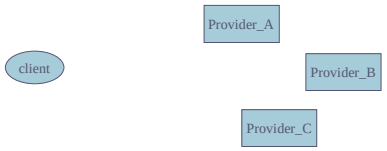
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



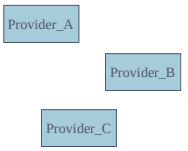
Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders



Service Discovery (cont'd)

- In broadcast (or multicast) based service discovery, potential clients will send out a request for a service they wish to use via broadcasting or multicasting
- A provider of the needed service responds and the client will select between the responders

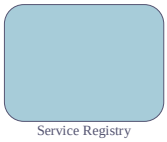


Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry

Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry

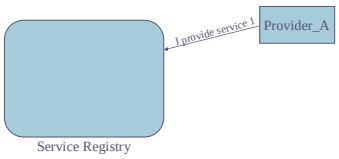


Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry

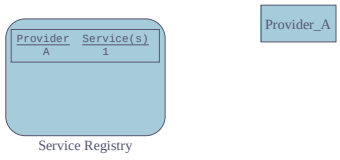
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



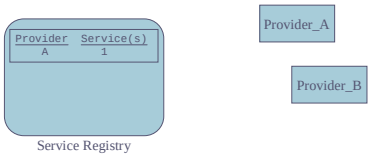
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



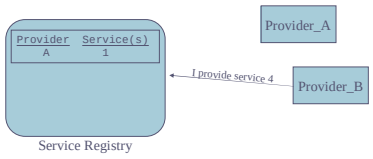
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



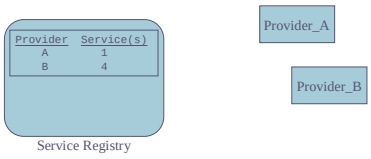
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



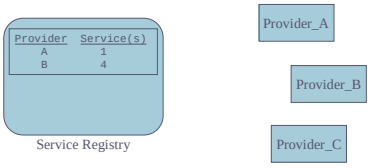
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



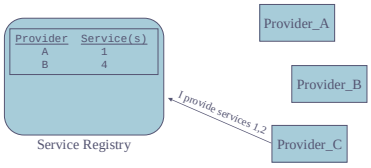
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



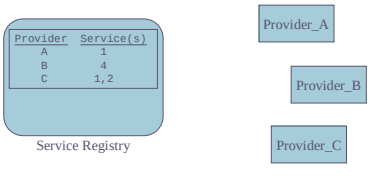
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



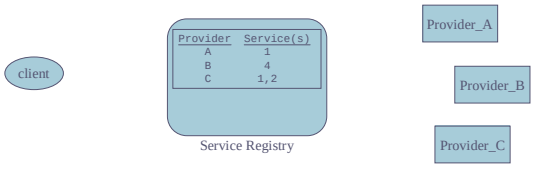
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



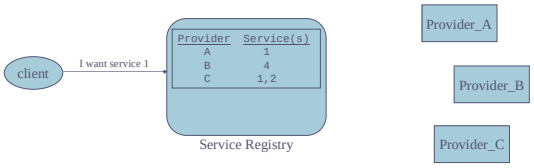
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



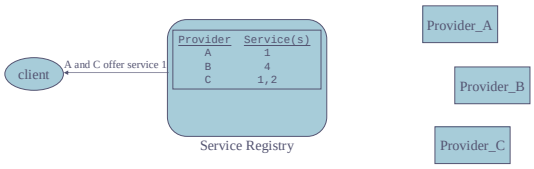
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



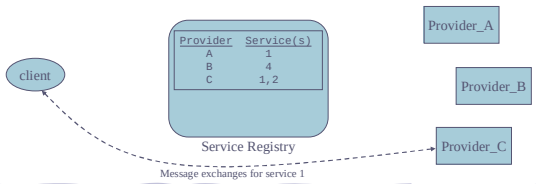
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



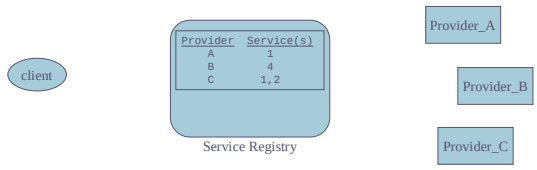
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



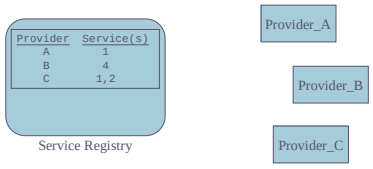
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



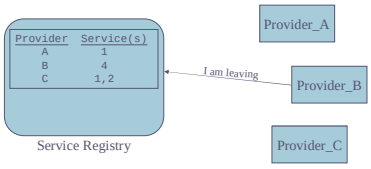
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



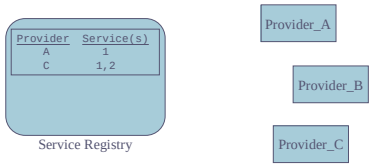
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



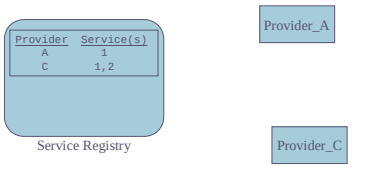
Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



Service Discovery (cont'd)

- In directory based service discovery, providers pre-register the services they offer with a third party (the service registry/directory)
- Clients then request service information from the registry



Service Discovery (cont'd)

- Regardless of the type of service discovery used, there are a number of issues to be addressed:
  - Service Identification – How do we identify a service of interest? By name or by function?
    - Service Naming – How do we name services and how can we assume a global agreement on naming?
    - Service Description – How do we describe what a service does and how can we assume global agreement on such descriptions?
      - INS revisited?
  - Service Selection – How do we choose between multiple responders
    - Which service is best? What is "best"?
  - Services and QoS – Can we incorporate non-functional parameters into service selection?
    - E.g. cost, reliability, latency, etc.

Service Discovery (cont'd)

- Other issues related to service discovery include:
  - Service Reliability – How do we deal with service failures?
    - Complicated by the presence of mobile devices which may leave and take their services with them
    - Possibly use leases whereby participants must renew lease for services periodically to ensure the "other end" is still present and willing to participate
  - Service Trustworthiness – How do we know that a service provided by an unknown device is trustworthy?
    - Particularly problematic in pervasive environments where there is great mobility, many new/unknown participants and few trusted third parties
    - Possibly use trust models where opinions of others are considered to assess unknown parties' trustworthiness
  - Privacy – we lose privacy when we are involved in service discovery
    - Sometimes this may be an issue

Service Composition

- Once a service has been discovered, it can be used
- But what if there isn't a single service to perform a function the user might want to do?
- It is possible that such a function might be implemented using two or more available services
- In such cases, we need a mechanism whereby this can be determined automatically
- Such systems are said to provide "Service Composition"
  - We "compose" two or more services to provide a "composite service" which may itself be composed with other services

Service Composition (cont'd)

- The biggest challenge in service composition is fully automating the process with no user involvement
  - Few systems provide anything close to this yet
  - Transparency is an identified requirement for pervasive systems
- In general, we must take specifications of the services and somehow determine that they can be combined to provide useful functions
- This can be done in at least two ways:
  - I/O compatibility matching
  - Repository/template based matching

Service Composition (cont'd)

- In I/O compatibility matching, two services are composable if the output of the first service is compatible with the input to the second service
  - That is, if the second service can accept what is produced by the first
    - We'll worry about physically connecting the two services separately
      - "Glue logic"
- Of course what is and is not "compatible" and how efficiently we can judge this largely determines the practicality of this approach
  - There are different notions of "compatible" and different criteria upon which to base compatibility

Service Composition (cont'd)

- In general, semantic descriptions of services may be used to improve the matching process
  - Define a pre-agreed upon type hierarchy ("ontology") relating different types of information produced/consumed by the various services
- This enables more flexible matching than simply using names (i.e. syntactic matching)
- In particular, we can use relationships in the type hierarchy to permit partial matching
  - E.g. where we expect a `PS_file` we can probably accept a `monochrome_PS_file`
    - Which would presumably be a sub-type of `PS_file`

Service Composition (cont'd)

- The process of using I/O compatibility to determine service compositions is complicated by a number of factors:
  - Sequences may be longer than just two sub-services
  - Sub-services may be combined in ways other than just sequences
    - What about conditionals, parallel invocations, repetition, etc.?
  - The large number of potential types and services and hence, the associated cost of matching
    - Further increased when conditionals, etc. are also considered
  - Are all I/O compatible composite services useful?

Service Composition (cont'd)

- In repository based matching, a repository of "known" composite services is maintained
  - Possibly pre-created by hand or created using I/O based matching and subsequently proven to be useful
- The problem then becomes one of matching whatever services are available in a pervasive environment against patterns in the repository
  - Still a numbers game since there will be many possible compositions
  - Matching should still be semantic
  - Can't deal with entirely new sub-services
    - Need I/O matching or human intervention for this

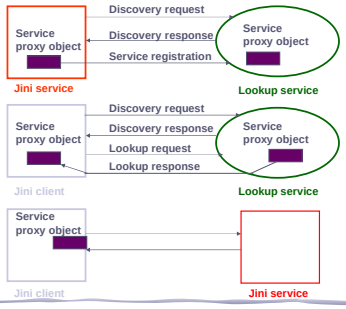
Pervasive Middleware

- A common problem with creating pervasive applications, especially dynamically, is the heterogenous nature of the devices and services
  - Few available standards, multiple device vendors, etc.
- To be able to make effective use of all the services in a given environment and to be able to use service composition we must provide support for interoperability
- Solving this problem is largely the domain of pervasive middleware such as UPnP and Jini, etc.

Jini

- Jini (from Sun Microsystems)
  - Is a directory-based SDP
    - Services must register themselves in the directory
  - Uses Java interfaces to advertise/lookup a service
  - Users of services can download actual implementation of the interfaces to those services
  - Clients use Remote Method Invocation to access the provided service
  - But, a Java Virtual Machine is required on every participating device
    - What about small devices, etc?

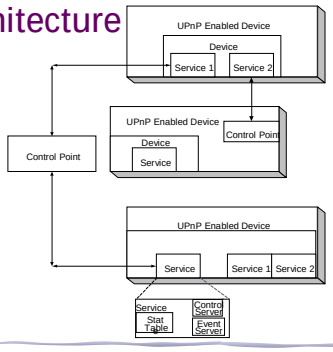
Jini: Service registration & discovery



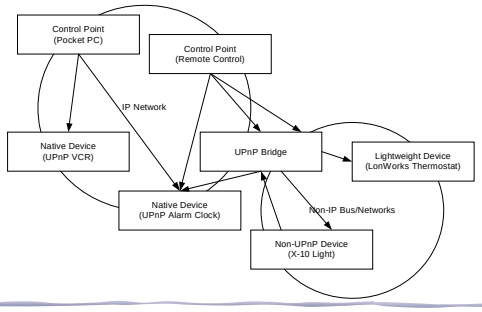
UPnP

- UPnP (from Microsoft)
  - "Universal Plug and Play"
  - Uses multi-cast for service announcement and lookup
  - UPnP devices are logical containers for one or more services or other devices
  - UPnP uses XML to specify service characteristics including their inputs and outputs
  - Uses XML-based RPC over HTTP (via SOAP) to invoke looked-up services
  - Uses events to send a device's "status change" information back to a client

UPnP Architecture



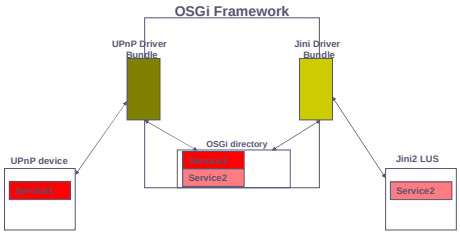
UPnP Configuration



OSGi

- Open Service Gateway initiative (OSGi) is an attempt to provide interoperability between different protocols (e.g. between UPnP and Jini)
- The main concept in OSGi is the *bundle*.
  - A bundle is needed for deploying a code
    - Provides code manifest and specifies methods for start and stop
  - A bundle is a Java program that has the required resources to provide some services to other bundles in the OSGi framework
- OSGi has a directory of services shared between different bundles
  - Thus providing interoperability

Interoperability in OSGi



Pervasive Computing – Other Issues

- There are a large number of other issues associated with pervasive computing including:
  - Security – how do we ensure we keep our data private?
  - Privacy – how do we prevent exposing our behaviour in ways we do not want to?
  - Social issues – are we creating dependency on machines?
  - Adaptivity – how do we adapt to support difference individuals?
    - E.g. persons with disabilities
  - Etc, etc, etc.